```
; ****************************************************************************
; TRDOS386.ASM (TRDOS 386 Kernel) - v2.0.0
; ----------------------------------------------------------------------------
; Last Update: 22/07/2017
; ----------------------------------------------------------------------------
; Beginning: 04/01/2016
; ----------------------------------------------------------------------------
; Assembler: NASM version 2.11 (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)
; ****************************************************************************

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

;; 27/12/2013
;KEND    equ KLOAD + 65536 ; (28/12/2013) (end of kernel space)
; 04/07/2016
KEND    equ KERNELFSIZE + KLOAD



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

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

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

```
                                ; 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. Alsways 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)


                                [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[F25C]                   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[EE5C]                   mov   [mem_1m_1k], cx
00000017 8916[F05C]                   mov   [mem_16m_64k], dx
                                      ; 05/11/2014
                                      ;and  dx, dx
                                      ;jz   short L2
0000001B 81F90004                       cmp     cx, 1024
0000001F 7351                         jnb   short L0
                                             ; insufficient memory_error
                                             ; Minimum 2 MB memory is needed...
                                      ; 05/11/2014
                                      ; (real mode error printing)
00000021 FB                           sti
00000022 BE[3600]                     mov   si, msg_out_of_memory
00000025 BB0700                       mov   bx, 7
00000028 B40E                         mov   ah, 0Eh      ; write tty
                                oom_1:
0000002A AC                           lodsb
0000002B 08C0                         or    al, al
0000002D 7404                         jz    short oom_2
0000002F CD10                         int   10h
00000031 EBF7                         jmp   short oom_1
                                oom_2:
00000033 F4                             hlt
00000034 EBFD                         jmp   short oom_2

                                ; 20/02/2017
                                ; 05/11/2014
                                msg_out_of_memory:
00000036 070D0A                       db    07h, 0Dh, 0Ah
00000039 496E73756666696369-            db      'Insufficient memory !'
00000042 656E74206D656D6F72-
0000004B 792021
0000004E 0D0A                         db    0Dh, 0Ah
                                _int13h_48h_buffer: ; 07/07/2016
00000050 284D696E696D756D20-          db    '(Minimum 2MB memory is needed.)'
00000059 324D42206D656D6F72-
00000062 79206973206E656564-
0000006B 65642E29
0000006F 0D0A00                       db    0Dh, 0Ah, 0
```

```
                                        ;
                                    L0:
                                    %include 'diskinit.s' ; 07/03/2015
                                <1> ; ****************************************************************************
                                <1> ; TRDOS386.ASM (TRDOS 386 Kernel) - v2.0.0 - diskinit.s
                                <1> ; ----------------------------------------------------------------------------
                                <1> ; Last Update: 09/07/2016
                                <1> ; ----------------------------------------------------------------------------
                                <1> ; Beginning: 24/01/2016
                                <1> ; ----------------------------------------------------------------------------
                                <1> ; Assembler: NASM version 2.11 (trdos386.s)
                                <1> ; ----------------------------------------------------------------------------
                                <1> ; Turkish Rational DOS
                                <1> ; Operating System Project v2.0 by ERDOGAN TAN (Beginning: 04/01/2016)
                                <1> ;
                                <1> ; Derived from 'Retro UNIX 386 Kernel - v0.2.1.0' source code by Erdogan Tan
                                <1> ; diskinit.inc (10/07/2015)
                                <1> ;
                                <1> ; Derived from 'IBM PC-XT-286' BIOS source code (1986)
                                <1> ; ****************************************************************************
                                <1>
                                <1> ; Retro UNIX 386 v1 Kernel - DISKINIT.INC
                                <1> ; Last Modification: 10/07/2015
                                <1>
                                <1> ; DISK I/O SYSTEM INITIALIZATION - Erdogan Tan (Retro UNIX 386 v1 project)
                                <1>
                                <1> ; //////// DISK I/O SYSTEM STRUCTURE INITIALIZATION ///////////////
                                <1>
                                <1>       ; 10/12/2014 - 02/02/2015 - dsectrm2.s
                                <1> ;L0:
                                <1>       ; 12/11/2014 (Retro UNIX 386 v1 - beginning)
                                <1>       ; Detecting disk drives... (by help of ROM-BIOS)
00000072 BA7F00                 <1>       mov   dx, 7Fh
                                <1> L1:
00000075 FEC2                   <1>       inc   dl
00000077 B441                   <1>       mov   ah, 41h ; Check extensions present
                                <1>                     ; Phoenix EDD v1.1 - EDD v3
00000079 BBAA55                 <1>       mov   bx, 55AAh
0000007C CD13                   <1>       int   13h
0000007E 721A                   <1>       jc    short L2
                                <1>
00000080 81FB55AA               <1>       cmp   bx, 0AA55h
00000084 7514                   <1>       jne   short L2
00000086 FE06[F55C]             <1>       inc   byte [hdc]   ; count of hard disks (EDD present)
0000008A 8816[F45C]             <1>         mov     [last_drv], dl  ; last hard disk number
0000008E BB[785C]               <1>       mov   bx, hd0_type - 80h
00000091 01D3                   <1>       add   bx, dx
00000093 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
00000095 80FA83                 <1>       cmp   dl, 83h       ; drive number < 83h
00000098 72DB                   <1>       jb    short L1
                                <1> L2:
                                <1>       ; 23/11/2014
                                <1>       ; 19/11/2014
0000009A 30D2                   <1>       xor   dl, dl  ; 0
                                <1>       ; 04/02/2016 (esi -> si)
0000009C BE[F65C]               <1>       mov   si, fd0_type
                                <1> L3:
                                <1>       ; 14/01/2015
0000009F 8816[F35C]             <1>       mov   [drv], dl
                                <1>       ;
000000A3 B408                   <1>       mov   ah, 08h ; Return drive parameters
000000A5 CD13                   <1>       int   13h
000000A7 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>             ;
000000A9 881C                   <1>       mov   [si], bl ;  Drive type
                                <1>                     ; 4 = 1.44 MB, 80 track, 3 1/2"
                                <1>       ; 14/01/2015
000000AB E8BC01                 <1>       call  set_disk_parms
                                <1>       ; 10/12/2014
000000AE 81FE[F65C]             <1>       cmp   si, fd0_type
000000B2 7705                   <1>       ja    short L4
000000B4 46                     <1>       inc   si ; fd1_type
000000B5 B201                   <1>       mov   dl, 1
000000B7 EBE6                   <1>       jmp   short L3
                                <1> L4:
                                <1>       ; Older BIOS (INT 13h, AH = 48h is not available)
000000B9 B27F                   <1>       mov   dl, 7Fh
                                <1>       ; 24/12/2014 (Temporary)
000000BB 803E[F55C]00           <1>       cmp   byte [hdc], 0 ; EDD present or not ?
000000C0 0F879000               <1>         ja      L10       ; yes, all fixed disk operations
                                <1>                         ; will be performed according to
                                <1>                         ; present EDD specification
                                <1> L6:
000000C4 FEC2                   <1>       inc   dl
000000C6 8816[F35C]             <1>         mov     [drv], dl
000000CA 8816[F45C]             <1>         mov     [last_drv], dl ; 14/01/2015
000000CE B408                   <1>       mov   ah, 08h ; Return drive parameters
000000D0 CD13                   <1>       int   13h    ; (conventional function)
000000D2 0F828601               <1>         jc      L13       ; fixed disk drive not ready
000000D6 8816[F55C]             <1>         mov     [hdc], dl ; number of drives
```

```
                                <1>       ;; 14/01/2013
                                <1>       ;;push cx
000000DA E88D01                 <1>       call   set_disk_parms
                                <1>       ;;pop  cx
                                <1>       ;
                                <1>       ;;and  cl, 3Fh       ; sectors per track (bits 0-6)
000000DD 8A16[F35C]             <1>         mov    dl, [drv]
000000E1 BB0401                 <1>       mov    bx, 65*4 ; hd0 parameters table (INT 41h)
000000E4 80FA80                 <1>       cmp    dl, 80h
000000E7 7603                   <1>       jna    short L7
000000E9 83C314                 <1>       add    bx, 5*4       ; hd1 parameters table (INT 46h)
                                <1> L7:
000000EC 31C0                   <1>       xor    ax, ax
000000EE 8ED8                   <1>       mov    ds, ax
000000F0 8B37                   <1>         mov    si, [bx]
000000F2 8B4702                 <1>         mov    ax, [bx+2]
000000F5 8ED8                   <1>       mov    ds, ax
000000F7 3A4C0E                 <1>         cmp    cl, [si+FDPT_SPT] ; sectors per track
000000FA 0F855A01               <1>         jne    L12 ; invalid FDPT
000000FE BF0000                 <1>       mov    di, HD0_DPT
00000101 80FA80                 <1>       cmp    dl, 80h
00000104 7603                   <1>       jna    short L8
00000106 BF2000                 <1>       mov    di, HD1_DPT
                                <1> L8:
                                <1>       ; 30/12/2014
00000109 B80090                 <1>       mov    ax, DPT_SEGM
0000010C 8EC0                   <1>       mov    es, ax
                                <1>       ; 24/12/2014
0000010E B90800                 <1>       mov    cx, 8
00000111 F3A5                   <1>       rep    movsw  ; copy 16 bytes to the kernel's DPT location
00000113 8CC8                   <1>       mov    ax, cs
00000115 8ED8                   <1>       mov    ds, ax
                                <1>       ; 02/02/2015
00000117 8A0E[F35C]             <1>         mov    cl, [drv]
0000011B 88CB                   <1>       mov    bl, cl
0000011D B8F001                 <1>       mov    ax, 1F0h
00000120 80E301                 <1>       and    bl, 1
00000123 7406                   <1>       jz     short L9
00000125 C0E304                 <1>       shl    bl, 4
00000128 2D8000                 <1>       sub    ax, 1F0h-170h
                                <1> L9:
0000012B AB                     <1>       stosw  ; I/O PORT Base Address (1F0h, 170h)
0000012C 050602                 <1>       add    ax, 206h
0000012F AB                     <1>       stosw  ; CONTROL PORT Address (3F6h, 376h)
00000130 88D8                   <1>       mov    al, bl
00000132 04A0                   <1>       add    al, 0A0h
00000134 AA                     <1>       stosb  ; Device/Head Register upper nibble
                                <1>       ;
00000135 FE06[F35C]             <1>       inc    byte [drv]
00000139 BB[785C]               <1>       mov    bx, hd0_type - 80h
0000013C 01CB                   <1>       add    bx, cx
0000013E 800F80                 <1>         or     byte [bx], 80h  ; present sign (when lower nibble is 0)
00000141 A0[F55C]               <1>       mov    al, [hdc]
00000144 FEC8                   <1>       dec    al
00000146 0F841201               <1>         jz     L13
0000014A 80FA80                 <1>       cmp    dl, 80h
0000014D 0F8673FF               <1>         jna    L6
00000151 E90801                 <1>         jmp    L13
                                <1> L10:
00000154 FEC2                   <1>       inc    dl
                                <1>       ; 25/12/2014
00000156 8816[F35C]             <1>       mov    [drv], dl
0000015A B408                   <1>       mov    ah, 08h ; Return drive parameters
0000015C CD13                   <1>       int    13h    ; (conventional function)
0000015E 0F82FA00               <1>         jc     L13
                                <1>       ; 14/01/2015
00000162 8A16[F35C]             <1>       mov    dl, [drv]
00000166 52                     <1>       push   dx
00000167 51                     <1>       push   cx
00000168 E8FF00                 <1>       call   set_disk_parms
0000016B 59                     <1>       pop    cx
0000016C 5A                     <1>       pop    dx
                                <1>       ; 06/07/2016 (BugFix for >64K kernel files)
                                <1>       ; 04/02/2016 (esi -> si)
                                <1>       ;mov   si, _end ; 30 byte temporary buffer address
                                <1>       ;              ; at the '_end' of kernel.
                                <1>       ;mov   word [si], 30
                                <1>       ; 06/07/2016
0000016D BE[5000]               <1>       mov    si, _int13h_48h_buffer
                                <1>       ; 09/07/2016
00000170 B81E00                 <1>       mov    ax, 001Eh
00000173 8824                   <1>       mov    [si], ah ; 0
00000175 46                     <1>       inc    si
00000176 8904                   <1>       mov    word [si], ax
                                <1>       ; word [si] = 30
                                <1>       ;
00000178 B448                   <1>       mov    ah, 48h       ; Get drive parameters (EDD function)
0000017A CD13                   <1>       int    13h
0000017C 0F82DC00               <1>         jc     L13
                                <1>       ; 04/02/2016 (ebx -> bx)
                                <1>       ; 14/01/2015
00000180 28FF                   <1>       sub    bh, bh
00000182 88D3                   <1>       mov    bl, dl
00000184 80EB80                 <1>       sub    bl, 80h
00000187 81C3[F85C]             <1>       add    bx, hd0_type
0000018B 8A07                   <1>       mov    al, [bx]
0000018D 0C80                   <1>       or     al, 80h
0000018F 8807                   <1>       mov    [bx], al
00000191 81EB[F65C]             <1>       sub    bx, hd0_type - 2 ; 15/01/2015
00000195 81C3[425D]             <1>       add    bx, drv.status
00000199 8807                   <1>       mov    [bx], al
                                <1>       ; 04/02/2016 (eax -> ax)
```

```
408 0000019B 8B4410             <1>       mov   ax, [si+16]
409 0000019E 854412             <1>       test  ax, [si+18]
410 000001A1 7412               <1>       jz    short L10_A0h
411                             <1>                   ; 'CHS only' disks on EDD system
412                             <1>                   ;  are reported with ZERO disk size
413 000001A3 81EB[425D]         <1>       sub   bx, drv.status
414 000001A7 C1E302             <1>       shl   bx, 2
415 000001AA 81C3[265D]         <1>       add   bx, drv.size ; disk size (in sectors)
416 000001AE 8907               <1>       mov   [bx], ax
417 000001B0 8B4412             <1>       mov   ax, [si+18]
418 000001B3 8907               <1>       mov   [bx], ax
419                             <1>
420                             <1> L10_A0h: ; Jump here to fix a ZERO (LBA) disk size problem
421                             <1>        ; for CHS disks (28/02/2015)
422                             <1>       ; 30/12/2014
423 000001B5 BF0000             <1>       mov   di, HD0_DPT
424 000001B8 88D0               <1>       mov   al, dl
425 000001BA 83E003             <1>       and   ax, 3
426 000001BD C0E005             <1>       shl   al, 5 ; *32
427 000001C0 01C7               <1>       add   di, ax
428 000001C2 B80090             <1>       mov   ax, DPT_SEGM
429 000001C5 8EC0               <1>       mov   es, ax
430                             <1>       ;
431 000001C7 88E8               <1>       mov   al, ch ; max. cylinder number (bits 0-7)
432 000001C9 88CC               <1>       mov   ah, cl
433 000001CB C0EC06             <1>       shr   ah, 6  ; max. cylinder number (bits 8-9)
434 000001CE 40                 <1>       inc   ax     ; logical cylinders (limit 1024)
435 000001CF AB                 <1>       stosw
436 000001D0 88F0               <1>       mov   al, dh ; max. head number
437 000001D2 FEC0               <1>       inc   al
438 000001D4 AA                 <1>       stosb        ; logical heads (limits 256)
439 000001D5 B0A0               <1>       mov   al, 0A0h ; Indicates translated table
440 000001D7 AA                 <1>       stosb
441 000001D8 8A440C             <1>       mov   al, [si+12]
442 000001DB AA                 <1>       stosb        ; physical sectors per track
443 000001DC 31C0               <1>       xor   ax, ax
444                             <1>       ;dec  ax     ; 02/01/2015
445 000001DE AB                 <1>       stosw        ; precompensation (obsolete)
446                             <1>       ;xor  al, al ; 02/01/2015
447 000001DF AA                 <1>       stosb        ; reserved
448 000001E0 B008               <1>       mov   al, 8  ; drive control byte
449                             <1>                    ; (do not disable retries,
450                             <1>                    ; more than 8 heads)
451 000001E2 AA                 <1>       stosb
452 000001E3 8B4404             <1>       mov   ax, [si+4]
453 000001E6 AB                 <1>       stosw        ; physical number of cylinders
454                             <1>       ;push ax     ; 02/01/2015
455 000001E7 8A4408             <1>       mov   al, [si+8]
456 000001EA AA                 <1>       stosb        ; physical num. of heads (limit 16)
457 000001EB 29C0               <1>       sub   ax, ax
458                             <1>       ;pop  ax     ; 02/01/2015
459 000001ED AB                 <1>       stosw        ; landing zone (obsolete)
460 000001EE 88C8               <1>       mov   al, cl ; logical sectors per track (limit 63)
461 000001F0 243F               <1>       and   al, 3Fh
462 000001F2 AA                 <1>       stosb
463                             <1>       ;sub  al, al ; checksum
464                             <1>       ;stosb
465                             <1>       ;
466 000001F3 83C61A             <1>       add   si, 26   ; (BIOS) DPTE address pointer
467 000001F6 AD                 <1>       lodsw
468 000001F7 50                 <1>       push  ax     ; (BIOS) DPTE offset
469 000001F8 AD                 <1>       lodsw
470 000001F9 50                 <1>       push  ax     ; (BIOS) DPTE segment
471                             <1>       ;
472                             <1>       ; checksum calculation
473 000001FA 89FE               <1>       mov   si, di
474 000001FC 06                 <1>       push  es
475 000001FD 1F                 <1>       pop   ds
476                             <1>       ;mov  cx, 16
477 000001FE B90F00             <1>       mov   cx, 15
478 00000201 29CE               <1>       sub   si, cx
479 00000203 30E4               <1>       xor   ah, ah
480                             <1>       ;del  cl
481                             <1> L11:
482 00000205 AC                 <1>       lodsb
483 00000206 00C4               <1>       add   ah, al
484 00000208 E2FB               <1>       loop  L11
485                             <1>       ;
486 0000020A 88E0               <1>       mov   al, ah
487 0000020C F6D8               <1>       neg   al     ; -x+x = 0
488 0000020E AA                 <1>       stosb        ; put checksum in byte 15 of the tbl
489                             <1>       ;
490 0000020F 1F                 <1>       pop   ds     ; (BIOS) DPTE segment
491 00000210 5E                 <1>       pop   si     ; (BIOS) DPTE offset
492                             <1>       ;
493                             <1>       ; 23/02/2015
494 00000211 57                 <1>       push  di
495                             <1>       ; ES:DI points to DPTE (FDPTE) location
496                             <1>       ;mov  cx, 8
497 00000212 B108               <1>       mov   cl, 8
498 00000214 F3A5               <1>       rep   movsw
499                             <1>       ;
500                             <1>       ; 23/02/2015
501                             <1>       ; (P)ATA drive and LBA validation
502                             <1>       ; (invalidating SATA drives and setting
503                             <1>       ; CHS type I/O for old type fixed disks)
504 00000216 5B                 <1>       pop   bx
505 00000217 8CC8               <1>       mov   ax, cs
506 00000219 8ED8               <1>       mov   ds, ax
507 0000021B 268B07             <1>       mov   ax, [es:bx]
508 0000021E 3DF001             <1>       cmp   ax, 1F0h
509 00000221 7418               <1>       je    short L11a
```

```
00000223 3D7001              <1>       cmp    ax, 170h
00000226 7413                <1>       je     short L11a
                             <1>       ; invalidation
                             <1>       ; (because base port address is not 1F0h or 170h)
00000228 30FF                <1>       xor    bh, bh
0000022A 88D3                <1>       mov    bl, dl
0000022C 80EB80              <1>       sub    bl, 80h
0000022F C687[F85C]00        <1>       mov    byte [bx+hd0_type], 0 ; not a valid disk drive !
00000234 808F[445D]F0        <1>         or      byte [bx+drv.status+2], 0F0h ; (failure sign)
00000239 EB14                <1>       jmp    short L11b
                             <1> L11a:
                             <1>       ; LBA validation
0000023B 268A4704            <1>       mov    al, [es:bx+4] ; Head register upper nibble
0000023F A840                <1>       test   al, 40h ; LBA bit (bit 6)
00000241 750C                <1>       jnz    short L11b ; LBA type I/O is OK! (E0h or F0h)
                             <1>       ; force CHS type I/O for this drive (A0h or B0h)
00000243 28FF                <1>       sub    bh, bh
00000245 88D3                <1>       mov    bl, dl
00000247 80EB80              <1>       sub    bl, 80h ; 26/02/2015
0000024A 80A7[445D]FE        <1>         and     byte [bx+drv.status+2], 0FEh ; clear bit 0
                             <1>                             ; bit 0 = LBA ready bit
                             <1>       ; 'diskio' procedure will check this bit !
                             <1> L11b:
0000024F 3A16[F45C]          <1>       cmp    dl, [last_drv] ; 25/12/2014
00000253 7307                <1>         jnb     short L13
00000255 E9FCFE              <1>         jmp     L10
                             <1> L12:
                             <1>       ; Restore data registers
00000258 8CC8                <1>       mov    ax, cs
0000025A 8ED8                <1>       mov    ds, ax
                             <1> L13:
                             <1>       ; 13/12/2014
0000025C 0E                  <1>       push   cs
0000025D 07                  <1>       pop    es
                             <1> L14:
0000025E B411                <1>       mov    ah, 11h
00000260 CD16                <1>       int    16h
00000262 7466                <1>       jz     short L16 ; no keys in keyboard buffer
00000264 B010                <1>       mov    al, 10h
00000266 CD16                <1>       int    16h
00000268 EBF4                <1>       jmp    short L14
                             <1>
                             <1> set_disk_parms:
                             <1>       ; 04/02/2016 (ebx -> bx)
                             <1>       ; 10/07/2015
                             <1>       ; 14/01/2015
                             <1>       ;push  bx
0000026A 28FF                <1>       sub    bh, bh
0000026C 8A1E[F35C]          <1>       mov    bl, [drv]
00000270 80FB80              <1>       cmp    bl, 80h
00000273 7203                <1>       jb     short sdp0
00000275 80EB7E              <1>       sub    bl, 7Eh
                             <1> sdp0:
00000278 81C3[425D]          <1>       add    bx, drv.status
0000027C C60780              <1>       mov    byte [bx], 80h ; 'Present' flag
                             <1>       ;
0000027F 88E8                <1>       mov    al, ch ; last cylinder (bits 0-7)
00000281 88CC                <1>       mov    ah, cl ;
00000283 C0EC06              <1>       shr    ah, 6  ; last cylinder (bits 8-9)
00000286 81EB[425D]          <1>       sub    bx, drv.status
0000028A D0E3                <1>       shl    bl, 1
0000028C 81C3[FC5C]          <1>       add    bx, drv.cylinders
00000290 40                  <1>       inc    ax  ; convert max. cyl number to cyl count
00000291 8907                <1>       mov    [bx], ax
00000293 50                  <1>       push   ax ; ** cylinders
00000294 81EB[FC5C]          <1>       sub    bx, drv.cylinders
00000298 81C3[0A5D]          <1>       add    bx, drv.heads
0000029C 30E4                <1>       xor    ah, ah
0000029E 88F0                <1>       mov    al, dh ; heads
000002A0 40                  <1>       inc    ax
000002A1 8907                <1>       mov    [bx], ax
000002A3 81EB[0A5D]          <1>         sub     bx, drv.heads
000002A7 81C3[185D]          <1>         add     bx, drv.spt
000002AB 30ED                <1>       xor    ch, ch
000002AD 80E13F              <1>       and    cl, 3Fh      ; sectors (bits 0-6)
000002B0 890F                <1>       mov    [bx], cx
000002B2 81EB[185D]          <1>         sub     bx, drv.spt
000002B6 D1E3                <1>       shl    bx, 1
000002B8 81C3[265D]          <1>       add    bx, drv.size ; disk size (in sectors)
                             <1>       ; LBA size = cylinders * heads * secpertrack
000002BC F7E1                <1>       mul    cx
000002BE 89C2                <1>       mov    dx, ax ; heads*spt
000002C0 58                  <1>       pop    ax ; ** cylinders
000002C1 48                  <1>       dec    ax ; 1 cylinder reserved (!?)
000002C2 F7E2                <1>       mul    dx ; cylinders * (heads*spt)
000002C4 8907                <1>       mov    [bx], ax
000002C6 895702              <1>       mov    [bx+2], dx
                             <1>       ;
                             <1>       ;pop   bx
000002C9 C3                  <1>       retn
                             <1>
                             <1> L16:  ; 28/05/2016
                                       ; 10/11/2014
000002CA 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
```

```
612 000002CB B080                       mov    al, 80h
613 000002CD E670                       out    70h, al            ; set bit 7 to 1 for disabling NMI
614                                     ;23/02/2015
615                                     ;nop               ;
616                                     ;in    al, 71h             ; read in 71h just after writing out to 70h
617                                                        ; for preventing unknown state (!?)
618                                     ;
619                                     ; 20/08/2014
620                                     ; Moving the kernel 64 KB back (to physical address 0)
621                                     ; DS = CS = 1000h
622                                     ; 05/11/2014
623 000002CF 31C0                       xor    ax, ax
624 000002D1 8EC0                       mov    es, ax ; ES = 0
625                                     ;
626                                     ; 04/07/2016 -  TRDOS 386 (64K - 128K kernel)
627 000002D3 31F6                             xor    si, si
628 000002D5 31FF                       xor    di, di
629 000002D7 B90040                     mov    cx, 16384
630 000002DA F366A5                     rep    movsd
631                                     ;
632 000002DD 06                         push   es ; 0
633 000002DE 68[E202]                   push   L17
634 000002E1 CB                         retf
635                                 L17:
636 000002E2 B90010                     mov    cx, 1000h
637 000002E5 8EC1                       mov    es, cx  ; 1000h
638 000002E7 01C9                       add    cx, cx
639 000002E9 8ED9                       mov    ds, cx  ; 2000h
640 000002EB 29F6                       sub    si, si
641 000002ED 29FF                       sub    di, di
642 000002EF B90040                     mov    cx, 16384
643 000002F2 F366A5                     rep    movsd
644
645                                     ; Turn off the floppy drive motor
646 000002F5 BAF203                       mov     dx, 3F2h
647 000002F8 EE                           out     dx, al ; 0 ; 31/12/2013
648
649                                     ; Enable access to memory above one megabyte
650                                 L18:
651 000002F9 E464                       in     al, 64h
652 000002FB A802                       test   al, 2
653 000002FD 75FA                         jnz     short L18
654 000002FF B0D1                       mov    al, 0D1h     ; Write output port
655 00000301 E664                       out    64h, al
656                                 L19:
657 00000303 E464                       in     al, 64h
658 00000305 A802                       test   al, 2
659 00000307 75FA                         jnz     short L19
660 00000309 B0DF                       mov    al, 0DFh     ; Enable A20 line
661 0000030B E660                       out    60h, al
662                                 ;L20:
663                                     ;
664                                     ; Load global descriptor table register
665
666                                       ;mov     ax, cs
667                                       ;mov     ds, ax
668
669 0000030D 2E0F0116[605C]               lgdt    [cs:gdtd]
670
671 00000313 0F20C0                       mov     eax, cr0
672                                     ; or   eax, 1
673 00000316 40                         inc     ax
674 00000317 0F22C0                     mov     cr0, eax
675
676                                     ; Jump to 32 bit code
677
678 0000031A 66                         db 66h                  ; Prefix for 32-bit
679 0000031B EA                         db 0EAh            ; Opcode for far jump
680 0000031C [22030000]                 dd StartPM         ; Offset to start, 32-bit
681                                                        ; (1000h:StartPM = StartPM + 10000h)
682 00000320 0800                       dw KCODE           ; This is the selector for CODE32_DESCRIPTOR,
683                                                        ; assuming that StartPM resides in code32
684
685                                 ; 20/02/2017
686
687
688                                 [BITS 32]
689
690                                 StartPM:
691                                       ; Kernel Base Address = 0 ; 30/12/2013
692 00000322 66B81000                     mov ax, KDATA           ; Save data segment identifier
693 00000326 8ED8                           mov ds, ax              ; Move a valid data segment into DS register
694 00000328 8EC0                               mov es, ax              ; Move data segment into ES register
695 0000032A 8EE0                               mov fs, ax              ; Move data segment into FS register
696 0000032C 8EE8                               mov gs, ax              ; Move data segment into GS register
697 0000032E 8ED0                           mov ss, ax              ; Move data segment into SS register
698 00000330 BC00000900                     mov esp, 90000h         ; Move the stack pointer to 090000h
699
700                                 clear_bss: ; Clear uninitialized data area
701                                       ; 11/03/2015
702 00000335 31C0                       xor  eax, eax ; 0
703 00000337 B9B0310100                 mov  ecx, (bss_end - bss_start)/4
704                                     ;shr  ecx, 2 ; bss section is already aligned for double words
705 0000033C BF[DA4A0100]               mov  edi, bss_start
706 00000341 F3AB                       rep  stosd
707
708                                 memory_init:
709                                       ; Initialize memory allocation table and page tables
710                                       ; 16/11/2014
711                                       ; 15/11/2014
712                                       ; 07/11/2014
713                                       ; 06/11/2014
```

```
                                        ; 05/11/2014
                                        ; 04/11/2014
                                        ; 31/10/2014 (Retro UNIX 386 v1 - Beginning)
                                        ;
                                ;       xor     eax, eax
                                ;       xor     ecx, ecx
00000343 B108                           mov     cl, 8
00000345 BF00001000                     mov     edi, MEM_ALLOC_TBL
0000034A F3AB                           rep     stosd           ; clear Memory Allocation Table
                                                                ; for the first 1 MB memory
                                        ;
0000034C 668B0D[EE5C0000]               mov     cx, [mem_1m_1k]         ; Number of contiguous KB between
                                                                ; 1 and 16 MB, max. 3C00h = 15 MB.
00000353 66C1E902                       shr     cx, 2           ; convert 1 KB count to 4 KB count
00000357 890D[D04D0100]                 mov     [free_pages], ecx
0000035D 668B15[F05C0000]               mov     dx, [mem_16m_64k]  ; Number of contiguous 64 KB blocks
                                                                ; between 16 MB and 4 GB.
00000364 6609D2                         or      dx, dx
00000367 7413                           jz      short mi_0
                                        ;
00000369 6689D0                         mov     ax, dx
0000036C C1E004                         shl     eax, 4          ; 64 KB -> 4 KB (page count)
0000036F 0105[D04D0100]                 add     [free_pages], eax
00000375 0500100000                     add     eax, 4096       ; 16 MB = 4096 pages
0000037A EB07                           jmp     short mi_1
                                mi_0:
0000037C 6689C8                         mov     ax, cx
0000037F 66050001                       add     ax, 256                 ; add 256 pages for the first 1 MB
                                mi_1:
00000383 A3[CC4D0100]                   mov     [memory_size], eax ; Total available memory in pages
                                                                ; 1 alloc. tbl. bit = 1 memory page
                                                                ; 32 allocation bits = 32 mem. pages
                                        ;
00000388 05FF7F0000                     add     eax, 32767      ; 32768 memory pages per 1 M.A.T. page
0000038D 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
00000390 66A3[E04D0100]                 mov     [mat_size], ax          ; Memory Alloc. Table Size in pages
00000396 C1E00C                         shl     eax, 12                 ; 1 M.A.T. page = 4096 bytes
                                        ;                       ; Max. 32 M.A.T. pages (4 GB memory)
00000399 89C3                           mov     ebx, eax        ; M.A.T. size in bytes
                                        ; Set/Calculate Kernel's Page Directory Address
0000039B 81C300001000                   add     ebx, MEM_ALLOC_TBL
000003A1 891D[C84D0100]                 mov     [k_page_dir], ebx  ; Kernel's Page Directory address
                                                                ; just after the last M.A.T. page
                                        ;
000003A7 83E804                         sub     eax, 4          ; convert M.A.T. size to offset value
000003AA A3[D84D0100]                   mov     [last_page], eax   ; last page ofset in the M.A.T.
                                        ;                       ; (allocation status search must be
                                                                ; stopped after here)
000003AF 31C0                           xor     eax, eax
000003B1 48                             dec     eax             ; FFFFFFFFh (set all bits to 1)
000003B2 6651                           push    cx
000003B4 C1E905                         shr     ecx, 5          ; convert 1 - 16 MB page count to
                                                                ; count of 32 allocation bits
000003B7 F3AB                           rep     stosd
000003B9 6659                           pop     cx
000003BB 40                             inc     eax             ; 0
000003BC 80E11F                         and     cl, 31          ; remain bits
000003BF 7412                           jz      short mi_4
000003C1 8907                           mov     [edi], eax      ; reset
                                mi_2:
000003C3 0FAB07                         bts     [edi], eax      ; 06/11/2014
000003C6 FEC9                           dec     cl
000003C8 7404                           jz      short mi_3
000003CA FEC0                           inc     al
000003CC EBF5                           jmp     short mi_2
                                mi_3:
000003CE 28C0                           sub     al, al          ; 0
000003D0 83C704                         add     edi, 4          ; 15/11/2014
                                mi_4:
000003D3 6609D2                         or      dx, dx          ; check 16M to 4G memory space
000003D6 7421                           jz      short mi_6      ; max. 16 MB memory, no more...
                                        ;
000003D8 B900021000                     mov     ecx, MEM_ALLOC_TBL + 512 ; End of first 16 MB memory
                                        ;
000003DD 29F9                           sub     ecx, edi        ; displacement (to end of 16 MB)
000003DF 7406                           jz      short mi_5      ; jump if EDI points to
                                                                ;         end of first 16 MB
000003E1 D1E9                           shr     ecx, 1          ; convert to dword count
000003E3 D1E9                           shr     ecx, 1          ; (shift 2 bits right)
000003E5 F3AB                           rep     stosd           ; reset all bits for reserved pages
                                                                ; (memory hole under 16 MB)
                                mi_5:
000003E7 6689D1                         mov     cx, dx          ; count of 64 KB memory blocks
000003EA D1E9                           shr     ecx, 1          ; 1 alloc. dword per 128 KB memory
000003EC 9C                             pushf                   ; 16/11/2014
000003ED 48                             dec     eax             ; FFFFFFFFh (set all bits to 1)
000003EE F3AB                           rep     stosd
000003F0 40                             inc     eax             ; 0
000003F1 9D                             popf                    ; 16/11/2014
000003F2 7305                           jnc     short mi_6
000003F4 6648                           dec     ax              ; eax = 0000FFFFh
000003F6 AB                             stosd
000003F7 6640                           inc     ax              ; 0
                                mi_6:
000003F9 39DF                           cmp     edi, ebx        ; check if EDI points to
000003FB 730A                           jnb     short mi_7      ; end of memory allocation table
                                        ;                       ; (>= MEM_ALLOC_TBL + 4906)
000003FD 89D9                           mov     ecx, ebx        ; end of memory allocation table
000003FF 29F9                           sub     ecx, edi        ; convert displacement/offset
00000401 D1E9                           shr     ecx, 1          ; to dword count
```

```
00000403 D1E9                   shr    ecx, 1          ; (shift 2 bits right)
00000405 F3AB                   rep    stosd           ; reset all remain M.A.T. bits
                        mi_7:
                                ; Reset M.A.T. bits in M.A.T. (allocate M.A.T. pages)
00000407 BA00001000             mov    edx, MEM_ALLOC_TBL
                                ;sub   ebx, edx        ; Mem. Alloc. Tbl. size in bytes
                                ;shr   ebx, 12               ; Mem. Alloc. Tbl. size in pages
0000040C 668B0D[E04D0100]       mov    cx, [mat_size]        ; Mem. Alloc. Tbl. size in pages
00000413 89D7                   mov    edi, edx
00000415 C1EF0F                 shr    edi, 15               ; convert M.A.T. address to
                                                             ; byte offset in M.A.T.
                                                             ; (1 M.A.T. byte points to
                                                             ;         32768 bytes)
                                                             ; Note: MEM_ALLOC_TBL address
                                                             ; must be aligned on 128 KB
                                                             ; boundary!
00000418 01D7                   add    edi, edx        ; points to M.A.T.'s itself
                                ; eax = 0
0000041A 290D[D04D0100]         sub    [free_pages], ecx ; 07/11/2014
                        mi_8:
00000420 0FB307                 btr    [edi], eax      ; clear bit 0 to bit x (1 to 31)
                                ;dec   bl
00000423 FEC9                   dec    cl
00000425 7404                   jz     short mi_9
00000427 FEC0                   inc    al
00000429 EBF5                   jmp    short mi_8
                        mi_9:
                                ;
                                ; Reset Kernel's Page Dir. and Page Table bits in M.A.T.
                                ;            (allocate pages for system page tables)

                                ; edx = MEM_ALLOC_TBL
0000042B 8B0D[CC4D0100]         mov    ecx, [memory_size] ; memory size in pages (PTEs)
00000431 81C1FF030000           add    ecx, 1023      ; round up (1024 PTEs per table)
00000437 C1E90A                 shr    ecx, 10              ; convert memory page count to
                                                             ; page table count (PDE count)
                                ;
0000043A 51                     push   ecx            ; (**) PDE count (<= 1024)
                                ;
0000043B 41                     inc    ecx            ; +1 for kernel page directory
                                ;
0000043C 290D[D04D0100]         sub    [free_pages], ecx ; 07/11/2014
                                ;
00000442 8B35[C84D0100]         mov    esi, [k_page_dir] ; Kernel's Page Directory address
00000448 C1EE0C                 shr    esi, 12              ; convert to page number
                        mi_10:
0000044B 89F0                   mov    eax, esi       ; allocation bit offset
0000044D 89C3                   mov    ebx, eax
0000044F C1EB03                 shr    ebx, 3         ; convert to alloc. byte offset
00000452 80E3FC                 and    bl,  0FCh      ; clear bit 0 and bit 1
                                                      ;   to align on dword boundary
00000455 83E01F                 and    eax, 31              ; set allocation bit position
                                                      ;  (bit 0 to bit 31)
                                ;
00000458 01D3                   add    ebx, edx       ; offset in M.A.T. + M.A.T. address
                                ;
0000045A 0FB303                 btr    [ebx], eax     ; reset relevant bit (0 to 31)
                                ;
0000045D 46                     inc    esi            ; next page table
0000045E E2EB                   loop   mi_10          ; allocate next kernel page table
                                                      ; (ecx = page table count + 1)
                                ;
00000460 59                     pop    ecx            ; (**) PDE count (= pg. tbl. count)
                                ;
                                ; Initialize Kernel Page Directory and Kernel Page Tables
                                ;
                                ; Initialize Kernel's Page Directory
00000461 8B3D[C84D0100]         mov    edi, [k_page_dir]
00000467 89F8                   mov    eax, edi
00000469 0C03                   or     al, PDE_A_PRESENT + PDE_A_WRITE
                                                             ; supervisor + read&write + present
0000046B 89CA                   mov    edx, ecx       ; (**) PDE count (= pg. tbl. count)
                        mi_11:
0000046D 0500100000             add    eax, 4096      ; Add page size (PGSZ)
                                                       ; EAX points to next page table
00000472 AB                     stosd
00000473 E2F8                   loop   mi_11
00000475 29C0                   sub    eax, eax       ; Empty PDE
00000477 66B90004               mov    cx, 1024       ; Entry count (PGSZ/4)
0000047B 29D1                   sub    ecx, edx
0000047D 7402                   jz     short mi_12
0000047F F3AB                   rep    stosd          ; clear remain (empty) PDEs
                                ;
                                ; Initialization of Kernel's Page Directory is OK, here.
                        mi_12:
                                ; Initialize Kernel's Page Tables
                                ;
                                ; (EDI points to address of page table 0)
                                ; eax = 0
00000481 8B0D[CC4D0100]         mov    ecx, [memory_size] ; memory size in pages
00000487 89CA                   mov    edx, ecx       ; (***)
00000489 B003                   mov    al, PTE_A_PRESENT + PTE_A_WRITE
                                                             ; supervisor + read&write + present
                        mi_13:
0000048B AB                     stosd
0000048C 0500100000             add    eax, 4096
00000491 E2F8                   loop   mi_13
00000493 6681E2FF03             and    dx, 1023       ; (***)
00000498 740B                   jz     short mi_14
0000049A 66B90004               mov    cx, 1024
0000049E 6629D1                 sub    cx, dx         ; from dx (<= 1023) to 1024
000004A1 31C0                   xor    eax, eax
```

```
000004A3 F3AB                   rep    stosd        ; clear remain (empty) PTEs
                                                    ; of the last page table
                        mi_14:
                                ;  Initialization of Kernel's Page Tables is OK, here.
                                ;
000004A5 89F8                   mov    eax, edi     ; end of the last page table page
                                                     ; (beginging of user space pages)
000004A7 C1E80F                 shr    eax, 15           ; convert to M.A.T. byte offset
000004AA 24FC                   and    al, 0FCh     ; clear bit 0 and bit 1 for
                                                    ; aligning on dword boundary
                                
000004AC A3[DC4D0100]           mov    [first_page], eax
000004B1 A3[D44D0100]           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
                                ;
000004B6 A1[C84D0100]             mov     eax, [k_page_dir]
000004BB 0F22D8                 mov    cr3, eax
000004BE 0F20C0                 mov    eax, cr0
000004C1 0D00000080             or     eax, 80000000h      ; set paging bit (bit 31)
000004C6 0F22C0                 mov    cr0, eax
                                  ;jmp    KCODE:StartPMP

000004C9 EA                     db 0EAh             ; Opcode for far jump
000004CA [D0040000]               dd StartPMP             ; 32 bit offset
000004CE 0800                   dw KCODE            ; kernel code segment descriptor

                        StartPMP:
                                ; 06/11//2014
                                ; Clear video page 0
                                ;
                                ; Temporary Code
                                ;
000004D0 B9E8030000             mov    ecx, 80*25/2
000004D5 BF00800B00             mov    edi, 0B8000h
                                ; 30/01/2016
                                ;xor   eax, eax     ; black background, black fore color
000004DA B800070007             mov    eax, 07000700h  ; black background, light gray fore color
000004DF 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)
000004E1 BE[210F0100]           mov    esi, starting_msg
                                ;; 14/08/2015 (kernel version message will appear
                                ;;           when protected mode and paging is enabled)
000004E6 BF00800B00             mov    edi, 0B8000h ; 27/08/2014
000004EB B40A                   mov    ah, 0Ah ; Black background, light green forecolor
                                ; 20/08/2014
000004ED E88F010000             call   printk

                                ; 'UNIX v7/x86' source code by Robert Nordier (1999)
                                ; // Set IRQ offsets
                                ;
                                ;  Linux (v0.12) source code by Linus Torvalds (1991)
                                ;
                                                        ;; ICW1
000004F2 B011                   mov    al, 11h                  ; Initialization sequence
000004F4 E620                   out    20h, al                  ;     8259A-1
                                ; jmp  $+2
000004F6 E6A0                   out    0A0h, al         ;     8259A-2
                                                        ;; ICW2
000004F8 B020                   mov    al, 20h                  ; Start of hardware ints (20h)
000004FA E621                   out    21h, al                  ;     for 8259A-1
                                ; jmp  $+2
000004FC B028                   mov    al, 28h                  ; Start of hardware ints (28h)
000004FE E6A1                   out    0A1h, al         ;     for 8259A-2
                                                        ;
00000500 B004                   mov    al, 04h                  ;; ICW3
00000502 E621                   out    21h, al                  ;     IRQ2 of 8259A-1 (master)
                                ; jmp  $+2
00000504 B002                   mov    al, 02h                  ;     is 8259A-2 (slave)
00000506 E6A1                   out    0A1h, al         ;
                                                        ;; ICW4
00000508 B001                   mov    al, 01h                  ;
0000050A E621                   out    21h, al                  ;     8086 mode, normal EOI
                                ; jmp  $+2
0000050C 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)
0000050E BE[E40B0100]                   mov    esi, ilist
00000513 8D3D[E04A0100]                 lea    edi, [idt]
                                        ; 26/03/2015
00000519 B930000000                     mov    ecx, 48             ; 48 hardware interrupts (INT 0 to INT 2Fh)
                                        ; 02/04/2015
0000051E BB00000800                     mov    ebx,  80000h
                                rp_sidt1:
00000523 AD                             lodsd
00000524 89C2                           mov    edx, eax
00000526 66BA008E                       mov    dx, 8E00h
0000052A 6689C3                         mov    bx, ax
0000052D 89D8                           mov    eax, ebx     ; /* selector = 0x0008 = cs */
                                                             ; /* interrupt gate - dpl=0, present */
0000052F AB                             stosd  ; selector & offset bits 0-15
00000530 89D0                           mov    eax, edx
00000532 AB                             stosd  ; attributes & offset bits 16-23
00000533 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)
00000535 B120                           mov    cl, 32       ; 32 software interrupts (INT 30h to INT 4Fh)
                                rp_sidt2:
00000537 AD                             lodsd
00000538 21C0                           and    eax, eax
0000053A 7413                           jz     short rp_sidt3
0000053C 89C2                           mov    edx, eax
0000053E 66BA00EE                       mov    dx, 0EE00h   ; P=1b/DPL=11b/01110b
00000542 6689C3                         mov    bx, ax
00000545 89D8                           mov    eax, ebx     ; selector & offset bits 0-15
00000547 AB                             stosd
00000548 89D0                           mov    eax, edx
0000054A AB                             stosd
0000054B E2EA                           loop   rp_sidt2
0000054D EB16                           jmp    short sidt_OK
                                rp_sidt3:
0000054F B8[AA0A0000]                   mov    eax, ignore_int
00000554 89C2                           mov    edx, eax
00000556 66BA00EE                       mov    dx, 0EE00h   ; P=1b/DPL=11b/01110b
0000055A 6689C3                         mov    bx, ax
0000055D 89D8                           mov    eax, ebx     ; selector & offset bits 0-15
                                rp_sidt4:
0000055F AB                             stosd
00000560 92                             xchg   eax, edx
00000561 AB                             stosd
00000562 92                             xchg   edx, eax
00000563 E2FA                           loop   rp_sidt4
                                sidt_OK:
00000565 0F011D[665C0000]               lidt   [idtd]
                                        ;
                                        ; TSS descriptor setup ; 24/03/2015
0000056C B8[604D0100]                   mov    eax, task_state_segment
00000571 66A3[5A5C0000]                 mov    [gdt_tss0], ax
00000577 C1C010                         rol    eax, 16
0000057A A2[5C5C0000]                   mov    [gdt_tss1], al
0000057F 8825[5F5C0000]                 mov    [gdt_tss2], ah
00000585 66C705[C64D0100]68-            mov    word [tss.IOPB], tss_end - task_state_segment
0000058D 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)
0000058E 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.)
00000592 0F00D8                         ltr    ax  ; Load task register
                                        ;
                                esp0_set0:
                                        ; 30/07/2015
00000595 8B0D[CC4D0100]                 mov    ecx, [memory_size] ; memory size in pages
0000059B C1E10C                         shl    ecx, 12 ; convert page count to byte count
0000059E 81F900004000                   cmp    ecx, CORE ; beginning of user's memory space (400000h)
                                                         ; (kernel mode virtual address)
000005A4 7605                           jna    short esp0_set1
                                        ;
                                        ; If available memory > CORE (end of the 1st 4 MB)
                                        ; set stack pointer to CORE
                                        ;(Because, PDE 0 is reserved for kernel space in user's page directory)
                                        ;(PDE 0 points to page table of the 1st 4 MB virtual address space)
000005A6 B900004000                     mov    ecx, CORE
                                esp0_set1:
000005AB 89CC                           mov    esp, ecx ; top of kernel stack (**tss.esp0**)
                                esp0_set_ok:
                                        ; 30/07/2015 (**tss.esp0**)
000005AD 8925[644D0100]                 mov    [tss.esp0], esp
000005B3 66C705[684D0100]10-              mov     word [tss.ss0], KDATA
000005BB 00
                                        ; 14/08/2015
                                        ; 10/11/2014 (Retro UNIX 386 v1 - Erdogan Tan)
                                        ;
                                        ;cli   ; Disable interrupts (for CPU)
                                        ;    (CPU will not handle hardware interrupts, except NMI!)
                                        ;
000005BC 30C0                           xor    al, al       ; Enable all hardware interrupts!
000005BE E621                           out    21h, al             ; (IBM PC-AT compatibility)
000005C0 EB00                           jmp    $+2          ; (All conventional PC-AT hardware
```

```
1122 000005C2 E6A1                      out    0A1h, al     ;  interrupts will be in use.)
1123                                                        ; (Even if related hardware component
1124                                                        ;  does not exist!)
1125                                    ; Enable NMI
1126 000005C4 B07F                      mov    al, 7Fh             ; Clear bit 7 to enable NMI (again)
1127 000005C6 E670                      out    70h, al
1128                                    ; 23/02/2015
1129 000005C8 90                        nop
1130 000005C9 E471                      in     al, 71h             ; read in 71h just after writing out to 70h
1131                                                        ; for preventing unknown state (!?)
1132                                    ;
1133                                    ; Only a NMI can occur here... (Before a 'STI' instruction)
1134                                    ;
1135                                    ; 02/09/2014
1136 000005CB 6631DB                    xor    bx, bx
1137 000005CE 66BA0002                  mov    dx, 0200h    ; Row 2, column 0  ; 07/03/2015
1138 000005D2 E871170000                call   _set_cpos    ; 24/01/2016
1139                                    ;
1140                                    ; 06/11/2014
1141 000005D7 E8782C0000                call   memory_info
1142                                    ; 14/08/2015
1143                                    ;call getch ; 28/02/2015
1144                                drv_init:
1145 000005DC FB                        sti    ; Enable Interrupts
1146                                    ; 06/02/2015
1147 000005DD 8B15[F85C0000]            mov    edx, [hd0_type] ; hd0, hd1, hd2, hd3
1148 000005E3 668B1D[F65C0000]          mov    bx, [fd0_type] ; fd0, fd1
1149                                    ; 22/02/2015
1150 000005EA 6621DB                    and    bx, bx
1151 000005ED 751C                      jnz    short di1
1152                                    ;
1153 000005EF 09D2                      or     edx, edx
1154 000005F1 752A                      jnz    short di2
1155                                    ;
1156                                setup_error:
1157 000005F3 BE[EA0E0100]              mov    esi, setup_error_msg
1158                                psem:
1159 000005F8 AC                        lodsb
1160 000005F9 08C0                      or     al, al
1161                                    ;jz    short haltx ; 22/02/2015
1162 000005FB 7427                      jz     short di3
1163 000005FD 56                        push   esi
1164                                    ; 13/05/2016
1165 000005FE BB07000000                mov    ebx, 7 ; Black background,
1166                                                   ; light gray forecolor
1167                                                   ; Video page 0 (BH=0)
1168 00000603 E8AA160000                call   _write_tty
1169 00000608 5E                        pop    esi
1170 00000609 EBED                      jmp    short psem
1171
1172                                di1:
1173                                    ; supress 'jmp short T6'
1174                                    ;  (activate fdc motor control code)
1175 0000060B 66C705[EB060000]90-       mov    word [T5], 9090h ; nop
1176 00000613 90
1177                                    ;
1178                                    ;mov   ax, int_0Eh  ; IRQ 6 handler
1179                                    ;mov   di, 0Eh*4    ; IRQ 6 vector
1180                                    ;stosw
1181                                    ;mov   ax, cs
1182                                    ;stosw
1183                                    ;; 16/02/2015
1184                                      ;;mov     dword [DISKETTE_INT], fdc_int ; IRQ 6 handler
1185                                    ;
1186 00000614 E8AF3B0000                CALL   DSKETTE_SETUP; Initialize Floppy Disks
1187                                    ;
1188 00000619 09D2                      or     edx, edx
1189 0000061B 7407                        jz      short di3
1190                                di2:
1191 0000061D E8EC3B0000                call         DISK_SETUP   ; Initialize Fixed Disks
1192 00000622 72CF                        jc      short setup_error
1193                                di3:
1194 00000624 E8FF2B0000                call   setup_rtc_int; 22/05/2015 (dsectrpm.s)
1195                                    ;
1196 00000629 E852070100                call   display_disks ; 07/03/2015  (Temporary)
1197                                ;haltx:
1198                                    ; 14/08/2015
1199                                    ;call  getch ; 22/02/2015
1200                                    ;sti   ; Enable interrupts (for CPU)
1201                                ;     ; 29/01/2016
1202                                ;     sub    ah, ah ;  read time count
1203                                ;     call   int1Ah
1204                                ;     mov    edx, ecx ; 18.2 * seconds
1205                                ;md_info_msg_wait1:
1206                                ;     ; 29/01/2016
1207                                ;     mov    ah, 1
1208                                ;     call   int16h
1209                                ;     jz     short md_info_msg_wait2
1210                                ;     xor    ah, ah ; 0
1211                                ;       call    int16h
1212                                ;     jmp    short md_info_msg_ok
1213                                ;md_info_msg_wait2:
1214                                ;     sub    ah, ah  ; read time count
1215                                ;     call   int1Ah
1216                                ;     cmp    edx, ecx ; ; 18.2 * seconds
1217                                ;     jna    short md_info_msg_wait3
1218                                ;     xchg   edx, ecx
1219                                ;md_info_msg_wait3:
1220                                ;     sub    ecx, edx
1221                                ;     cmp    ecx, 127 ; 7 seconds (18.2 * 7)
1222                                ;     jb     short md_info_msg_wait1
1223                                ;md_info_msg_ok:
```

```
1224                                    ; 08/09/2016
1225 0000062E 0F20C0                    mov    eax, cr0
1226 00000631 A810                      test   al, 10h  ; Bit 4, ET (Extension Type)
1227 00000633 7408                      jz     short sysinit
1228                                    ; 27/02/2017
1229 00000635 FE05[885B0100]            inc    byte [fpready]
1230                                    ; 80387 (FPU) is ready
1231 0000063B DBE3                      fninit ; Initialize Floating-Point Unit
1232                                sysinit:
1233                                    ; 30/06/2015
1234 0000063D E80C5C0000                call   sys_init
1235                                    ;
1236                                    ;jmp   cpu_reset ; 22/02/2015
1237                                hang:
1238                                    ; 23/02/2015
1239                                    ;sti               ; Enable interrupts
1240 00000642 F4                        hlt
1241                                    ;
1242                                    ;nop
1243                                    ;; 03/12/2014
1244                                    ;; 28/08/2014
1245                                    ;mov   ah, 11h
1246                                    ;call  getc
1247                                    ;jz      _c8
1248                                    ;
1249                                    ; 23/02/2015
1250                                    ; 06/02/2015
1251                                    ; 07/09/2014
1252 00000643 31DB                      xor    ebx, ebx
1253 00000645 8A1D[F64D0100]            mov    bl, [ptty]   ; active_page
1254 0000064B 89DE                      mov    esi, ebx
1255 0000064D 66D1E6                    shl    si, 1
1256 00000650 81C6[F84D0100]            add    esi, ttychr
1257 00000656 668B06                    mov    ax, [esi]
1258 00000659 6621C0                    and    ax, ax
1259                                    ;jz    short _c8
1260 0000065C 74E4                      jz     short hang
1261 0000065E 66C7060000                mov    word [esi], 0
1262 00000663 80FB03                    cmp    bl, 3        ; Video page 3
1263                                    ;jb    short _c8
1264 00000666 72DA                      jb     short hang
1265                                    ;
1266                                    ; 13/05/2016
1267                                    ; 07/09/2014
1268                                nxtl:
1269 00000668 6653                      push   bx
1270 0000066A 66BB0E00                  mov    bx, 0Eh      ; Yellow character
1271                                                        ; on black background
1272                                                        ; bh = 0 (video page 0)
1273                                                        ; Retro UNIX 386 v1 - Video Mode 0
1274                                                        ; (PC/AT Video Mode 3 - 80x25 Alpha.)
1275 0000066E 6650                      push   ax
1276 00000670 E83D160000                call   _write_tty
1277 00000675 6658                      pop    ax
1278 00000677 665B                      pop    bx
1279 00000679 3C0D                      cmp    al, 0Dh             ; carriage return (enter)
1280                                    ;jne   short _c8
1281 0000067B 75C5                      jne    short hang
1282 0000067D B00A                      mov    al, 0Ah             ; next line
1283 0000067F EBE7                      jmp    short nxtl
1284
1285                                ;_c8:
1286                                ;       ; 25/08/2014
1287                                ;       cli                        ; Disable interrupts
1288                                ;       mov    al, [scounter + 1]
1289                                ;       and    al, al
1290                                ;       jnz    hang
1291                                ;       call   rtc_p
1292                                ;       jmp     hang
1293
1294
1295                                    ; 27/08/2014
1296                                    ; 20/08/2014
1297                                printk:
1298                                      ;mov    edi, [scr_row]
1299                                pkl:
1300 00000681 AC                        lodsb
1301 00000682 08C0                      or     al, al
1302 00000684 7404                      jz     short pkr
1303 00000686 66AB                      stosw
1304 00000688 EBF7                      jmp    short pkl
1305                                pkr:
1306 0000068A C3                        retn
1307
1308                                ; 28/02/2017
1309                                ; 22/01/2017
1310                                ; 15/01/2017
1311                                ; 14/01/2017
1312                                ; 02/01/2017
1313                                ; 25/12/2016
1314                                ; 19/12/2016
1315                                ; 10/12/2016 (callback)
1316                                ; 06/06/2016
1317                                ; 23/05/2016
1318                                ; 22/05/2016 - TRDOS 386 (TRDOS v2.0) Timer Event Modifications
1319                                ; 25/07/2015
1320                                ; 14/05/2015 (multi tasking -time sharing- 'clock', x_timer)
1321                                ; 17/02/2015
1322                                ; 06/02/2015 (unix386.s)
1323                                ; 11/12/2014 - 22/12/2014 (dsectrm2.s)
1324                                ;
1325                                ; IBM PC-XT Model 286 Source Code - BIOS2.ASM (06/10/85)
```

```
1326                                    ;
1327                                    ;-- HARDWARE INT  08 H - ( IRQ LEVEL 0 ) ----------------------------------------
1328                                    ;     THIS ROUTINE HANDLES THE TIMER INTERRUPT FROM FROM CHANNEL 0 OF        :
1329                                    ;     THE 8254 TIMER.  INPUT FREQUENCY IS 1.19318 MHZ AND THE DIVISOR        :
1330                                    ;     IS 65536, RESULTING IN APPROXIMATELY 18.2 INTERRUPTS EVERY SECOND.     :
1331                                    ;                                                            :
1332                                    ;     THE INTERRUPT HANDLER MAINTAINS A COUNT (40:6C) OF INTERRUPTS SINCE    :
1333                                    ;     POWER ON TIME, WHICH MAY BE USED TO ESTABLISH TIME OF DAY.        :
1334                                    ;     THE INTERRUPT HANDLER ALSO DECREMENTS THE MOTOR CONTROL COUNT (40:40)  :
1335                                    ;     OF THE DISKETTE, AND WHEN IT EXPIRES, WILL TURN OFF THE            :
1336                                    ;     DISKETTE MOTOR(s), AND RESET THE MOTOR RUNNING FLAGS.                   :
1337                                    ;     THE INTERRUPT HANDLER WILL ALSO INVOKE A USER ROUTINE THROUGH            :
1338                                    ;     INTERRUPT 1CH AT EVERY TIME TICK.  THE USER MUST CODE A           :
1339                                    ;     ROUTINE AND PLACE THE CORRECT ADDRESS IN THE VECTOR TABLE.         :
1340                                    ;--------------------------------------------------------------------------
1341                                    ;
1342
1343                                    timer_int:   ; IRQ 0
1344                                    ;int_08h:    ; Timer
1345                                          ; 14/10/2015
1346                                          ; Here, we are simulating system call entry (for task switch)
1347                                          ; (If multitasking is enabled,
1348                                          ; 'clock' procedure may jump to 'sysrelease')
1349
1350 0000068B 1E                            push   ds
1351 0000068C 06                            push   es
1352 0000068D 0FA0                          push   fs
1353 0000068F 0FA8                          push   gs
1354
1355 00000691 60                            pushad ; eax, ecx, edx, ebx, esp -before pushad-, ebp, esi, edi
1356 00000692 66B91000                      mov     cx, KDATA
1357 00000696 8ED9                            mov     ds, cx
1358 00000698 8EC1                            mov     es, cx
1359 0000069A 8EE1                            mov     fs, cx
1360 0000069C 8EE9                            mov     gs, cx
1361
1362 0000069E 0F20D9                        mov    ecx, cr3
1363 000006A1 890D[5C040600]                mov    [cr3reg], ecx ; save current cr3 register value/content
1364
1365                                        ; 14/01/2017
1366 000006A7 3B0D[C84D0100]                cmp    ecx, [k_page_dir]
1367 000006AD 7409                          je     short T3
1368
1369 000006AF 8B0D[C84D0100]                mov    ecx, [k_page_dir]
1370 000006B5 0F22D9                        mov    cr3, ecx
1371                                    T3:
1372                                        ;sti                        ; INTERRUPTS BACK ON
1373 000006B8 66FF05[484E0100]              INC    word [TIMER_LOW]     ; INCREMENT TIME
1374 000006BF 7507                          JNZ    short T4             ; GO TO TEST_DAY
1375 000006C1 66FF05[4A4E0100]              INC    word [TIMER_HIGH]    ; INCREMENT HIGH WORD OF TIME
1376                                    T4:                             ; TEST_DAY
1377 000006C8 66833D[4A4E0100]18            CMP    word [TIMER_HIGH],018H    ; TEST FOR COUNT EQUALING 24 HOURS
1378 000006D0 7519                          JNZ    short T5             ; GO TO DISKETTE_CTL
1379 000006D2 66813D[484E0100]B0-           CMP    word [TIMER_LOW],0B0H
1380 000006DA 00
1381 000006DB 750E                          JNZ    short T5             ; GO TO DISKETTE_CTL
1382
1383                                    ;-----      TIMER HAS GONE 24 HOURS
1384                                        ;;SUB  AX,AX
1385                                        ;MOV   [TIMER_HIGH],AX
1386                                        ;MOV   [TIMER_LOW],AX
1387 000006DD 29C0                          sub    eax, eax
1388 000006DF A3[484E0100]                  mov    [TIMER_LH], eax
1389                                        ;
1390 000006E4 C605[4C4E0100]01              MOV    byte [TIMER_OFL],1
1391
1392                                    ;-----      TEST FOR DISKETTE TIME OUT
1393
1394                                    T5:
1395                                        ; 23/12/2014
1396 000006EB EB1D                          jmp    short T6            ; will be replaced with nop, nop
1397                                                                    ; (9090h) if a floppy disk
1398                                                                    ; is detected.
1399                                        ;mov   al,[CS:MOTOR_COUNT]
1400 000006ED A0[4F4E0100]                  mov    al, [MOTOR_COUNT]
1401 000006F2 FEC8                          dec    al
1402                                        ;mov   [CS:MOTOR_COUNT], al      ; DECREMENT DISKETTE MOTOR CONTROL
1403 000006F4 A2[4F4E0100]                  mov    [MOTOR_COUNT], al
1404                                        ;mov   [ORG_MOTOR_COUNT], al
1405 000006F9 750F                          JNZ    short T6            ; RETURN IF COUNT NOT OUT
1406 000006FB B0F0                          mov    al,0F0h
1407                                        ;AND   [CS:MOTOR_STATUS],al      ; TURN OFF MOTOR RUNNING BITS
1408 000006FD 2005[4E4E0100]                and    [MOTOR_STATUS], al
1409                                        ;and   [ORG_MOTOR_STATUS], al
1410 00000703 B00C                          MOV    AL,0CH               ; bit 3 = enable IRQ & DMA,
1411                                                                    ; bit 2 = enable controller
1412                                                                    ;      1 = normal operation
1413                                                                    ;      0 = reset
1414                                                                    ; bit 0, 1 = drive select
1415                                                                    ; bit 4-7 = motor running bits
1416 00000705 66BAF203                      MOV    DX,03F2H             ; FDC CTL PORT
1417 00000709 EE                            OUT    DX,AL                ; TURN OFF THE MOTOR
1418                                    T6:
1419                                        ;inc   word [CS:wait_count]      ; 22/12/2014 (byte -> word)
1420                                                                    ; TIMER TICK INTERRUPT
1421                                        ;;inc  word [wait_count] ;;27/02/2015
1422                                        ;INT   1CH                  ; TRANSFER CONTROL TO A USER ROUTINE
1423                                        ;cli
1424 0000070A E857040000                    call   u_timer                  ; TRANSFER CONTROL TO A USER ROUTINE
1425                                        ; 23/05/2016
1426 0000070F E8E9EB0000                    call   clock                ; Multi Tasking control procedure
1427                                    T7:
```

```
                                        ; 14/10/2015
00000714 B020                           MOV    AL,EOI              ; GET END OF INTERRUPT MASK
00000716 FA                             CLI                        ; DISABLE INTERRUPTS TILL STACK CLEARED
00000717 E620                           OUT    INTA00,AL           ; END OF INTERRUPT TO 8259 - 1
                                        ;
                                rtc_int_2:
                                        ; 26/12/2016
                                        ;mov   ecx, [cr3reg]
                                        ; 13/01/2017
00000719 803D[D4030600]00               cmp    byte [u.t_lock], 0        ; T_LOCK
00000720 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!)
00000722 803D[D8030600]00               cmp    byte [u.r_lock], 0  ; R_LOCK, IRQ callback service lock !
00000729 7727                           ja     short timer_int_return  ; Timer Lock : 'sysrele' is needed !
                                        ; 15/01/2017
0000072B 803D[5C5B0100]02               cmp    byte [priority], 2
00000732 733A                           jnb    short T8  ; current process has a timer event (15/01/2017)
                                        ; 22/05/2016
00000734 803D[5D5B0100]00               cmp    byte [p_change], 0 ; in 'set_run_sequence', in 'rtc_p'
0000073B 7615                           jna    short timer_int_return ; 23/05/2016

                                        ; 15/01/2017

                                        ; present process must be changed with high priority process
                                        ;xor   al, al
0000073D 31C0                           xor    eax, eax ; 26/12/2016
0000073F A2[5D5B0100]                   mov    [p_change], al ; 0
                                        ;mov   byte [priority], 2 ; 15/01/2017 (there is a timer event)

00000744 803D[5B030600]FF               cmp     byte [sysflg], 0FFh ; user or system space ?
0000074B 7416                           je     short rtc_int_3     ; user space ([sysflg]= 0FFh)

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

0000074D A2[A8030600]                   mov    [u.quant], al ; 0

                                timer_int_return: ; 23/05/2016 - jump from 'rtc_int' ('rtc_int_2')
00000752 8B0D[5C040600]                 mov    ecx, [cr3reg]       ; previous value/content of cr3 register
00000758 0F22D9                         mov    cr3, ecx     ; restore cr3 register content
                                        ;
0000075B 61                             popad ; edi, esi, ebp, temp (icrement esp by 4), ebx, edx, ecx, eax
                                        ;
0000075C 0FA9                           pop    gs
0000075E 0FA1                           pop    fs
00000760 07                             pop    es
00000761 1F                             pop    ds
                                        ;
00000762 CF                             iretd  ; return from interrupt

                                rtc_int_3:
00000763 FE05[5B030600]                 inc    byte [sysflg]       ; now, we are in system space
                                        ;
00000769 E961BD0000                       jmp     sysrelease ; change running process immediatelly

                                T8:
                                        ; 13/01/2017 (eax -> ebx)
                                        ; callback checking... (19/12/2016)
0000076E 31DB                           xor    ebx, ebx
00000770 871D[D0030600]                 xchg   ebx, [u.tcb] ; callback address (0 = normal return)
00000776 09DB                           or     ebx, ebx
00000778 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)
0000077A FE05[D4030600]                 inc    byte [u.t_lock]
00000780 8A0D[5B030600]                 mov    cl, [sysflg]
00000786 880D[D5030600]                 mov    [u.t_mode], cl

0000078C 8B2D[644D0100]                 mov    ebp, [tss.esp0] ; kernel stack address (for ring 0)
00000792 83ED14                         sub    ebp, 20             ; eip, cs, eflags, esp, ss
00000795 892D[5C030600]                 mov    [u.sp], ebp
0000079B 8925[60030600]                 mov    [u.usp], esp

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

000007A1 8B44241C                       mov    eax, [esp+28] ; pushed eax
000007A5 A3[64030600]                   mov    [u.r0], eax

000007AA E878DF0000                     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

000007AF C605[5B030600]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)

000007B6 8B4510                         mov    eax, [ebp+16]; SS (UDATA
000007B9 89E6                           mov    esi, esp
```

```
1530 000007BB 50                        push   eax
1531 000007BC 50                        push   eax
1532 000007BD 89E7                      mov    edi, esp
1533 000007BF 893D[60030600]            mov    [u.usp], edi
1534 000007C5 B908000000                mov    ecx, ((ESPACE/4) - 4) ; except DS, ES, FS, GS
1535 000007CA F3A5                      rep    movsd
1536 000007CC B104                      mov    cl, 4
1537 000007CE F3AB                      rep    stosd
1538 000007D0 893D[5C030600]            mov    [u.sp], edi
1539 000007D6 89EE                      mov    esi, ebp
1540 000007D8 B105                      mov    cl, 5 ; EIP (u), CS (UCODE), EFLAGS (u), ESP (u), SS (UDATA)
1541 000007DA F3A5                      rep    movsd
1542
1543 000007DC 8B0D[B8030600]            mov    ecx, [u.pgdir]
1544 000007E2 890D[5C040600]            mov    [cr3reg], ecx
1545
1546                                    ; 13/01/207 (eax -> ebx)
1547                                    ; EBX = callback routine address (virtual, not physical address!)
1548
1549                                    ; 09/01/2017
1550                                    ; !!! CALLBACK ROUTINE MUST BE ENDED/RETURNED WITH 'sysrele'
1551                                    ;     system call !!!
1552                                    ; 25/12/2016
1553                                    ; Callback Note: (19/12/2016)
1554                                    ; !!! CALLBACK ROUTINE MUST BE ENDED/RETURNED WITH 'RETN' !!!
1555                                    ;      pushf ; save flags
1556                                    ;      <callback service code>
1557                                    ;      popf  ; restore flags
1558                                    ;      retn ; return to normal running address
1559                                    ;
1560
1561                                    ; 15/01/2017
1562                                    ; 14/01/2017
1563                                    ; 13/01/2017 (eax -> ebx)
1564                                    ; 10/01/2017
1565                                set_callback_addr:
1566                                    ; 09/01/2017 (**)
1567                                    ; 02/01/2017 (*)
1568                                    ; 25/12/2016 (*)
1569                                    ; 19/12/2016 (TRDOS 386 feature only!)
1570                                    ;
1571                                    ; This routine sets return address
1572                                    ; to start of user's interrupt
1573                                    ; service (callback) address
1574                                    ;; and sets callback 'retn' address to normal
1575                                    ;; return address of user's running code!
1576                                    ;
1577                                    ; INPUT:
1578                                    ;      EBX = callback routine/service address
1579                                    ;            (virtual, not physical address!)
1580                                    ;      [u.sp] = kernel stack, points to
1581                                    ;               user's EIP,CS,EFLAGS,ESP,SS
1582                                    ;               registers.
1583                                    ; OUTPUT:
1584                                    ;      EIP (user) = callback (service) address
1585                                    ;      CS (user) = UCODE
1586                                    ;      EFLAGS (user) = flags before callback
1587                                    ;        ESP (user) = ESP-4 (user, before callback)
1588                                    ;      [ESP](user) = EIP (user) before callback
1589                                    ;
1590                                    ; Note: If CPU was in user mode while entering
1591                                    ;      the timer interrupt service routine,
1592                                    ;      'IRET' will get return to callback routine
1593                                    ;      immediately. If CPU was in system/kernel mode
1594                                    ;      'iret' will get return to system call and
1595                                    ;      then, callback routine will be return address
1596                                    ;      from system call. (User's callback/service code
1597                                    ;      will be able to return to normal return address
1598                                    ;      via an 'retn' at the end.)
1599                                    ;
1600                                    ; Note(**): User's callback service code must be ended
1601                                    ;      with a 'sysrele' sytstem call ! (09/01/2017)
1602                                    ;
1603                                    ;      For example:
1604                                    ;
1605                                    ;      timer_callback:
1606                                    ;          ...
1607                                    ;          inc      dword [time_counter]
1608                                    ;          ...
1609                                    ;          mov eax, 39 ; 'sysrele'
1610                                    ;          int 40h ; TRDOS 386 system call (interrupt)
1611                                    ;
1612                                    ;
1613                                    ;; Note(*): User's callback service code must preserve cpu
1614                                    ;;    flags if it has any instructions which changes
1615                                    ;;    flags in the service code. (25/12/2016)
1616                                    ;;
1617                                    ;;    For example:
1618                                    ;;
1619                                    ;;    timer_callback:
1620                                    ;;        pushf ; save flags
1621                                    ;;        ; this instruction changes zero flag
1622                                    ;;        inc      dword [time_counter]
1623                                    ;;        popf ; restore flags
1624                                    ;;        retn ; return to normal user code
1625                                    ;;             (which is interrupted by the
1626                                    ;;              timer interput)
1627                                    ;;
1628
1629                                    ; 15/01/2017
1630 000007E8 8B2D[5C030600]            mov    ebp, [u.sp]; kernel's stack, points to EIP (user)
1631 000007EE 895D00                    mov    [ebp], ebx
```

```
1632 000007F1 E95CFFFFFF                jmp    timer_int_return
1633
1634                                    ; 15/01/2017
1635                                    ; 13/01/2017
1636                                    ; 19/12/2016
1637                                    ; 06/06/2016
1638                                    ; 23/05/2016
1639                                    ; 22/05/2016
1640                                    ; 19/05/2016 - TRDOS 386 (TRDOS v2.0)
1641                                    ; 26/02/2015
1642                                    ; 07/09/2014
1643                                    ; 25/08/2014
1644                                rtc_int:       ; Real Time Clock Interrupt (IRQ 8)
1645                                    ; 22/05/2016
1646 000007F6 1E                       push   ds ; ** ; 23/05/2016
1647 000007F7 50                       push   eax ; *
1648 000007F8 66B81000                 mov    ax, KDATA
1649 000007FC 8ED8                     mov    ds, ax
1650                                    ;
1651 000007FE 8A25[464E0100]           mov    ah, [RTC_2Hz] ;  2 Hz interrupt to 1 Hz function
1652 00000804 80F401                   xor    ah, 1
1653 00000807 8825[464E0100]           mov    [RTC_2Hz], ah ; 1 = 0.5 second, 0 = 1 second
1654 0000080D 753B                     jnz    short rtc_int_return ; half second
1655                                    ; 1 second
1656                                rtc_int_0:
1657                                    ; 22/05/2016
1658 0000080F 58                       pop    eax ; *
1659                                    ;
1660                                    ; 14/10/2015 ('timer_int')
1661                                    ; Here, we are simulating system call entry (for task switch)
1662                                    ; (If multitasking is enabled,
1663                                    ; 'clock' procedure may jump to 'sysrelease')
1664                                    ;push  ds ; ** ; 23/05/2016
1665 00000810 06                       push   es
1666 00000811 0FA0                     push   fs
1667 00000813 0FA8                     push   gs
1668 00000815 60                       pushad  ; eax, ecx, edx, ebx, esp -before pushad-, ebp, esi, edi
1669 00000816 66B91000                 mov     cx, KDATA
1670                                      ;mov    ds, cx ; 06/06/2016
1671 0000081A 8EC1                       mov     es, cx
1672 0000081C 8EE1                       mov     fs, cx
1673 0000081E 8EE9                       mov     gs, cx
1674                                    ;
1675 00000820 0F20D9                   mov    ecx, cr3
1676 00000823 890D[5C040600]           mov    [cr3reg], ecx ; save current cr3 register value/content
1677                                    ;
1678 00000829 803D[D4030600]00         cmp    byte [u.t_lock], 0 ; timer lock (callback) status ?
1679 00000830 7711                     ja     short rtc_int_1        ; yes
1680
1681                                    ; 15/01/2017
1682 00000832 3B0D[C84D0100]           cmp    ecx, [k_page_dir]
1683 00000838 7409                     je     short rtc_int_1
1684
1685 0000083A 8B0D[C84D0100]           mov    ecx, [k_page_dir]
1686 00000840 0F22D9                   mov    cr3, ecx
1687                                rtc_int_1:
1688                                    ; Timer event (kernel) functions must be performed with
1689                                    ; 1 second intervals - TRDOS 386 (TRDOS v2.0) feature ! -
1690                                    ;
1691                                    ; 25/08/2014
1692 00000843 E81A030000               call   rtc_p  ; 19/05/2016 - major modification
1693
1694                                    ; 23/05/2016
1695 00000848 28E4                     sub    ah, ah ; 0
1696                                    ; 22/05/2016 - TRDOS 386 timer event modifications
1697                                rtc_int_return: ; 19/05/2016
1698                                    ; 22/02/2015 - dsectpm.s
1699                                    ; [ source: http://wiki.osdev.org/RTC ]
1700                                    ; read status register C to complete procedure
1701                                    ;(it is needed to get a next IRQ 8)
1702 0000084A B00C                     mov    al, 0Ch ;
1703 0000084C E670                     out    70h, al ; select register C
1704 0000084E 90                       nop
1705 0000084F E471                     in     al, 71h ; just throw away contents
1706                                    ; 22/02/2015
1707 00000851 B020                     MOV    AL,EOI       ; END OF INTERRUPT
1708                                    ;CLI                ; DISABLE INTERRUPTS TILL STACK CLEARED
1709 00000853 E6A0                     OUT    INTB00,AL    ; FOR CONTROLLER #2
1710
1711                                    ; 23/05/2016
1712 00000855 B020                     MOV    AL,EOI       ; GET END OF INTERRUPT MASK
1713 00000857 FA                       CLI                 ; DISABLE INTERRUPTS TILL STACK CLEARED
1714 00000858 E620                     OUT    INTA00,AL    ; END OF INTERRUPT TO 8259 - 1
1715                                    ;
1716                                    ; 23/05/2016
1717 0000085A 20E4                     and    ah, ah
1718 0000085C 0F84B7FEFFFF               jz      rtc_int_2
1719
1720                                    ; ah = 1 (half second)
1721 00000862 58                       pop    eax ; *
1722 00000863 1F                       pop    ds  ; **
1723 00000864 CF                       iretd
1724
1725                                ; ///////////////
1726
1727                                    ; 28/08/2014
1728                                irq0:
1729 00000865 6A00                       push       dword 0
1730 00000867 EB48                     jmp    short which_irq
1731                                irq1:
1732 00000869 6A01                       push       dword 1
1733 0000086B EB44                     jmp    short which_irq
```

```
                                irq2:
0000086D 6A02                           push    dword 2
0000086F EB40                   irq3:   jmp     short which_irq
                                        ; 20/11/2015
                                        ; 24/10/2015
00000871 2EFF15[9DF50000]               call    dword [cs:com2_irq3]
00000878 6A03                           push    dword 3
0000087A EB35                           jmp     short which_irq
                                irq4:
                                        ; 20/11/2015
                                        ; 24/10/2015
0000087C 2EFF15[99F50000]               call    dword [cs:com1_irq4]
00000883 6A04                           push    dword 4
00000885 EB2A                           jmp     short which_irq
                                irq5:
00000887 6A05                           push    dword 5
00000889 EB26                           jmp     short which_irq
                                irq6:
0000088B 6A06                           push    dword 6
0000088D EB22                           jmp     short which_irq
                                irq7:
0000088F 6A07                           push    dword 7
00000891 EB1E                           jmp     short which_irq
                                irq8:
00000893 6A08                           push    dword 8
00000895 EB1A                           jmp     short which_irq
                                irq9:
00000897 6A09                           push    dword 9
00000899 EB16                           jmp     short which_irq
                                irq10:
0000089B 6A0A                           push    dword 10
0000089D EB12                           jmp     short which_irq
                                irq11:
0000089F 6A0B                           push    dword 11
000008A1 EB0E                           jmp     short which_irq
                                irq12:
000008A3 6A0C                           push    dword 12
000008A5 EB0A                           jmp     short which_irq
                                irq13:
000008A7 6A0D                           push    dword 13
000008A9 EB06                           jmp     short which_irq
                                irq14:
000008AB 6A0E                           push    dword 14
000008AD EB02                           jmp     short which_irq
                                irq15:
000008AF 6A0F                           push    dword 15
                                        ;jmp    short which_irq

                                        ; 22/01/2017
                                        ; 19/10/2015
                                        ; 29/08/2014
                                        ; 21/08/2014
                                which_irq:
000008B1 870424                         xchg    eax, [esp]  ; 28/08/2014
000008B4 53                             push    ebx
000008B5 56                             push    esi
000008B6 57                             push    edi
000008B7 1E                             push    ds
000008B8 06                             push    es
                                        ;
000008B9 88C3                           mov     bl, al
                                        ;
000008BB B810000000                     mov     eax, KDATA
000008C0 8ED8                           mov     ds, ax
000008C2 8EC0                           mov     es, ax
                                        ; 19/10/2015
000008C4 FC                             cld
                                          ; 27/08/2014
000008C5 8105[DC0B0100]A000-              add     dword [scr_row], 0A0h
000008CD 0000
                                        ;
000008CF B417                           mov     ah, 17h      ; blue (1) background,
                                                             ; light gray (7) forecolor
000008D1 8B3D[DC0B0100]                   mov     edi, [scr_row]
000008D7 B049                           mov     al, 'I'
000008D9 66AB                           stosw
000008DB B052                           mov     al, 'R'
000008DD 66AB                           stosw
000008DF B051                           mov     al, 'Q'
000008E1 66AB                           stosw
000008E3 B020                           mov     al, ' '
000008E5 66AB                           stosw
000008E7 88D8                           mov     al, bl
000008E9 3C0A                           cmp     al, 10
000008EB 7208                           jb      short ii1
000008ED B031                           mov     al, '1'
000008EF 66AB                           stosw
000008F1 88D8                           mov     al, bl
000008F3 2C0A                           sub     al, 10
                                ii1:
000008F5 0430                           add     al, '0'
000008F7 66AB                           stosw
000008F9 B020                           mov     al, ' '
000008FB 66AB                           stosw
000008FD B021                           mov     al, '!'
000008FF 66AB                           stosw
00000901 B020                           mov     al, ' '
00000903 66AB                           stosw
                                        ; 23/02/2015
00000905 80FB07                         cmp     bl, 7 ; check for IRQ 8 to IRQ 15
00000908 7604                           jna     ii2
```

```
                                        ; 22/01/2017
0000090A B020                           mov    al, 20h  ; END OF INTERRUPT COMMAND TO
0000090C E6A0                           out    0A0h, al ; the 2nd 8259
                                ii2:
0000090E B020                           mov    al, 20h  ; END OF INTERRUPT COMMAND TO
00000910 E620                           out    20h, al ; the 2nd 8259
00000912 E9CD010000                     jmp     iiret
                                        ;
                                        ; 22/08/2014
                                        ;mov   al, 20h ; END OF INTERRUPT COMMAND TO 8259
                                        ;out   20h, al      ; 8259 PORT
                                        ;
                                        ;pop   es
                                        ;pop   ds
                                        ;pop   edi
                                        ;pop   esi
                                        ;pop   ebx
                                        ;pop   eax
                                        ;iret

                                        ; 02/04/2015
                                        ; 25/08/2014
                                exc0:
00000917 6A00                             push      dword 0
00000919 E990000000                       jmp     cpu_except
                                exc1:
0000091E 6A01                             push      dword 1
00000920 E989000000                       jmp     cpu_except
                                exc2:
00000925 6A02                             push      dword 2
00000927 E982000000                       jmp     cpu_except
                                exc3:
0000092C 6A03                             push      dword 3
0000092E EB7E                             jmp     cpu_except
                                exc4:
00000930 6A04                             push      dword 4
00000932 EB7A                             jmp     cpu_except
                                exc5:
00000934 6A05                             push      dword 5
00000936 EB76                             jmp     cpu_except
                                exc6:
00000938 6A06                             push      dword 6
0000093A EB72                             jmp     cpu_except
                                exc7:
0000093C 6A07                             push      dword 7
0000093E EB6E                             jmp     cpu_except
                                exc8:
                                        ; [esp] = Error code
00000940 6A08                             push      dword 8
00000942 EB5C                             jmp     cpu_except_en
                                exc9:
00000944 6A09                             push      dword 9
00000946 EB66                             jmp     cpu_except
                                exc10:
                                        ; [esp] = Error code
00000948 6A0A                             push      dword 10
0000094A EB54                             jmp     cpu_except_en
                                exc11:
                                        ; [esp] = Error code
0000094C 6A0B                             push      dword 11
0000094E EB50                             jmp     cpu_except_en
                                exc12:
                                        ; [esp] = Error code
00000950 6A0C                             push      dword 12
00000952 EB4C                             jmp     cpu_except_en
                                exc13:
                                        ; [esp] = Error code
00000954 6A0D                             push      dword 13
00000956 EB48                             jmp     cpu_except_en
                                exc14:
                                        ; [esp] = Error code
00000958 6A0E                             push      dword 14
0000095A EB44                           jmp    short cpu_except_en
                                exc15:
0000095C 6A0F                             push      dword 15
0000095E EB4E                             jmp     cpu_except
                                exc16:
00000960 6A10                             push      dword 16
00000962 EB4A                             jmp     cpu_except
                                exc17:
                                        ; [esp] = Error code
00000964 6A11                             push      dword 17
00000966 EB38                           jmp    short cpu_except_en
                                exc18:
00000968 6A12                             push      dword 18
0000096A EB42                           jmp    short cpu_except
                                exc19:
0000096C 6A13                             push      dword 19
0000096E EB3E                           jmp    short cpu_except
                                exc20:
00000970 6A14                             push      dword 20
00000972 EB3A                           jmp    short cpu_except
                                exc21:
00000974 6A15                             push      dword 21
00000976 EB36                           jmp    short cpu_except
                                exc22:
00000978 6A16                             push      dword 22
0000097A EB32                           jmp    short cpu_except
                                exc23:
0000097C 6A17                             push      dword 23
0000097E EB2E                           jmp    short cpu_except
                                exc24:
```

```
1938 00000980 6A18                      push       dword 24
1939 00000982 EB2A                    jmp    short cpu_except
1940                                exc25:
1941 00000984 6A19                      push       dword 25
1942 00000986 EB26                    jmp    short cpu_except
1943                                exc26:
1944 00000988 6A1A                      push       dword 26
1945 0000098A EB22                    jmp    short cpu_except
1946                                exc27:
1947 0000098C 6A1B                      push       dword 27
1948 0000098E EB1E                    jmp    short cpu_except
1949                                exc28:
1950 00000990 6A1C                      push       dword 28
1951 00000992 EB1A                    jmp    short cpu_except
1952                                exc29:
1953 00000994 6A1D                      push       dword 29
1954 00000996 EB16                    jmp    short cpu_except
1955                                exc30:
1956 00000998 6A1E                      push       dword 30
1957 0000099A EB04                    jmp    short cpu_except_en
1958                                exc31:
1959 0000099C 6A1F                      push       dword 31
1960 0000099E EB0E                      jmp     short cpu_except
1961
1962                                  ; 19/10/2015
1963                                  ; 19/09/2015
1964                                  ; 01/09/2015
1965                                  ; 28/08/2015
1966                                  ; 28/08/2014
1967                                cpu_except_en:
1968 000009A0 87442404                xchg   eax, [esp+4] ; Error code
1969 000009A4 36A3[78050600]          mov    [ss:error_code], eax
1970 000009AA 58                      pop    eax  ; Exception number
1971 000009AB 870424                  xchg   eax, [esp]
1972                                          ; eax = eax before exception
1973                                          ; [esp] -> exception number
1974                                          ; [esp+4] -> EIP to return
1975                                  ; 22/01/2017
1976                                  ; 19/10/2015
1977                                  ; 19/09/2015
1978                                  ; 01/09/2015
1979                                  ; 28/08/2015
1980                                  ; 29/08/2014
1981                                  ; 28/08/2014
1982                                  ; 25/08/2014
1983                                  ; 21/08/2014
1984                              cpu_except:  ; CPU Exceptions
1985 000009AE FC                      cld
1986 000009AF 870424                  xchg   eax, [esp]
1987                                          ; eax = Exception number
1988                                          ; [esp] = eax (before exception)
1989 000009B2 53                      push   ebx
1990 000009B3 56                      push   esi
1991 000009B4 57                      push   edi
1992 000009B5 1E                      push   ds
1993 000009B6 06                      push   es
1994                                  ; 28/08/2015
1995 000009B7 66BB1000                mov    bx, KDATA
1996 000009BB 8EDB                    mov    ds, bx
1997 000009BD 8EC3                    mov    es, bx
1998 000009BF 0F20DB                  mov    ebx, cr3
1999 000009C2 53                      push   ebx ; (*) page directory
2000                                  ; 19/10/2015
2001 000009C3 FC                      cld
2002                                  ; 25/03/2015
2003 000009C4 8B1D[C84D0100]          mov    ebx, [k_page_dir]
2004 000009CA 0F22DB                  mov    cr3, ebx
2005                                  ; 28/08/2015
2006 000009CD 83F80E                  cmp    eax, 0Eh ; 14, PAGE FAULT
2007 000009D0 750F                    jne    short cpu_except_nfp
2008 000009D2 E87A440000              call   page_fault_handler
2009 000009D7 21C0                    and    eax, eax
2010 000009D9 0F8401010000              jz   iiretp ; 01/09/2015
2011 000009DF B00E                    mov    al, 0Eh ; 14
2012                              cpu_except_nfp:
2013                                  ; 23/08/2016
2014 000009E1 803D[C25E0000]03        cmp    byte [CRT_MODE], 3
2015 000009E8 7409                    je     short cpu_except_mode_3
2016 000009EA 50                      push   eax
2017 000009EB B003                    mov    al, 3
2018 000009ED E8730B0000              call   _set_mode
2019 000009F2 58                      pop    eax
2020                              cpu_except_mode_3:
2021                                  ; 02/04/2015
2022 000009F3 BB[42060000]            mov    ebx, hang
2023 000009F8 875C241C                xchg   ebx, [esp+28]
2024                                          ; EIP (points to instruction which faults)
2025                                          ; New EIP (hang)
2026 000009FC 891D[7C050600]          mov    [FaultOffset], ebx
2027 00000A02 C744242008000000        mov    dword [esp+32], KCODE ; kernel's code segment
2028 00000A0A 814C242400020000        or     dword [esp+36], 200h ; enable interrupts (set IF)
2029                                  ;
2030 00000A12 88C4                    mov    ah, al
2031 00000A14 240F                    and    al, 0Fh
2032 00000A16 3C09                    cmp    al, 9
2033 00000A18 7602                    jna    short h1ok
2034 00000A1A 0407                    add    al, 'A'-':'
2035                              h1ok:
2036 00000A1C C0EC04                  shr    ah, 4
2037 00000A1F 80FC09                  cmp    ah, 9
2038 00000A22 7603                    jna    short h2ok
2039 00000A24 80C407                  add    ah, 'A'-':'
```

```
2040                                    h2ok:
2041 00000A27 86E0                              xchg   ah, al
2042 00000A29 66053030                          add    ax, '00'
2043 00000A2D 66A3[340E0100]                    mov    [excnstr], ax
2044                                            ;
2045                                            ; 29/08/2014
2046 00000A33 A1[7C050600]                      mov    eax, [FaultOffset]
2047 00000A38 51                                push   ecx
2048 00000A39 52                                push   edx
2049 00000A3A 89E3                              mov    ebx, esp
2050                                            ; 28/08/2015
2051 00000A3C B910000000                        mov    ecx, 16         ; divisor value to convert binary number
2052                                                                   ; to hexadecimal string
2053                                            ;mov   ecx, 10          ; divisor to convert
2054                                                                   ; binary number to decimal string
2055                                    b2d1:
2056 00000A41 31D2                              xor    edx, edx
2057 00000A43 F7F1                              div    ecx
2058 00000A45 6652                              push   dx
2059 00000A47 39C8                              cmp    eax, ecx
2060 00000A49 73F6                              jnb    short b2d1
2061 00000A4B BF[3F0E0100]                      mov    edi, EIPstr ; EIP value
2062                                                                   ; points to instruction which faults
2063                                            ; 28/08/2015
2064 00000A50 89C2                              mov    edx, eax
2065                                    b2d2:
2066                                            ;add   al, '0'
2067 00000A52 8A82[1B330000]                    mov    al, [edx+hexchrs]
2068 00000A58 AA                                stosb                  ; write hexadecimal digit to its place
2069 00000A59 39E3                              cmp    ebx, esp
2070 00000A5B 7606                              jna    short b2d3
2071 00000A5D 6658                              pop    ax
2072 00000A5F 88C2                              mov    dl, al
2073 00000A61 EBEF                              jmp    short b2d2
2074                                    b2d3:
2075 00000A63 B068                              mov    al, 'h' ; 28/08/2015
2076 00000A65 AA                                stosb
2077 00000A66 B020                              mov    al, 20h          ; space
2078 00000A68 AA                                stosb
2079 00000A69 30C0                              xor    al, al     ; to do it an ASCIIZ string
2080 00000A6B AA                                stosb
2081                                            ;
2082 00000A6C 5A                                pop    edx
2083 00000A6D 59                                pop    ecx
2084                                            ;
2085 00000A6E B44F                              mov    ah, 4Fh      ; red (4) background,
2086                                                                  ; white (F) forecolor
2087 00000A70 BE[240E0100]                      mov    esi, exc_msg ; message offset
2088                                            ;
2089                                            ; 20/01/2017 (!cpu exception!)
2090                                            ;
2091 00000A75 8105[DC0B0100]A000-                 add    dword [scr_row], 0A0h
2092 00000A7D 0000
2093 00000A7F 8B3D[DC0B0100]                      mov    edi, [scr_row]
2094                                            ;
2095 00000A85 C605[5B030600]00                  mov    byte [sysflg], 0  ; system mode
2096 00000A8C FB                                  sti
2097                                            ;
2098 00000A8D E8EFFBFFFF                        call   printk
2099                                            ;
2100 00000A92 B410                              mov    ah, 10h
2101 00000A94 E87D010000                        call   int16h ; getc
2102                                            ;
2103 00000A99 B003                              mov    al, 3
2104 00000A9B E8C50A0000                        call   _set_mode
2105                                            ;
2106 00000AA0 B801000000                        mov    eax, 1
2107 00000AA5 E98CBB0000                        jmp    sysexit ; terminate process !!!
2108
2109                                            ; 22/01/2017
2110                                            ; 18/04/2016
2111                                            ; 28/08/2015
2112                                            ; 23/02/2015
2113                                            ; 20/08/2014
2114                                    ignore_int:
2115 00000AAA 50                                push   eax
2116 00000AAB 53                                push   ebx ; 23/02/2015
2117 00000AAC 56                                push   esi
2118 00000AAD 57                                push   edi
2119 00000AAE 1E                                push   ds
2120 00000AAF 06                                push   es
2121                                            ; 18/04/2016
2122 00000AB0 66B81000                          mov    ax, KDATA
2123 00000AB4 8ED8                              mov    ds, ax
2124 00000AB6 8EC0                              mov    es, ax
2125                                            ; 28/08/2015
2126 00000AB8 0F20D8                            mov    eax, cr3
2127 00000ABB 50                                push   eax ; (*) page directory
2128                                            ;
2129 00000ABC B467                              mov    ah, 67h      ; brown (6) background,
2130                                                                  ; light gray (7) forecolor
2131 00000ABE BE[EC0C0100]                      mov    esi, int_msg ; message offset
2132                                    piemsg:
2133                                              ; 27/08/2014
2134 00000AC3 8105[DC0B0100]A000-                 add     dword [scr_row], 0A0h
2135 00000ACB 0000
2136 00000ACD 8B3D[DC0B0100]                      mov     edi, [scr_row]
2137                                              ;
2138 00000AD3 E8A9FBFFFF                        call   printk
2139                                            ;
2140                                            ; 23/02/2015
2141 00000AD8 B020                              mov    al, 20h  ; END OF INTERRUPT COMMAND TO
```

```
2142 00000ADA E6A0                      out    0A0h, al ; the 2nd 8259
2143                                    ; 22/08/2014
2144 00000ADC B020                      mov    al, 20h ; END OF INTERRUPT COMMAND TO 8259
2145 00000ADE E620                      out    20h, al      ; 8259 PORT
2146                                iiretp:
2147                                    ; 22/01/2017
2148                                    ; 01/09/2015
2149                                    ; 28/08/2015
2150 00000AE0 58                        pop    eax ; (*) page directory
2151 00000AE1 0F22D8                    mov    cr3, eax
2152                                iiret:
2153 00000AE4 07                        pop    es
2154 00000AE5 1F                        pop    ds
2155 00000AE6 5F                        pop    edi
2156 00000AE7 5E                        pop    esi
2157 00000AE8 5B                        pop    ebx ; 29/08/2014
2158 00000AE9 58                        pop    eax
2159 00000AEA CF                        iretd
2160
2161                                    ; 23/05/2016
2162                                    ; 22/08/2014
2163                                    ; IBM PC/AT BIOS source code ----- 10/06/85 (bios.asm)
2164                                    ; (INT 1Ah)
2165                                    ;; Linux (v0.12) source code (main.c) by Linus Torvalds (1991)
2166                                time_of_day:
2167 00000AEB E8ED500000                call   UPD_IPR                  ; WAIT TILL UPDATE NOT IN PROGRESS
2168 00000AF0 726F                        jc     short time_of_day_retn ; 23/05/2016
2169 00000AF2 B000                      mov    al, CMOS_SECONDS
2170 00000AF4 E8FF500000                call   CMOS_READ
2171 00000AF9 A2[384E0100]              mov    [time_seconds], al
2172 00000AFE B002                      mov    al, CMOS_MINUTES
2173 00000B00 E8F3500000                call   CMOS_READ
2174 00000B05 A2[394E0100]              mov    [time_minutes], al
2175 00000B0A B004                      mov    al, CMOS_HOURS
2176 00000B0C E8E7500000                call   CMOS_READ
2177 00000B11 A2[3A4E0100]                mov    [time_hours], al
2178 00000B16 B006                      mov    al, CMOS_DAY_WEEK
2179 00000B18 E8DB500000                call   CMOS_READ
2180 00000B1D A2[3B4E0100]              mov    [date_wday], al
2181 00000B22 B007                      mov    al, CMOS_DAY_MONTH
2182 00000B24 E8CF500000                call   CMOS_READ
2183 00000B29 A2[3C4E0100]              mov    [date_day], al
2184 00000B2E B008                      mov    al, CMOS_MONTH
2185 00000B30 E8C3500000                call   CMOS_READ
2186 00000B35 A2[3D4E0100]              mov    [date_month], al
2187 00000B3A B009                      mov    al, CMOS_YEAR
2188 00000B3C E8B7500000                call   CMOS_READ
2189 00000B41 A2[3E4E0100]              mov    [date_year], al
2190 00000B46 B032                      mov    al, CMOS_CENTURY
2191 00000B48 E8AB500000                call   CMOS_READ
2192 00000B4D A2[3F4E0100]              mov    [date_century], al
2193                                    ;
2194 00000B52 B000                      mov    al, CMOS_SECONDS
2195 00000B54 E89F500000                call   CMOS_READ
2196 00000B59 3A05[384E0100]            cmp    al, [time_seconds]
2197 00000B5F 758A                      jne    short time_of_day
2198
2199                                time_of_day_retn:
2200 00000B61 C3                        retn
2201
2202                                    ; 15/01/2017
2203                                    ; 10/06/2016
2204                                    ; 07/06/2016
2205                                    ; 06/06/2016
2206                                    ; 23/05/2016
2207                                rtc_p:
2208 00000B62 B101                      mov    cl, 1 ; 15/01/2017
2209 00000B64 EB02                      jmp    short rtc_p0
2210                                u_timer:
2211                                    ; Timer Events with 18.2 Hz Timer Ticks
2212                                    ; (and also timer events with RTC seconds)
2213 00000B66 28C9                      sub    cl, cl ; mov cl, 0 ; 15/01/2017
2214                                rtc_p0:
2215                                    ; 19/05/2016 - TRDOS 386 (TRDOS v2.0)
2216                                    ; Major Modification:
2217                                    ; Check and Perform Timer Events (for RTC)
2218                                    ; 25/08/2014 - 07/09/2014
2219                                    ; Retro UNIX 386 v1:
2220                                    ; Print Real Time Clock content
2221
2222                                    ; 15/01/2017
2223 00000B68 880D[5C5B0100]            mov    byte [priority], cl ; 0 or 1 (not 2)
2224 00000B6E 8A2D[5F5B0100]            mov    ch, [timer_events]
2225 00000B74 20ED                      and    ch, ch
2226 00000B76 7420                      jz     short rtc_p3
2227
2228 00000B78 BE[60040600]              mov    esi, timer_set  ; beginning address of
2229                                                            ; timer events space
2230                                rtc_p1:
2231 00000B7D 8B06                      mov    eax, [esi]
2232 00000B7F 20C0                      and    al, al ; 0 = free, >0 = process no.
2233 00000B81 7416                      jz     short rtc_p4
2234                                    ;
2235 00000B83 C1C810                    ror    eax, 16
2236                                    ; ah = response value, al = interrupt type
2237                                    ; 15/01/2017
2238                                    ; cl = interrupt source
2239                                    ;       1 = RTC, 0 = PIT
2240 00000B86 38C8                      cmp    al, cl
2241 00000B88 750A                      jne    short rtc_p2 ; not as requested or undefined !
2242 00000B8A 3C01                      cmp    al, 1 ; 1 ; RTC interrupt ?
2243 00000B8C 7410                      je     short rtc_p5 ; yes, check for response
```

```
                                        ; 06/06/2016 - 18.2 Hz Timer Ticks
00000B8E 836E080A                       sub    dword [esi+8], 10 ; 1 tick = 10
00000B92 7613                           jna    short rtc_p6  ; continue for responding
                                rtc_p2:
                                        ; 15/01/2017 (cl -> ch)
                                        ; 07/06/2016
00000B94 FECD                           dec    ch    ; remain count of timer events
00000B96 7501                           jnz    short rtc_p4
                                rtc_p3:
00000B98 C3                             retn
                                rtc_p4:
                                        ;cmp   esi, timer_set + 240 ; 15*16 (last event)
                                        ;jnb   short rtc_p3 ; end of timer event space
00000B99 83C610                         add    esi, 16 ; next timer event
00000B9C EBDF                           jmp    short rtc_p1
                                rtc_p5:
                                        ; current timer count ; 06/06/2016 (182)
00000B9E 816E08B6000000                 sub    dword [esi+8], 182 ; 1 second (10*18.2)
00000BA5 77ED                           ja     short rtc_p2  ; check for the next
                                rtc_p6:
                                        ; it is the time of response!
00000BA7 8B5E04                         mov    ebx, [esi+4] ; set (count limit) value
00000BAA 895E08                         mov    [esi+8], ebx ; reset count down value
                                                             ; to count limit
                                        ; 19/12/2016
                                        ; 10/12/2016 - timer callback modification
00000BAD 8B7E0C                         mov    edi, [esi+12] ; response (or callback) address
00000BB0 807E0100                       cmp    byte [esi+1], 0 ; >0 = callback
00000BB4 762A                           jna    short rtc_p8

                                        ; timer callback !
00000BB6 0FB61E                         movzx  ebx, byte [esi] ; process number (>0)
00000BB9 89D8                           mov    eax, ebx
00000BBB C0E302                         shl    bl, 2 ; *4
00000BBE 89BB[0C010600]                 mov    [ebx+p.tcb-4], edi ; user's callback service addr
00000BC4 3A05[B3030600]                 cmp    al, [u.uno]
00000BCA 7521                           jne    short rtc_p9
00000BCC 893D[D0030600]                 mov    [u.tcb], edi
                                rtc_p7:
                                        ; 15/01/2017
00000BD2 B002                           mov    al, 2
00000BD4 A2[5C5B0100]                   mov    [priority], al ; 2
                                        ; 10/01/2017
                                        ;mov   byte [u.pri], 2
00000BD9 A2[A9030600]                   mov    [u.pri], al ; 2
00000BDE 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
00000BE0 8827                           mov    [edi], ah     ; response value
                                        ;
00000BE2 C1C010                         rol    eax, 16
                                        ; 15/01/2017
00000BE5 3A05[B3030600]                 cmp    al, [u.uno] ; running process ?
00000BEB 74E5                           je     short rtc_p7
                                rtc_p9:
                                        ; al = process number  ; 10/06/2016
00000BED B202                           mov    dl, 2 ; priority, 2 = event (high)
00000BEF E8BDE60000                     call   set_run_sequence ; 19/05/2016
00000BF4 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:
00000BF6 6650                           push   ax
00000BF8 B00B                           mov    al, 0Bh  ; In-Service register
00000BFA E620                           out    20h, al
00000BFC EB00                             jmp short $+2
00000BFE EB00                           jmp short $+2
00000C00 E420                           in     al, 20h
00000C02 2480                           and    al, 80h ; bit 7 (is it real IRQ 7 or fake?)
00000C04 7404                             jz     short irq7_iret ; Fake (spurious) IRQ, do not send EOI
00000C06 B020                             mov    al, 20h ; EOI
00000C08 E620                           out    20h, al
                                irq7_iret:
00000C0A 6658                           pop    ax
00000C0C 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)
                                        ;
00000C0D D410                           db 0D4h,10h                     ; Undocumented inst. AAM
                                                                ; AH = AL / 10h
                                                                ; AL = AL MOD 10h
00000C0F 660D3030                       or ax,'00'                      ; Make it ASCII based

00000C13 86E0                             xchg ah, al

00000C15 C3                             retn


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

```
                                <1> ; TRDOS386.ASM (TRDOS 386 Kernel) - v2.0.0 - keyboard.s
                                <1> ; ----------------------------------------------------------------------------
                                <1> ; Last Update: 15/01/2017
                                <1> ; ----------------------------------------------------------------------------
                                <1> ; Beginning: 17/01/2016
                                <1> ; ----------------------------------------------------------------------------
                                <1> ; Assembler: NASM version 2.11 (trdos386.s)
                                <1> ; ----------------------------------------------------------------------------
                                <1> ; Turkish Rational DOS
                                <1> ; Operating System Project v2.0 by ERDOGAN TAN (Beginning: 04/01/2016)
                                <1> ;
                                <1> ; Derived from 'Retro UNIX 386 Kernel - v0.2.1.0' source code by Erdogan Tan
                                <1> ; keyboard.inc (17/10/2015)
                                <1> ;
                                <1> ; Derived from 'IBM PC-XT-286' BIOS source code (1986)
                                <1> ; ****************************************************************************
                                <1>
                                <1> ; Retro UNIX 386 v1 Kernel - KEYBOARD.INC
                                <1> ; Last Modification: 17/10/2015
                                <1> ;               (Keyboard Data is in 'KYBDATA.INC')
                                <1> ;
                                <1> ; ///////// KEYBOARD FUNCTIONS (PROCEDURES) ///////////////
                                <1>
                                <1> ; 17/01/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>
                                <1> ; 03/12/2014
                                <1> ; 26/08/2014
                                <1> ; KEYBOARD I/O
                                <1> ; (INT_16h - Retro UNIX 8086 v1 - U9.ASM, 30/06/2014)
                                <1>
                                <1> ;NOTE: 'k0' to 'k7' are name of OPMASK registers.
                                <1> ;     (The reason of using '_k' labels!!!) (27/08/2014)
                                <1> ;NOTE: 'NOT' keyword is '~' unary operator in NASM.
                                <1> ;     ('NOT LC_HC' --> '~LC_HC') (bit reversing operator)
                                <1>
                                <1> int16h:     ; 30/06/2015
                                <1> ;getc:
00000C16 9C                     <1>       pushfd ; 28/08/2014
00000C17 0E                     <1>       push   cs
00000C18 E801000000             <1>       call   KEYBOARD_IO_1 ; getc_int
00000C1D C3                     <1>       retn
                                <1>
                                <1> getc_int:
                                <1>       ; 28/02/2015
                                <1>       ; 03/12/2014 (derivation from pc-xt-286 bios source code -1986-,
                                <1>       ;           instead of pc-at bios - 1985-)
                                <1>       ; 28/08/2014 (_k1d)
                                <1>       ; 30/06/2014
                                <1>       ; 03/03/2014
                                <1>       ; 28/02/2014
                                <1>       ; Derived from "KEYBOARD_IO_1" procedure of IBM "pc-xt-286"
                                <1>       ; rombios source code (21/04/1986)
                                <1>       ;      'keybd.asm', INT 16H, KEYBOARD_IO
                                <1>       ;
                                <1>       ; KYBD --- 03/06/86  KEYBOARD BIOS
                                <1>       ;
                                <1>       ;--- INT 16 H ----------------------------------------------------------------
                                <1>       ; KEYBOARD I/O                                                          :
                                <1>       ;     THESE ROUTINES PROVIDE READ KEYBOARD SUPPORT                       :
                                <1>       ; INPUT                                                                 :
                                <1>       ;     (AH)= 00H  READ THE NEXT ASCII CHARACTER ENTERED FROM THE KEYBOARD,   :
                                <1>       ;               RETURN THE RESULT IN (AL), SCAN CODE IN (AH).             :
                                <1>       ;               THIS IS THE COMPATIBLE READ INTERFACE, EQUIVALENT TO THE   :
                                <1>       ;                  STANDARD PC OR PCAT KEYBOARD                             :
                                <1>       ;-------------------------------------------------------------------------:
                                <1>       ;     (AH)= 01H  SET THE ZERO FLAG TO INDICATE IF AN ASCII CHARACTER IS     :
                                <1>       ;               AVAILABLE TO BE READ FROM THE KEYBOARD BUFFER.          :
                                <1>       ;               (ZF)= 1 -- NO CODE AVAILABLE                             :
                                <1>       ;               (ZF)= 0 -- CODE IS AVAILABLE  (AX)= CHARACTER            :
                                <1>       ;               IF (ZF)= 0, THE NEXT CHARACTER IN THE BUFFER TO BE READ IS :
                                <1>       ;               IN (AX), AND THE ENTRY REMAINS IN THE BUFFER.             :
                                <1>       ;               THIS WILL RETURN ONLY PC/PCAT KEYBOARD COMPATIBLE CODES    :
                                <1>       ;-------------------------------------------------------------------------:
                                <1>       ;     (AH)= 02H  RETURN THE CURRENT SHIFT STATUS IN AL REGISTER            :
                                <1>       ;               THE BIT SETTINGS FOR THIS CODE ARE INDICATED IN THE        :
                                <1>       ;               EQUATES FOR @KB_FLAG                                      :
                                <1>       ;-------------------------------------------------------------------------:
                                <1>       ;     (AH)= 03H  SET TYPAMATIC RATE AND DELAY                             :
                                <1>       ;           (AL) = 05H                                                    :
                                <1>       ;           (BL) = TYPAMATIC RATE (BITS 5 - 7 MUST BE RESET TO 0)         :
                                <1>       ;                                                                        :
                                <1>       ;                  REGISTER     RATE      REGISTER     RATE             :
                                <1>       ;                   VALUE     SELECTED     VALUE     SELECTED           :
                                <1>       ;                  ----------------------------------------           :
                                <1>       ;                    00H        30.0        10H        7.5              :
                                <1>       ;                    01H        26.7        11H        6.7              :
                                <1>       ;                    02H        24.0        12H        6.0              :
                                <1>       ;                    03H        21.8        13H        5.5              :
                                <1>       ;                    04H        20.0        14H        5.0              :
                                <1>       ;                    05H        18.5        15H        4.6              :
                                <1>       ;                    06H        17.1        16H        4.3              :
                                <1>       ;                    07H        16.0        17H        4.0              :
                                <1>       ;                    08H        15.0        18H        3.7              :
                                <1>       ;                    09H        13.3        19H        3.3              :
                                <1>       ;                    0AH        12.0        1AH        3.0              :
                                <1>       ;                    0BH        10.9        1BH        2.7              :
                                <1>       ;                    0CH        10.0        1CH        2.5              :
                                <1>       ;                    0DH         9.2        1DH        2.3              :
                                <1>       ;                    0EH         8.6        1EH        2.1              :
                                <1>       ;                    0FH         8.0        1FH        2.0              :
                                <1>       ;                                                                       :
                                <1>       ;           (BH) = TYPAMATIC DELAY  (BITS 2 - 7 MUST BE RESET TO 0)       :
```

```
2448                                    <1>     ;                                                       :
2449                                    <1>     ;                   REGISTER     DELAY                      :
2450                                    <1>     ;                    VALUE       VALUE                      :
2451                                    <1>     ;                   ------------------                      :
2452                                    <1>     ;                 00H        250 ms                        :
2453                                    <1>     ;                 01H        500 ms                        :
2454                                    <1>     ;                 02H        750 ms                        :
2455                                    <1>     ;                 03H       1000 ms                        :
2456                                    <1>     ;-----------------------------------------------------------------------:
2457                                    <1>     ;     (AH)= 05H  PLACE ASCII CHARACTER/SCAN CODE COMBINATION IN KEYBOARD    :
2458                                    <1>     ;               BUFFER AS IF STRUCK FROM KEYBOARD                       :
2459                                    <1>     ;               ENTRY:  (CL) = ASCII CHARACTER                        :
2460                                    <1>     ;                       (CH) = SCAN CODE                              :
2461                                    <1>     ;               EXIT:   (AH) = 00H = SUCCESSFUL OPERATION             :
2462                                    <1>     ;                       (AL) = 01H = UNSUCCESSFUL - BUFFER FULL       :
2463                                    <1>     ;               FLAGS:  CARRY IF ERROR                                :
2464                                    <1>     ;-----------------------------------------------------------------------:

2465                                    <1>     ;     (AH)= 10H  EXTENDED READ INTERFACE FOR THE ENHANCED KEYBOARD,         :
2466                                    <1>     ;               OTHERWISE SAME AS FUNCTION AH=0                         :
2467                                    <1>     ;-----------------------------------------------------------------------:
2468                                    <1>     ;     (AH)= 11H  EXTENDED ASCII STATUS FOR THE ENHANCED KEYBOARD,           :
2469                                    <1>     ;               OTHERWISE SAME AS FUNCTION AH=1                         :
2470                                    <1>     ;-----------------------------------------------------------------------:
2471                                    <1>     ;     (AH)= 12H  RETURN THE EXTENDED SHIFT STATUS IN AX REGISTER             :
2472                                    <1>     ;               AL = BITS FROM KB_FLAG, AH = BITS FOR LEFT AND RIGHT       :
2473                                    <1>     ;               CTL AND ALT KEYS FROM KB_FLAG_1 AND KB_FLAG_3              :
2474                                    <1>     ; OUTPUT                                                              :
2475                                    <1>     ;     AS NOTED ABOVE, ONLY (AX) AND FLAGS CHANGED                        :
2476                                    <1>     ;     ALL REGISTERS RETAINED                                              :
2477                                    <1>     ;-----------------------------------------------------------------------
2478                                    <1>
2479                                    <1> ; 15/01/2017
2480                                    <1> ; 14/01/2017
2481                                    <1> ; 02/01/2017
2482                                    <1> ; 29/05/2016
2483                                    <1> ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
2484                                    <1> int32h:  ; Keyboard BIOS
2485                                    <1>
2486                                    <1> KEYBOARD_IO_1:
2487                                    <1>       ;sti                    ; INTERRUPTS BACK ON
2488                                    <1>       ; 29/05/2016
2489 00000C1E 80642408BE                <1>         and     byte [esp+8], 10111110b ; clear zero flag and cary flag
2490                                    <1>       ;
2491 00000C23 1E                       <1>       push   ds               ; SAVE CURRENT DS
2492 00000C24 53                       <1>       push   ebx              ; SAVE BX TEMPORARILY
2493                                    <1>       ;push  ecx              ; SAVE CX TEMPORARILY
2494 00000C25 66BB1000                 <1>         mov     bx, KDATA
2495 00000C29 8EDB                     <1>       mov    ds, bx           ; PUT SEGMENT VALUE OF DATA AREA INTO DS
2496                                    <1>
2497                                    <1>       ; 14/01/2017
2498 00000C2B 8B1C24                   <1>       mov    ebx, [esp]
2499                                    <1>       ;; 15/01/2017
2500                                    <1>       ; 02/01/2017
2501                                    <1>       ;;mov  byte [intflg], 32h  ; keyboard interrupt
2502 00000C2E FB                       <1>       sti
2503                                    <1>       ;
2504                                    <1>
2505 00000C2F 08E4                     <1>       or     ah, ah           ; CHECK FOR (AH)= 00H
2506 00000C31 743A                     <1>       jz     short _K1        ; ASCII_READ
2507 00000C33 FECC                     <1>       dec    ah                  ; CHECK FOR (AH)= 01H
2508 00000C35 7453                     <1>         jz      short _K2           ; ASCII_STATUS
2509 00000C37 FECC                     <1>       dec    ah               ; CHECK FOR (AH)= 02H
2510 00000C39 0F8494000000             <1>         jz      _K3                 ; SHIFT STATUS
2511 00000C3F FECC                     <1>       dec    ah               ; CHECK FOR (AH)= 03H
2512 00000C41 0F8493000000             <1>         jz      _K300               ; SET TYPAMATIC RATE/DELAY
2513 00000C47 80EC02                   <1>       sub    ah, 2            ; CHECK FOR (AH)= 05H
2514 00000C4A 0F84BC000000             <1>         jz      _K500               ; KEYBOARD WRITE
2515                                    <1> _KIO1:
2516 00000C50 80EC0B                   <1>       sub    ah, 11           ; AH =  10H
2517 00000C53 740C                     <1>       jz     short _K1E       ; EXTENDED ASCII READ
2518 00000C55 FECC                     <1>       dec    ah               ; CHECK FOR (AH)= 11H
2519 00000C57 7422                     <1>       jz     short _K2E       ; EXTENDED_ASCII_STATUS
2520 00000C59 FECC                     <1>       dec    ah               ; CHECK FOR (AH)= 12H
2521 00000C5B 7458                     <1>       jz     short _K3E       ; EXTENDED_SHIFT_STATUS
2522                                    <1> _KIO_EXIT:
2523                                    <1>       ; 02/01/2017
2524 00000C5D FA                       <1>       cli
2525                                    <1>       ;;mov  byte [intflg], 0 ;; 15/01/2017
2526                                    <1>       ;
2527                                    <1>       ;pop   ecx              ; RECOVER REGISTER
2528 00000C5E 5B                       <1>       pop    ebx              ; RECOVER REGISTER
2529 00000C5F 1F                       <1>       pop    ds               ; RECOVER SEGMENT
2530 00000C60 CF                       <1>       iretd                   ; INVALID COMMAND, EXIT
2531                                    <1>
2532                                    <1>       ;-----ASCII CHARACTER
2533                                    <1> _K1E:
2534 00000C61 E8D3000000               <1>       call   _K1S             ; GET A CHARACTER FROM THE BUFFER (EXTENDED)
2535 00000C66 E848010000               <1>       call   _KIO_E_XLAT      ; ROUTINE TO XLATE FOR EXTENDED CALLS
2536 00000C6B EBF0                     <1>       jmp    short _KIO_EXIT      ; GIVE IT TO THE CALLER
2537                                    <1> _K1:
2538 00000C6D E8C7000000               <1>       call   _K1S             ; GET A CHARACTER FROM THE BUFFER
2539 00000C72 E847010000               <1>       call   _KIO_S_XLAT      ; ROUTINE TO XLATE FOR STANDARD CALLS
2540 00000C77 72F4                     <1>       jc     short _K1        ; CARRY SET MEANS TROW CODE AWAY
2541                                    <1> _K1A:
2542 00000C79 EBE2                     <1>       jmp    short _KIO_EXIT      ; RETURN TO CALLER
2543                                    <1>
2544                                    <1>       ;-----ASCII STATUS
2545                                    <1> _K2E:
2546 00000C7B E804010000               <1>       call   _K2S             ; TEST FOR CHARACTER IN BUFFER (EXTENDED)
2547 00000C80 7420                     <1>       jz     short _K2B       ; RETURN IF BUFFER EMPTY
2548 00000C82 9C                       <1>       pushf                   ; SAVE ZF FROM TEST
```

```
2549 00000C83 E82B010000          <1>       call   _KIO_E_XLAT         ; ROUTINE TO XLATE FOR EXTENDED CALLS
2550 00000C88 EB17                <1>       jmp    short _K2A           ; GIVE IT TO THE CALLER
2551                              <1> _K2:
2552 00000C8A E8F5000000          <1>       call   _K2S                 ; TEST FOR CHARACTER IN BUFFER
2553 00000C8F 7411                <1>       jz     short _K2B           ; RETURN IF BUFFER EMPTY
2554 00000C91 9C                  <1>       pushf                       ; SAVE ZF FROM TEST
2555 00000C92 E827010000          <1>       call   _KIO_S_XLAT          ; ROUTINE TO XLATE FOR STANDARD CALLS
2556 00000C97 7308                <1>       jnc    short _K2A           ; CARRY CLEAR MEANS PASS VALID CODE
2557 00000C99 9D                  <1>       popf                        ; INVALID CODE FOR THIS TYPE OF CALL
2558 00000C9A E89A000000          <1>       call   _K1S                 ; THROW THE CHARACTER AWAY
2559 00000C9F EBE9                <1>       jmp    short _K2            ; GO LOOK FOR NEXT CHAR, IF ANY
2560                              <1> _K2A:
2561 00000CA1 9D                  <1>       popf                        ; RESTORE ZF FROM TEST
2562                              <1> _K2B:
2563                              <1>       ; 02/01/2017
2564 00000CA2 FA                  <1>       cli
2565                              <1>       ;; mov byte [intflg], 0 ;; 15/01/2017
2566                              <1>       ;
2567                              <1>       ;pop   ecx                  ; RECOVER REGISTER
2568 00000CA3 5B                  <1>       pop    ebx                  ; RECOVER REGISTER
2569 00000CA4 1F                  <1>       pop    ds                   ; RECOVER SEGMENT
2570                              <1>       ; (*) 29/05/2016
2571                              <1>       ; (*) retf 4                ; THROW AWAY (e)FLAGS
2572 00000CA5 7208                <1>       jc     short _k2d
2573 00000CA7 7505                <1>       jnz    short _k2c
2574 00000CA9 804C240840          <1>       or     byte [esp+8], 01000000b   ; set zero flag bit of eflags register
2575                              <1> _k2c:
2576 00000CAE CF                  <1>       iretd
2577                              <1> _k2d:
2578                              <1>       ; 29/05/2016 -set carry flag on stack-
2579                              <1>       ; [esp] = EIP
2580                              <1>       ; [esp+4] = CS
2581                              <1>       ; [esp+8] = E-FLAGS
2582 00000CAF 804C240801          <1>       or     byte [esp+8], 1  ; set carry bit of eflags register
2583                              <1>       ; [esp+12] = ESP (user)
2584                              <1>       ; [esp+16] = SS (User)
2585 00000CB4 CF                  <1>       iretd
2586                              <1>
2587                              <1>
2588                              <1>       ; (*) 29/05/2016 - 'ref 4' intruction causes to stack fault
2589                              <1>       ; (OUTER-PRIVILEGE-LEVEL)
2590                              <1>       ; INTEL 80386 PROGRAMMER'S REFERENCE MANUAL 1986
2591                              <1>       ; // RETF instruction:
2592                              <1>       ;
2593                              <1>       ; IF OperandMode=32 THEN
2594                              <1>       ;    Load CS:EIP from stack;
2595                              <1>       ;    Set CS RPL to CPL;
2596                              <1>       ;    Increment eSP by 8 plus the immediate offset if it exists;
2597                              <1>       ;    Load SS:eSP from stack;
2598                              <1>       ; ELSE (* OperandMode=16 *)
2599                              <1>       ;    Load CS:IP from stack;
2600                              <1>       ;    Set CS RPL to CPL;
2601                              <1>       ;    Increment eSP by 4 plus the immediate offset if it exists;
2602                              <1>       ;    Load SS:eSP from stack;
2603                              <1>       ; FI;
2604                              <1>       ;
2605                              <1>       ; //
2606                              <1>
2607                              <1>       ;----- SHIFT STATUS
2608                              <1> _K3E:                                 ; GET THE EXTENDED SHIFT STATUS FLAGS
2609 00000CB5 8A25[8E5E0000]      <1>       mov    ah, [KB_FLAG_1]          ; GET SYSTEM SHIFT KEY STATUS
2610 00000CBB 80E404              <1>       and    ah, SYS_SHIFT       ; MASK ALL BUT SYS KEY BIT
2611                              <1>       ;mov   cl, 5               ; SHIFT THEW SYSTEMKEY BIT OVER TO
2612                              <1>       ;shl   ah, cl              ; BIT 7 POSITION
2613 00000CBE C0E405              <1>         shl  ah, 5
2614 00000CC1 A0[8E5E0000]        <1>       mov    al, [KB_FLAG_1]          ; GET SYSTEM SHIFT STATES BACK
2615 00000CC6 2473                <1>       and    al, 01110011b       ; ELIMINATE SYS SHIFT, HOLD_STATE AND INS_SHIFT
2616 00000CC8 08C4                <1>       or     ah, al                 ; MERGE REMAINING BITS INTO AH
2617 00000CCA A0[905E0000]        <1>       mov    al, [KB_FLAG_3]          ; GET RIGHT CTL AND ALT
2618 00000CCF 240C                <1>       and    al, 00001100b       ; ELIMINATE LC_E0 AND LC_E1
2619 00000CD1 08C4                <1>       or     ah, al              ; OR THE SHIFT FLAGS TOGETHER
2620                              <1> _K3:
2621 00000CD3 A0[8D5E0000]        <1>       mov    al, [KB_FLAG]       ; GET THE SHIFT STATUS FLAGS
2622                              <1>       ;jmp   short _KIO_EXIT           ; RETURN TO CALLER
2623 00000CD8 EB83                <1>       jmp    _KIO_EXIT
2624                              <1>
2625                              <1>       ;----- SET TYPAMATIC RATE AND DELAY
2626                              <1> _K300:
2627 00000CDA 3C05                <1>       cmp    al, 5               ; CORRECT FUNCTION CALL?
2628                              <1>       ;jne   short _KIO_EXIT           ; NO, RETURN
2629 00000CDC 0F857BFFFFFF        <1>       jne    _KIO_EXIT
2630 00000CE2 F6C3E0              <1>       test   bl, 0E0h            ; TEST FOR OUT-OF-RANGE RATE
2631 00000CE5 0F8572FFFFFF        <1>         jnz    _KIO_EXIT              ; RETURN IF SO
2632 00000CEB F6C7FC              <1>       test   BH, 0FCh            ; TEST FOR OUT-OF-RANGE DELAY
2633 00000CEE 0F8569FFFFFF        <1>         jnz    _KIO_EXIT              ; RETURN IF SO
2634 00000CF4 B0F3                <1>       mov    al, KB_TYPA_RD           ; COMMAND FOR TYPAMATIC RATE/DELAY
2635 00000CF6 E8DA060000          <1>       call   SND_DATA            ; SEND TO KEYBOARD
2636                              <1>       ;mov   cx, 5               ; SHIFT COUNT
2637                              <1>       ;shl   bh, cl              ; SHIFT DELAY OVER
2638 00000CFB C0E705              <1>       shl    bh, 5
2639 00000CFE 88D8                <1>       mov    al, bl              ; PUT IN RATE
2640 00000D00 08F8                <1>       or     al, bh              ; AND DELAY
2641 00000D02 E8CE060000          <1>       call   SND_DATA            ; SEND TO KEYBOARD
2642 00000D07 E951FFFFFF          <1>         jmp    _KIO_EXIT              ; RETURN TO CALLER
2643                              <1>
2644                              <1>       ;----- WRITE TO KEYBOARD BUFFER
2645                              <1> _K500:
2646 00000D0C 56                  <1>       push   esi                 ; SAVE SI (esi)
2647 00000D0D FA                  <1>       cli                        ;
2648 00000D0E 8B1D[9E5E0000]      <1>       mov    ebx, [BUFFER_TAIL]  ; GET THE 'IN TO' POINTER TO THE BUFFER
2649 00000D14 89DE                <1>       mov    esi, ebx            ; SAVE A COPY IN CASE BUFFER NOT FULL
2650 00000D16 E8D3000000          <1>       call   _K4                 ; BUMP THE POINTER TO SEE IF BUFFER IS FULL
```

```
2651 00000D1B 3B1D[9A5E0000]      <1>       cmp    ebx, [BUFFER_HEAD]  ; WILL THE BUFFER OVERRUN IF WE STORE THIS?
2652 00000D21 740D                <1>       je     short _K502         ; YES - INFORM CALLER OF ERROR
2653 00000D23 66890E              <1>       mov    [esi], cx           ; NO - PUT ASCII/SCAN CODE INTO BUFFER
2654 00000D26 891D[9E5E0000]      <1>       mov    [BUFFER_TAIL], ebx  ; ADJUST 'IN TO' POINTER TO REFLECT CHANGE
2655 00000D2C 28C0                <1>       sub    al, al              ; TELL CALLER THAT OPERATION WAS SUCCESSFUL
2656 00000D2E EB02                <1>       jmp    short _K504         ; SUB INSTRUCTION ALSO RESETS CARRY FLAG
2657                              <1> _K502:
2658 00000D30 B001                <1>       mov    al, 01h                   ; BUFFER FULL INDICATION
2659                              <1> _K504:
2660 00000D32 FB                  <1>       sti
2661 00000D33 5E                  <1>       pop    esi                 ; RECOVER SI (esi)
2662 00000D34 E924FFFFFF          <1>         jmp     _KIO_EXIT               ; RETURN TO CALLER WITH STATUS IN AL
2663                              <1>
2664                              <1>       ;----- READ THE KEY TO FIGURE OUT WHAT TO DO -----
2665                              <1> _K1S:
2666 00000D39 FA                  <1>       cli    ; 03/12/2014
2667 00000D3A 8B1D[9A5E0000]      <1>         mov     ebx, [BUFFER_HEAD]     ; GET POINTER TO HEAD OF BUFFER
2668 00000D40 3B1D[9E5E0000]      <1>         cmp     ebx, [BUFFER_TAIL]     ; TEST END OF BUFFER
2669                              <1>       ;jne   short _K1U          ; IF ANYTHING IN BUFFER SKIP INTERRUPT
2670 00000D46 750F                <1>       jne    short _k1x ; 03/12/2014
2671                              <1>       ;
2672                              <1>       ; 03/12/2014
2673                              <1>       ; 28/08/2014
2674                              <1>       ; PERFORM OTHER FUNCTION ?? here !
2675                              <1>       ;; MOV AX, 9002h           ; MOVE IN WAIT CODE & TYPE
2676                              <1>       ;; INT       15H                ; PERFORM OTHER FUNCTION
2677                              <1> _K1T:                                 ; ASCII READ
2678 00000D48 FB                  <1>       sti                        ; INTERRUPTS BACK ON DURING LOOP
2679 00000D49 90                  <1>       nop                        ; ALLOW AN INTERRUPT TO OCCUR
2680                              <1> _K1U:
2681 00000D4A FA                  <1>       cli                        ; INTERRUPTS BACK OFF
2682 00000D4B 8B1D[9A5E0000]      <1>         mov        ebx, [BUFFER_HEAD]  ; GET POINTER TO HEAD OF BUFFER
2683 00000D51 3B1D[9E5E0000]      <1>         cmp     ebx, [BUFFER_TAIL]     ; TEST END OF BUFFER
2684                              <1> _k1x:
2685 00000D57 53                  <1>       push   ebx                ; SAVE ADDRESS
2686 00000D58 9C                  <1>       pushf                     ; SAVE FLAGS
2687 00000D59 E82F070000          <1>       call   MAKE_LED           ; GO GET MODE INDICATOR DATA BYTE
2688 00000D5E 8A1D[8F5E0000]      <1>       mov    bl, [KB_FLAG_2]    ; GET PREVIOUS BITS
2689 00000D64 30C3                <1>       xor    bl, al             ; SEE IF ANY DIFFERENT
2690 00000D66 80E307              <1>       and    bl, 07h      ; KB_LEDS    ; ISOLATE INDICATOR BITS
2691 00000D69 7406                <1>       jz     short _K1V         ; IF NO CHANGE BYPASS UPDATE
2692 00000D6B E8C9060000          <1>       call   SND_LED1
2693 00000D70 FA                  <1>       cli                       ; DISABLE INTERRUPTS
2694                              <1> _K1V:
2695 00000D71 9D                  <1>       popf                      ; RESTORE FLAGS
2696 00000D72 5B                  <1>       pop    ebx                ; RESTORE ADDRESS
2697 00000D73 74D3                <1>         je      short _K1T             ; LOOP UNTIL SOMETHING IN BUFFER
2698                              <1>       ;
2699 00000D75 668B03              <1>       mov    ax, [ebx]          ; GET SCAN CODE AND ASCII CODE
2700 00000D78 E871000000          <1>         call    _K4                    ; MOVE POINTER TO NEXT POSITION
2701 00000D7D 891D[9A5E0000]      <1>         mov     [BUFFER_HEAD], ebx     ; STORE VALUE IN VARIABLE
2702 00000D83 C3                  <1>       retn                      ; RETURN
2703                              <1>
2704                              <1>       ;----- READ THE KEY TO SEE IF ONE IS PRESENT -----
2705                              <1> _K2S:
2706 00000D84 FA                  <1>       cli                       ; INTERRUPTS OFF
2707 00000D85 8B1D[9A5E0000]      <1>         mov     ebx, [BUFFER_HEAD]     ; GET HEAD POINTER
2708 00000D8B 3B1D[9E5E0000]      <1>         cmp     ebx, [BUFFER_TAIL]     ; IF EQUAL (Z=1) THEN NOTHING THERE
2709 00000D91 668B03              <1>       mov    ax, [ebx]
2710 00000D94 9C                  <1>       pushf                     ; SAVE FLAGS
2711 00000D95 6650                <1>       push   ax                 ; SAVE CODE
2712 00000D97 E8F1060000          <1>       call   MAKE_LED           ; GO GET MODE INDICATOR DATA BYTE
2713 00000D9C 8A1D[8F5E0000]      <1>       mov    bl, [KB_FLAG_2]    ; GET PREVIOUS BITS
2714 00000DA2 30C3                <1>       xor    bl, al             ; SEE IF ANY DIFFERENT
2715 00000DA4 80E307              <1>       and    bl, 07h ; KB_LEDS  ; ISOLATE INDICATOR BITS
2716 00000DA7 7405                <1>       jz     short _K2T         ; IF NO CHANGE BYPASS UPDATE
2717 00000DA9 E874060000          <1>       call   SND_LED                 ; GO TURN ON MODE INDICATORS
2718                              <1> _K2T:
2719 00000DAE 6658                <1>       pop    ax                 ; RESTORE CODE
2720 00000DB0 9D                  <1>       popf                      ; RESTORE FLAGS
2721 00000DB1 FB                  <1>       sti                       ; INTERRUPTS BACK ON
2722 00000DB2 C3                  <1>       retn                      ; RETURN
2723                              <1>
2724                              <1>       ;----- ROUTINE TO TRANSLATE SCAN CODE PAIRS FOR EXTENDED CALLS -----
2725                              <1> _KIO_E_XLAT:
2726 00000DB3 3CF0                <1>       cmp    al, 0F0h           ; IS IT ONE OF THE FILL-INs?
2727 00000DB5 7506                <1>       jne    short _KIO_E_RET   ; NO, PASS IT ON
2728 00000DB7 08E4                <1>         or    ah, ah            ; AH = 0 IS SPECIAL CASE
2729 00000DB9 7402                <1>         jz    short _KIO_E_RET      ; PASS THIS ON UNCHANGED
2730 00000DBB 30C0                <1>       xor    al, al             ; OTHERWISE SET AL = 0
2731                              <1> _KIO_E_RET:
2732 00000DBD C3                  <1>       retn                      ; GO BACK
2733                              <1>
2734                              <1>       ;----- ROUTINE TO TRANSLATE SCAN CODE PAIRS FOR STANDARD CALLS -----
2735                              <1> _KIO_S_XLAT:
2736 00000DBE 80FCE0              <1>       cmp    ah, 0E0h           ; IS IT KEYPAD ENTER OR / ?
2737 00000DC1 750F                <1>       jne    short _KIO_S2      ; NO, CONTINUE
2738 00000DC3 3C0D                <1>       cmp    al, 0Dh                 ; KEYPAD ENTER CODE?
2739 00000DC5 7408                <1>         je    short _KIO_S1      ; YES, MASSAGE A BIT
2740 00000DC7 3C0A                <1>       cmp    al, 0Ah                 ; CTRL KEYPAD ENTER CODE?
2741 00000DC9 7404                <1>         je    short _KIO_S1      ; YES, MASSAGE THE SAME
2742 00000DCB B435                <1>       mov    ah, 35h                 ; NO, MUST BE KEYPAD /
2743                              <1> _kio_ret: ; 03/12/2014
2744 00000DCD F8                  <1>       clc
2745 00000DCE C3                  <1>       retn
2746                              <1>       ;jmp   short _KIO_USE           ; GIVE TO CALLER
2747                              <1> _KIO_S1:
2748 00000DCF B41C                <1>       mov    ah, 1Ch                 ; CONVERT TO COMPATIBLE OUTPUT
2749                              <1>       ;jmp   short _KIO_USE           ; GIVE TO CALLER
2750 00000DD1 C3                  <1>       retn
2751                              <1> _KIO_S2:
2752 00000DD2 80FC84              <1>       cmp    ah, 84h                 ; IS IT ONE OF EXTENDED ONES?
```

```
2753 00000DD5 7715                <1>       ja    short _KIO_DIS            ; YES, THROW AWAY AND GET ANOTHER CHAR
2754 00000DD7 3CF0                <1>       cmp   al, 0F0h            ; IS IT ONE OF THE FILL-INs?
2755 00000DD9 7506                <1>         jne short _KIO_S3       ; NO, TRY LAST TEST
2756 00000DDB 08E4                <1>       or    ah, ah              ; AH = 0 IS SPECIAL CASE
2757 00000DDD 740C                <1>         jz  short _KIO_USE            ; PASS THIS ON UNCHANGED
2758 00000DDF EB0B                <1>       jmp   short _KIO_DIS            ; THROW AWAY THE REST
2759                              <1> _KIO_S3:
2760 00000DE1 3CE0                <1>       cmp   al, 0E0h            ; IS IT AN EXTENSION OF A PREVIOUS ONE?
2761                              <1>       ;jne  short _KIO_USE            ; NO, MUST BE A STANDARD CODE
2762 00000DE3 75E8                <1>       jne   short _kio_ret
2763 00000DE5 08E4                <1>       or    ah, ah              ; AH = 0 IS SPECIAL CASE
2764 00000DE7 7402                <1>         jz  short _KIO_USE            ; JUMP IF AH = 0
2765 00000DE9 30C0                <1>       xor   al, al              ; CONVERT TO COMPATIBLE OUTPUT
2766                              <1>       ;jmp  short _KIO_USE            ; PASS IT ON TO CALLER
2767                              <1> _KIO_USE:
2768                              <1>       ;clc                      ; CLEAR CARRY TO INDICATE GOOD CODE
2769 00000DEB C3                  <1>       retn                      ; RETURN
2770                              <1> _KIO_DIS:
2771 00000DEC F9                  <1>       stc                       ; SET CARRY TO INDICATE DISCARD CODE
2772 00000DED C3                  <1>       retn                      ; RETURN
2773                              <1>
2774                              <1>       ;----- INCREMENT BUFFER POINTER ROUTINE -----
2775                              <1> _K4:
2776 00000DEE 43                  <1>       inc     ebx
2777 00000DEF 43                  <1>       inc   ebx                 ; MOVE TO NEXT WORD IN LIST
2778 00000DF0 3B1D[965E0000]      <1>         cmp     ebx, [BUFFER_END]     ; AT END OF BUFFER?
2779                              <1>         ;jne    short _K5              ; NO, CONTINUE
2780 00000DF6 7206                <1>       jb    short _K5
2781 00000DF8 8B1D[925E0000]      <1>         mov     ebx, [BUFFER_START]     ; YES, RESET TO BUFFER BEGINNING
2782                              <1> _K5:
2783 00000DFE C3                  <1>       retn
2784                              <1>
2785                              <1> ; 20/02/2015
2786                              <1> ; 05/12/2014
2787                              <1> ; 26/08/2014
2788                              <1> ; KEYBOARD (HARDWARE) INTERRUPT -  IRQ LEVEL 1
2789                              <1> ; (INT_09h - Retro UNIX 8086 v1 - U9.ASM, 07/03/2014)
2790                              <1> ;
2791                              <1> ; Derived from "KB_INT_1" procedure of IBM "pc-at"
2792                              <1> ; rombios source code (06/10/1985)
2793                              <1> ; 'keybd.asm', HARDWARE INT 09h - (IRQ Level 1)
2794                              <1>
2795                              <1> ; EQUATES (IBM PC-XT-286 BIOS, 1986, 'POSQEQU.INC')
2796                              <1>
2797                              <1> ;--------- 8042 COMMANDS -------------------------------------------------------
2798                              <1> ENA_KBD           equ   0AEh          ; ENABLE KEYBOARD COMMAND
2799                              <1> DIS_KBD           equ   0ADh          ; DISABLE KEYBOARD COMMAND
2800                              <1> SHUT_CMD    equ   0FEh          ; CAUSE A SHUTDOWN COMMAND
2801                              <1> ;--------- 8042 KEYBOARD INTERFACE AND DIAGNOSTIC CONTROL REGISTERS ------------
2802                              <1> STATUS_PORT equ   064h          ; 8042 STATUS PORT
2803                              <1> INPT_BUF_FULL     equ   00000010b     ; 1 = +INPUT BUFFER FULL
2804                              <1> PORT_A            equ   060h          ; 8042 KEYBOARD SCAN CODE/CONTROL PORT
2805                              <1> ;---------- 8042 KEYBOARD RESPONSE ----------------------------------------------
2806                              <1> KB_ACK            equ   0FAh          ; ACKNOWLEDGE PROM TRANSMISSION
2807                              <1> KB_RESEND   equ   0FEh          ; RESEND REQUEST
2808                              <1> KB_OVER_RUN equ   0FFh          ; OVER RUN SCAN CODE
2809                              <1> ;---------- KEYBOARD/LED COMMANDS -----------------------------------------------
2810                              <1> KB_ENABLE   equ   0F4h          ; KEYBOARD ENABLE
2811                              <1> LED_CMD           equ   0EDh          ; LED WRITE COMMAND
2812                              <1> KB_TYPA_RD  equ   0F3h          ; TYPAMATIC RATE/DELAY COMMAND
2813                              <1> ;---------- KEYBOARD SCAN CODES -------------------------------------------------
2814                              <1> NUM_KEY           equ   69            ; SCAN CODE FOR      NUMBER LOCK KEY
2815                              <1> SCROLL_KEY  equ   70            ; SCAN CODE FOR      SCROLL LOCK KEY
2816                              <1> ALT_KEY           equ   56            ; SCAN CODE FOR      ALTERNATE SHIFT KEY
2817                              <1> CTL_KEY           equ   29            ; SCAN CODE FOR      CONTROL KEY
2818                              <1> CAPS_KEY    equ   58            ; SCAN CODE FOR      SHIFT LOCK KEY
2819                              <1> DEL_KEY           equ   83            ; SCAN CODE FOR      DELETE KEY
2820                              <1> INS_KEY           equ   82            ; SCAN CODE FOR      INSERT KEY
2821                              <1> LEFT_KEY    equ   42            ; SCAN CODE FOR      LEFT SHIFT
2822                              <1> RIGHT_KEY   equ   54            ; SCAN CODE FOR      RIGHT SHIFT
2823                              <1> SYS_KEY           equ   84            ; SCAN CODE FOR      SYSTEM KEY
2824                              <1> ;---------- ENHANCED KEYBOARD SCAN CODES ----------------------------------------
2825                              <1> ID_1        equ   0ABh          ; 1ST ID CHARACTER FOR KBX
2826                              <1> ID_2        equ   041h          ; 2ND ID CHARACTER FOR KBX
2827                              <1> ID_2A       equ   054h          ; ALTERNATE 2ND ID CHARACTER FOR KBX
2828                              <1> F11_M       equ   87            ; F11 KEY MAKE
2829                              <1> F12_M       equ   88            ; F12 KEY MAKE
2830                              <1> MC_E0       equ   224           ; GENERAL MARKER CODE
2831                              <1> MC_E1       equ   225           ; PAUSE KEY MARKER CODE
2832                              <1> ;---------- FLAG EQUATES WITHIN @KB_FLAG-----------------------------------------
2833                              <1> RIGHT_SHIFT equ   00000001b     ; RIGHT SHIFT KEY DEPRESSED
2834                              <1> LEFT_SHIFT  equ   00000010b     ; LEFT SHIFT KEY DEPRESSED
2835                              <1> CTL_SHIFT   equ   00000100b     ; CONTROL SHIFT KEY DEPRESSED
2836                              <1> ALT_SHIFT   equ   00001000b     ; ALTERNATE SHIFT KEY DEPRESSED
2837                              <1> SCROLL_STATE equ  00010000b     ; SCROLL LOCK STATE IS ACTIVE
2838                              <1> NUM_STATE   equ   00100000b     ; NUM LOCK STATE IS ACTIVE
2839                              <1> CAPS_STATE  equ   01000000b     ; CAPS LOCK STATE IS ACTIVE
2840                              <1> INS_STATE   equ   10000000b     ; INSERT STATE IS ACTIVE
2841                              <1> ;---------- FLAG EQUATES WITHIN @KB_FLAG_1 -------------------------------------
2842                              <1> L_CTL_SHIFT equ   00000001b     ; LEFT CTL KEY DOWN
2843                              <1> L_ALT_SHIFT equ   00000010b     ; LEFT ALT KEY DOWN
2844                              <1> SYS_SHIFT   equ   00000100b     ; SYSTEM KEY DEPRESSED AND HELD
2845                              <1> HOLD_STATE  equ   00001000b     ; SUSPEND KEY HAS BEEN TOGGLED
2846                              <1> SCROLL_SHIFT equ  00010000b     ; SCROLL LOCK KEY IS DEPRESSED
2847                              <1> NUM_SHIFT   equ   00100000b     ; NUM LOCK KEY IS DEPRESSED
2848                              <1> CAPS_SHIFT  equ   01000000b     ; CAPS LOCK KEY IS DEPRE55ED
2849                              <1> INS_SHIFT   equ   10000000b     ; INSERT KEY IS DEPRESSED
2850                              <1> ;---------- FLAGS EQUATES WITHIN @KB_FLAG_2 ---------------------------------
2851                              <1> KB_LEDS           equ   00000111b     ; KEYBOARD LED STATE BITS
2852                              <1> ;           equ   00000001b     ; SCROLL LOCK INDICATOR
2853                              <1> ;           equ   00000010b     ; NUM LOCK INDICATOR
2854                              <1> ;           equ   00000100b     ; CAPS LOCK INDICATOR
```

```
2855                            <1> ;           equ    00001000b    ; RESERVED (MUST BE ZERO)
2856                            <1> KB_FA       equ    00010000b    ; ACKNOWLEDGMENT RECEIVED
2857                            <1> KB_FE       equ    00100000b    ; RESEND RECEIVED FLAG
2858                            <1> KB_PR_LED   equ    01000000b    ; MODE INDICATOR UPDATE
2859                            <1> KB_ERR              equ    10000000b    ; KEYBOARD TRANSMIT ERROR FLAG
2860                            <1> ;----------- FLAGS EQUATES WITHIN @KB_FLAG_3 ----------------------------------
2861                            <1> LC_E1       equ    00000001b    ; LAST CODE WAS THE E1 HIDDEN CODE
2862                            <1> LC_E0       equ    00000010b    ; LAST CODE WAS THE E0 HIDDEN CODE
2863                            <1> R_CTL_SHIFT equ    00000100b    ; RIGHT CTL KEY DOWN
2864                            <1> R_ALT_SHIFT equ    00001000b    ; RIGHT ALT KEY DOWN
2865                            <1> GRAPH_ON    equ    00001000b    ; ALT GRAPHICS KEY DOWN (WT ONLY)
2866                            <1> KBX         equ    00010000b    ; ENHANCED KEYBOARD INSTALLED
2867                            <1> SET_NUM_LK  equ    00100000b    ; FORCE NUM LOCK IF READ ID AND KBX
2868                            <1> LC_AB       equ    01000000b    ; LAST CHARACTER WAS FIRST ID CHARACTER
2869                            <1> RD_ID       equ    10000000b    ; DOING A READ ID (MUST BE BIT0)
2870                            <1> ;
2871                            <1> ;----------- INTERRUPT EQUATES -----------------------------------------------
2872                            <1> EOI         equ    020h         ; END OF INTERRUPT COMMAND TO 8259
2873                            <1> INTA00              equ    020h         ; 8259 PORT
2874                            <1>
2875                            <1>
2876                            <1> kb_int:
2877                            <1>
2878                            <1> ; 17/10/2015 ('ctrlbrk')
2879                            <1> ; 05/12/2014
2880                            <1> ; 04/12/2014 (derived from pc-xt-286 bios source code -1986-)
2881                            <1> ; 26/08/2014
2882                            <1> ;
2883                            <1> ; 03/06/86  KEYBOARD BIOS
2884                            <1> ;
2885                            <1> ;--- HARDWARE INT 09H -- (IRQ LEVEL 1) ----------------------------------------
2886                            <1> ;                                                                        ;
2887                            <1> ;     KEYBOARD INTERRUPT ROUTINE                                         ;
2888                            <1> ;                                                                        ;
2889                            <1> ;--------------------------------------------------------------------------------
2890                            <1>
2891                            <1> KB_INT_1:
2892 00000DFF FB                <1>       sti                          ; ENABLE INTERRUPTS
2893                            <1>       ;push  ebp
2894 00000E00 50                <1>       push   eax
2895 00000E01 53                <1>       push   ebx
2896 00000E02 51                <1>       push   ecx
2897 00000E03 52                <1>       push   edx
2898 00000E04 56                <1>       push   esi
2899 00000E05 57                <1>       push   edi
2900 00000E06 1E                <1>       push   ds
2901 00000E07 06                <1>       push   es
2902 00000E08 FC                <1>       cld                          ; FORWARD DIRECTION
2903 00000E09 66B81000          <1>       mov    ax, KDATA
2904 00000E0D 8ED8              <1>       mov    ds, ax
2905 00000E0F 8EC0              <1>       mov    es, ax
2906                            <1>       ;
2907                            <1>       ;-----WAIT FOR KEYBOARD DISABLE COMMAND TO BE ACCEPTED
2908 00000E11 B0AD              <1>       mov    al, DIS_KBD          ; DISABLE THE KEYBOARD COMMAND
2909 00000E13 E8A9050000        <1>       call   SHIP_IT                    ; EXECUTE DISABLE
2910 00000E18 FA                <1>       cli                          ; DISABLE INTERRUPTS
2911 00000E19 B900000100        <1>       mov    ecx, 10000h          ; SET MAXIMUM TIMEOUT
2912                            <1> KB_INT_01:
2913 00000E1E E464              <1>       in     al, STATUS_PORT            ; READ ADAPTER STATUS
2914 00000E20 A802              <1>       test   al, INPT_BUF_FULL    ; CHECK INPUT BUFFER FULL STATUS BIT
2915 00000E22 E0FA              <1>       loopnz KB_INT_01            ; WAIT FOR COMMAND TO BE ACCEPTED
2916                            <1>       ;
2917                            <1>       ;-----READ CHARACTER FROM KEYBOARD INTERFACE
2918 00000E24 E460              <1>       in     al, PORT_A           ; READ IN THE CHARACTER
2919                            <1>       ;
2920                            <1>       ;-----SYSTEM HOOK INT 15H - FUNCTION 4FH (ON HARDWARE INT LEVEL 9H)
2921                            <1>       ;MOV   AH, 04FH             ; SYSTEM INTERCEPT - KEY CODE FUNCTION
2922                            <1>       ;STC                        ; SET CY=1 (IN CASE OF IRET)
2923                            <1>       ;INT   15H                  ; CASETTE CALL (AL)=KEY SCAN CODE
2924                            <1>       ;                           ; RETURNS CY=1 FOR INVALID FUNCTION
2925                            <1>       ;JC    KB_INT_02            ; CONTINUE IF CARRY FLAG SET ((AL)=CODE)
2926                            <1>       ;JMP   K26                  ; EXIT IF SYSTEM HANDLES SCAN CODE
2927                            <1>       ;                           ; EXIT HANDLES HARDWARE EOI AND ENABLE
2928                            <1>       ;
2929                            <1>       ;-----CHECK FOR A RESEND COMMAND TO KEYBOARD
2930                            <1> KB_INT_02:                        ;        (AL)= SCAN CODE
2931 00000E26 FB                <1>       sti                          ; ENABLE INTERRUPTS AGAIN
2932 00000E27 3CFE              <1>       cmp    al, KB_RESEND        ; IS THE INPUT A RESEND
2933 00000E29 7411              <1>         je     short KB_INT_4           ; GO IF RESEND
2934                            <1>       ;
2935                            <1>       ;-----CHECK FOR RESPONSE TO A COMMAND TO KEYBOARD
2936 00000E2B 3CFA              <1>       cmp    al, KB_ACK           ; IS THE INPUT AN ACKNOWLEDGE
2937 00000E2D 751A              <1>         jne    short KB_INT_2           ; GO IF NOT
2938                            <1>       ;
2939                            <1>       ;-----A COMMAND TO THE KEYBOARD WAS ISSUED
2940 00000E2F FA                <1>       cli                          ; DISABLE INTERRUPTS
2941 00000E30 800D[8F5E0000]10  <1>       or     byte [KB_FLAG_2], KB_FA ; INDICATE ACK RECEIVED
2942 00000E37 E97A020000        <1>         jmp    K26                  ; RETURN IF NOT (ACK RETURNED FOR DATA)
2943                            <1>       ;
2944                            <1>       ;-----RESEND THE LAST BYTE
2945                            <1> KB_INT_4:
2946 00000E3C FA                <1>       cli                          ; DISABLE INTERRUPTS
2947 00000E3D 800D[8F5E0000]20  <1>       or     byte [KB_FLAG_2], KB_FE ; INDICATE RESEND RECEIVED
2948 00000E44 E96D020000        <1>         jmp    K26                  ; RETURN IF NOT ACK RETURNED FOR DATA)
2949                            <1>       ;
2950                            <1> ;-----      UPDATE MODE INDICATORS IF CHANGE IN STATE
2951                            <1> KB_INT_2:
2952 00000E49 6650              <1>       push   ax                   ; SAVE DATA IN
2953 00000E4B E83D060000        <1>       call   MAKE_LED             ; GO GET MODE INDICATOR DATA BYTE
2954 00000E50 8A1D[8F5E0000]    <1>       mov    bl, [KB_FLAG_2]      ; GET PREVIOUS BITS
2955 00000E56 30C3              <1>       xor    bl, al               ; SEE IF ANY DIFFERENT
2956 00000E58 80E307            <1>       and    bl, KB_LEDS          ; ISOLATE INDICATOR BITS
```

```
2957 00000E5B 7405               <1>       jz    short UP0           ; IF NO CHANGE BYPASS UPDATE
2958 00000E5D E8C0050000         <1>       call  SND_LED                   ; GO TURN ON MODE INDICATORS
2959                             <1> UP0:
2960 00000E62 6658               <1>       pop   ax                  ; RESTORE DATA IN
2961                             <1> ;--------------------------------------------------------------------
2962                             <1> ;     START OF KEY PROCESSING                                      ;
2963                             <1> ;--------------------------------------------------------------------
2964 00000E64 88C4               <1>       mov   ah, al              ; SAVE SCAN CODE IN AH ALSO
2965                             <1>       ;
2966                             <1>       ;-----TEST FOR OVERRUN SCAN CODE FROM KEYBOARD
2967 00000E66 3CFF               <1>       cmp   al, KB_OVER_RUN           ; IS THIS AN OVERRUN CHAR
2968 00000E68 0F843F050000       <1>         je      K62                   ; BUFFER_FULL_BEEP
2969                             <1>       ;
2970                             <1> K16:
2971 00000E6E 8A3D[905E0000]     <1>       mov   bh, [KB_FLAG_3]           ; LOAD FLAGS FOR TESTING
2972                             <1>       ;
2973                             <1>       ;-----TEST TO SEE IF A READ_ID IS IN PROGRESS
2974 00000E74 F6C7C0             <1>       test  bh, RD_ID+LC_AB     ; ARE WE DOING A READ ID?
2975 00000E77 7449               <1>       jz    short NOT_ID        ; CONTINUE IF NOT
2976 00000E79 7917               <1>       jns   short TST_ID_2            ; IS THE RD_ID FLAG ON?
2977 00000E7B 3CAB               <1>       cmp   al, ID_1            ; IS THIS THE 1ST ID CHARACTER?
2978 00000E7D 7507               <1>       jne   short RST_RD_ID
2979 00000E7F 800D[905E0000]40   <1>       or    byte [KB_FLAG_3], LC_AB ; INDICATE 1ST ID WAS OK
2980                             <1> RST_RD_ID:
2981 00000E86 8025[905E0000]7F   <1>       and   byte [KB_FLAG_3], ~RD_ID ; RESET THE READ ID FLAG
2982                             <1>         ;jmp    short ID_EX           ; AND EXIT
2983 00000E8D E924020000         <1>       jmp   K26
2984                             <1>       ;
2985                             <1> TST_ID_2:
2986 00000E92 8025[905E0000]BF   <1>       and   byte [KB_FLAG_3], ~LC_AB ; RESET FLAG
2987 00000E99 3C54               <1>       cmp   al, ID_2A           ; IS THIS THE 2ND ID CHARACTER?
2988 00000E9B 7419               <1>         je  short KX_BIT        ; JUMP IF SO
2989 00000E9D 3C41               <1>       cmp   al, ID_2            ; IS THIS THE 2ND ID CHARACTER?
2990                             <1>         ;jne short ID_EX        ; LEAVE IF NOT
2991 00000E9F 0F8511020000       <1>       jne   K26
2992                             <1>       ;
2993                             <1>       ;-----A READ ID SAID THAT IT WAS ENHANCED KEYBOARD
2994 00000EA5 F6C720             <1>       test  bh, SET_NUM_LK            ; SHOULD WE SET NUM LOCK?
2995 00000EA8 740C               <1>         jz      short KX_BIT          ; EXIT IF NOT
2996 00000EAA 800D[8D5E0000]20   <1>       or    byte [KB_FLAG], NUM_STATE ; FORCE NUM LOCK ON
2997 00000EB1 E86C050000         <1>       call  SND_LED                   ; GO SET THE NUM LOCK INDICATOR
2998                             <1> KX_BIT:
2999 00000EB6 800D[905E0000]10   <1>       or    byte [KB_FLAG_3], KBX     ; INDICATE ENHANCED KEYBOARD WAS FOUND
3000 00000EBD E9F4010000         <1> ID_EX:      jmp     K26               ; EXIT
3001                             <1>       ;
3002                             <1> NOT_ID:
3003 00000EC2 3CE0               <1>       cmp   al, MC_E0           ; IS THIS THE GENERAL MARKER CODE?
3004 00000EC4 750C               <1>       jne   short TEST_E1
3005 00000EC6 800D[905E0000]12   <1>       or    byte [KB_FLAG_3], LC_E0+KBX ; SET FLAG BIT, SET KBX, AND
3006                             <1>       ;jmp  short EXIT          ; THROW AWAY THIS CODE
3007 00000ECD E9EB010000         <1>       jmp   K26A
3008                             <1> TEST_E1:
3009 00000ED2 3CE1               <1>       cmp   al, MC_E1           ; IS THIS THE PAUSE KEY?
3010 00000ED4 750C               <1>       jne   short NOT_HC
3011 00000ED6 800D[905E0000]11   <1>       or    byte [KB_FLAG_3], LC_E1+KBX ; SET FLAG BIT, SET KBX, AND
3012 00000EDD E9DB010000         <1> EXIT: jmp   K26A                ; THROW AWAY THIS CODE
3013                             <1>       ;
3014                             <1> NOT_HC:
3015 00000EE2 247F               <1>       and   al, 07Fh            ; TURN OFF THE BREAK BIT
3016 00000EE4 F6C702             <1>       test  bh, LC_E0           ; LAST CODE THE E0 MARKER CODE
3017 00000EE7 7414               <1>       jz    short NOT_LC_E0           ; JUMP IF NOT
3018                             <1>       ;
3019 00000EE9 BF[7A5D0000]       <1>       mov   edi, _K6+6          ; IS THIS A SHIFT KEY?
3020 00000EEE AE                 <1>       scasb
3021 00000EEF 0F84C1010000       <1>         je      K26 ; K16B              ; YES, THROW AWAY & RESET FLAG
3022 00000EF5 AE                 <1>       scasb
3023 00000EF6 757C               <1>       jne   short K16A          ; NO, CONTINUE KEY PROCESSING
3024                             <1>       ;jmp  short K16B          ; YES, THROW AWAY & RESET FLAG
3025 00000EF8 E9B9010000         <1>       jmp   K26
3026                             <1>       ;
3027                             <1> NOT_LC_E0:
3028 00000EFD F6C701             <1>       test  bh, LC_E1           ; LAST CODE THE E1 MARKER CODE?
3029 00000F00 7435               <1>       jz    short T_SYS_KEY           ; JUMP IF NOT
3030 00000F02 B904000000         <1>       mov   ecx, 4              ; LENGHT OF SEARCH
3031 00000F07 BF[785D0000]       <1>       mov   edi, _K6+4          ; IS THIS AN ALT, CTL, OR SHIFT?
3032 00000F0C F2AE               <1>       repne scasb               ; CHECK IT
3033                             <1>       ;je   short EXIT          ; THROW AWAY IF SO
3034 00000F0E 0F84A9010000       <1>       je    K26A
3035                             <1>       ;
3036 00000F14 3C45               <1>       cmp   al, NUM_KEY         ; IS IT THE PAUSE KEY?
3037                             <1>       ;jne  short K16B          ; NO, THROW AWAY & RESET FLAG
3038 00000F16 0F859A010000       <1>       jne   K26
3039 00000F1C F6C480             <1>       test  ah, 80h                   ; YES, IS IT THE BREAK OF THE KEY?
3040                             <1>       ;jnz  short K16B          ; YES, THROW THIS AWAY, TOO
3041 00000F1F 0F8591010000       <1>       jnz   K26
3042                             <1>         ; 20/02/2015
3043 00000F25 F605[8E5E0000]08   <1>       test  byte [KB_FLAG_1],HOLD_STATE ;  NO, ARE WE PAUSED ALREADY?
3044                             <1>       ;jnz  short K16B          ;  YES, THROW AWAY
3045 00000F2C 0F8584010000       <1>       jnz   K26
3046 00000F32 E9E1020000         <1>       jmp     K39P                    ; NO, THIS IS THE REAL PAUSE STATE
3047                             <1>       ;
3048                             <1>       ;-----TEST FOR SYSTEM KEY
3049                             <1> T_SYS_KEY:
3050 00000F37 3C54               <1>       cmp   al, SYS_KEY         ; IS IT THE SYSTEM KEY?
3051 00000F39 7539               <1>       jnz   short K16A          ; CONTINUE IF NOT
3052                             <1>       ;
3053 00000F3B F6C480             <1>       test  ah, 80h                   ; CHECK IF THIS A BREAK CODE
3054 00000F3E 7524               <1>       jnz   short K16C          ; DO NOT TOUCH SYSTEM INDICATOR IF TRUE
3055                             <1>       ;
3056 00000F40 F605[8E5E0000]04   <1>       test  byte [KB_FLAG_1], SYS_SHIFT ; SEE IF IN SYSTEM KEY HELD DOWN
3057                             <1>         ;jnz short K16B         ; IF YES, DO NOT PROCESS SYSTEM INDICATOR
3058 00000F47 0F8569010000       <1>       jnz     K26
```

```
3059                                 <1>       ;
3060 00000F4D 800D[8E5E0000]04      <1>       or    byte [KB_FLAG_1], SYS_SHIFT ; INDICATE SYSTEM KEY DEPRESSED
3061 00000F54 B020                  <1>       mov   al, EOI                     ; END OF INTERRUPT COMMAND
3062 00000F56 E620                  <1>       out   20h, al ;out INTA00, al   ; SEND COMMAND TO INTERRUPT CONTROL PORT
3063                                 <1>                                     ; INTERRUPT-RETURN-NO-EOI
3064 00000F58 B0AE                  <1>       mov   al, ENA_KBD         ; INSURE KEYBOARD IS ENABLED
3065 00000F5A E862040000            <1>       call  SHIP_IT                   ; EXECUTE ENABLE
3066                                 <1>       ; !!! SYSREQ !!! function/system call (INTERRUPT) must be here !!!
3067                                 <1>       ;MOV  AL, 8500H           ; FUNCTION VALUE FOR MAKE OF SYSTEM KEY
3068                                 <1>       ;STI                      ; MAKE SURE INTERRUPTS ENABLED
3069                                 <1>       ;INT  15H                 ; USER INTERRUPT
3070 00000F5F E965010000            <1>         jmp    K27A                  ; END PROCESSING
3071                                 <1>       ;
3072                                 <1> ;K16B:      jmp   K26               ; IGNORE SYSTEM KEY
3073                                 <1>       ;
3074                                 <1> K16C:
3075 00000F64 8025[8E5E0000]FB      <1>       and   byte [KB_FLAG_1], ~SYS_SHIFT ; TURN OFF SHIFT KEY HELD DOWN
3076 00000F6B B020                  <1>       mov   al, EOI                     ; END OF INTERRUPT COMMAND
3077 00000F6D E620                  <1>       out   20h, al ;out INTA00, al ; SEND COMMAND TO INTERRUPT CONTROL PORT
3078                                 <1>                                     ; INTERRUPT-RETURN-NO-EOI
3079                                 <1>       ;MOV  AL, ENA_KBD         ; INSURE KEYBOARD IS ENABLED
3080                                 <1>       ;CALL SHIP_IT                   ; EXECUTE ENABLE
3081                                 <1>       ;
3082                                 <1>       ;MOV  AX, 8501H           ; FUNCTION VALUE FOR BREAK OF SYSTEM KEY
3083                                 <1>       ;STI                      ; MAKE SURE INTERRUPTS ENABLED
3084                                 <1>       ;INT  15H                 ; USER INTERRUPT
3085                                 <1>       ;JMP  K27A                ; INGONRE SYSTEM KEY
3086                                 <1>       ;
3087 00000F6F E94E010000            <1>       jmp    K27              ; IGNORE SYSTEM KEY
3088                                 <1>       ;
3089                                 <1>       ;----- TEST FOR SHIFT KEYS
3090                                 <1> K16A:
3091 00000F74 8A1D[8D5E0000]        <1>       mov   bl, [KB_FLAG]       ; PUT STATE FLAGS IN BL
3092 00000F7A BF[745D0000]          <1>       mov   edi, _K6            ; SHIFT KEY TABLE offset
3093 00000F7F B908000000            <1>       mov   ecx, _K6L           ; LENGTH
3094 00000F84 F2AE                  <1>       repne scasb               ; LOOK THROUGH THE TABLE FOR A MATCH
3095 00000F86 88E0                  <1>       mov   al, ah              ; RECOVER SCAN CODE
3096 00000F88 0F8510010000          <1>         jne    K25                    ; IF NO MATCH, THEN SHIFT NOT FOUND
3097                                 <1>       ;
3098                                 <1>       ;------      SHIFT KEY FOUND
3099                                 <1> K17:
3100 00000F8E 81EF[755D0000]        <1>         sub    edi, _K6+1             ; ADJUST PTR TO SCAN CODE MATCH
3101 00000F94 8AA7[7C5D0000]        <1>              mov    ah, [edi+_K7]           ; GET MASK INTO AH
3102 00000F9A B102                  <1>       mov   cl, 2               ; SETUP COUNT FOR FLAG SHIFTS
3103 00000F9C A880                  <1>       test  al, 80h                   ; TEST FOR BREAK KEY
3104 00000F9E 0F8596000000          <1>         jnz    K23                    ; JUMP OF BREAK
3105                                 <1>       ;
3106                                 <1>       ;----- SHIFT MAKE FOUND, DETERMINE SET OR TOGGLE
3107                                 <1> K17C:
3108 00000FA4 80FC10                <1>       cmp   ah, SCROLL_SHIFT
3109 00000FA7 732B                  <1>       jae   short K18           ; IF SCROLL SHIFT OR ABOVE, TOGGLE KEY
3110                                 <1>       ;
3111                                 <1>       ;----- PLAIN SHIFT KEY, SET SHIFT ON
3112 00000FA9 0825[8D5E0000]        <1>       or    [KB_FLAG], ah       ; TURN ON SHIFT BIT
3113 00000FAF A80C                  <1>         test al, CTL_SHIFT+ALT_SHIFT ; IS IT ALT OR CTRL?
3114                                 <1>       ;jnz  short K17D          ; YES, MORE FLAGS TO SET
3115 00000FB1 0F84FF000000          <1>       jz    K26                 ; NO, INTERRUPT RETURN
3116                                 <1> K17D:
3117 00000FB7 F6C702                <1>       test  bh, LC_E0           ; IS THIS ONE OF NEW KEYS?
3118 00000FBA 740B                  <1>       jz    short K17E          ; NO, JUMP
3119 00000FBC 0825[905E0000]        <1>       or    [KB_FLAG_3], ah           ; SET BITS FOR RIGHT CTRL, ALT
3120 00000FC2 E9EF000000            <1>       jmp   K26                 ; INTERRUPT RETURN
3121                                 <1> K17E:
3122 00000FC7 D2EC                  <1>       shr   ah, cl              ; MOVE FLAG BITS TWO POSITIONS
3123 00000FC9 0825[8E5E0000]        <1>       or    [KB_FLAG_1], ah           ; SET BITS FOR LEFT CTRL, ALT
3124 00000FCF E9E2000000            <1>       jmp   K26
3125                                 <1>       ;
3126                                 <1>       ;----- TOGGLED SHIFT KEY, TEST FOR 1ST MAKE OR NOT
3127                                 <1> K18:                            ; SHIFT-TOGGLE
3128 00000FD4 F6C304                <1>       test  bl, CTL_SHIFT             ; CHECK CTL SHIFT STATE
3129                                 <1>         ;jz       short K18A            ; JUMP IF NOT CTL STATE
3130 00000FD7 0F85C1000000          <1>         jnz    K25                    ; JUMP IF CTL STATE
3131                                 <1> K18A:
3132 00000FDD 3C52                  <1>       cmp   al, INS_KEY         ; CHECK FOR INSERT KEY
3133 00000FDF 7524                  <1>       jne   short K22           ; JUMP IF NOT INSERT KEY
3134 00000FE1 F6C308                <1>       test  bl, ALT_SHIFT             ; CHECK FOR ALTERNATE SHIFT
3135                                 <1>             ;jz   short K18B          ; JUMP IF NOT ALTERNATE SHIFT
3136 00000FE4 0F85B4000000          <1>         jnz    K25                    ; JUMP IF ALTERNATE SHIFT
3137                                 <1> K18B:
3138 00000FEA F6C702                <1>       test  bh, LC_E0 ;20/02/2015     ; IS THIS NEW INSERT KEY?
3139 00000FED 7516                  <1>       jnz   short K22           ; YES, THIS ONE'S NEVER A '0'
3140                                 <1> K19:
3141 00000FEF F6C320                <1>       test  bl, NUM_STATE             ; CHECK FOR BASE STATE
3142 00000FF2 750C                  <1>       jnz   short K21           ; JUMP IF NUM LOCK IS ON
3143 00000FF4 F6C303                <1>       test  bl, LEFT_SHIFT+RIGHT_SHIFT ; TEST FOR SHIFT STATE
3144 00000FF7 740C                  <1>       jz    short K22           ; JUMP IF BASE STATE
3145                                 <1> K20:                            ; NUMERIC ZERO, NOT INSERT KEY
3146 00000FF9 88C4                  <1>       mov   ah, al              ; PUT SCAN CODE BACK IN AH
3147 00000FFB E99E000000            <1>         jmp    K25                    ; NUMERAL '0', STNDRD. PROCESSING
3148                                 <1> K21:                            ; MIGHT BE NUMERIC
3149 00001000 F6C303                <1>       test  bl, LEFT_SHIFT+RIGHT_SHIFT
3150 00001003 74F4                  <1>       jz    short K20           ; IS NUMERIC, STD. PROC.
3151                                 <1>       ;
3152                                 <1> K22:                            ; SHIFT TOGGLE KEY HIT; PROCESS IT
3153 00001005 8425[8E5E0000]        <1>       test  ah, [KB_FLAG_1]     ; IS KEY ALREADY DEPRESSED
3154 0000100B 0F85A5000000          <1>         jnz    K26                    ; JUMP IF KEY ALREADY DEPRESSED
3155                                 <1> K22A:
3156 00001011 0825[8E5E0000]        <1>         or     [KB_FLAG_1], ah ; INDICATE THAT THE KEY IS DEPRESSED
3157 00001017 3025[8D5E0000]        <1>       xor   [KB_FLAG], ah       ; TOGGLE THE SHIFT STATE
3158                                 <1>       ;
3159                                 <1>       ;----- TOGGLE LED IF CAPS, NUM  OR SCROLL KEY DEPRESSED
3160 0000101D F6C470                <1>       test  ah, CAPS_SHIFT+NUM_SHIFT+SCROLL_SHIFT ; SHIFT TOGGLE?
```

```
3161 00001020 7409              <1>       jz     short K22B          ; GO IF NOT
3162                            <1>       ;
3163 00001022 6650              <1>       push   ax                  ; SAVE SCAN CODE AND SHIFT MASK
3164 00001024 E8F9030000        <1>       call   SND_LED                   ; GO TURN MODE INDICATORS ON
3165 00001029 6658              <1>       pop    ax                  ; RESTORE SCAN CODE
3166                            <1> K22B:
3167 0000102B 3C52              <1>       cmp    al, INS_KEY         ; TEST FOR 1ST MAKE OF INSERT KEY
3168 0000102D 0F8583000000      <1>         jne     K26                     ; JUMP IF NOT INSERT KEY
3169 00001033 88C4              <1>       mov    ah, al               ; SCAN CODE IN BOTH HALVES OF AX
3170 00001035 E999000000        <1>         jmp     K28                     ; FLAGS UPDATED, PROC. FOR BUFFER
3171                            <1>       ;
3172                            <1>       ;----- BREAK SHIFT FOUND
3173                            <1> K23:                           ; BREAK-SHIFT-FOUND
3174 0000103A 80FC10            <1>       cmp    ah, SCROLL_SHIFT    ; IS THIS A TOGGLE KEY
3175 0000103D F6D4              <1>       not    ah                  ; INVERT MASK
3176 0000103F 7355              <1>       jae    short K24           ; YES, HANDLE BREAK TOGGLE
3177 00001041 2025[8D5E0000]    <1>       and    [KB_FLAG], ah       ; TURN OFF SHIFT BIT
3178 00001047 80FCFB            <1>       cmp    ah, ~CTL_SHIFT           ; IS THIS ALT OR CTL?
3179 0000104A 7730              <1>       ja     short K23D          ; NO, ALL DONE
3180                            <1>       ;
3181 0000104C F6C702            <1>       test   bh, LC_E0           ; 2ND ALT OR CTL?
3182 0000104F 7408              <1>       jz     short K23A          ; NO, HANSLE NORMALLY
3183 00001051 2025[905E0000]    <1>       and    [KB_FLAG_3], ah           ; RESET BIT FOR RIGHT ALT OR CTL
3184 00001057 EB08              <1>       jmp    short K23B          ; CONTINUE
3185                            <1> K23A:
3186 00001059 D2FC              <1>       sar    ah, cl              ; MOVE THE MASK BIT TWO POSITIONS
3187 0000105B 2025[8E5E0000]    <1>       and    [KB_FLAG_1], ah           ; RESET BIT FOR LEFT ALT AND CTL
3188                            <1> K23B:
3189 00001061 88C4              <1>       mov    ah, al              ; SAVE SCAN CODE
3190 00001063 A0[905E0000]      <1>       mov    al, [KB_FLAG_3]           ; GET RIGHT ALT & CTRL FLAGS
3191 00001068 D2E8              <1>       shr    al, cl              ; MOVE TO BITS 1 & 0
3192 0000106A 0A05[8E5E0000]    <1>       or     al, [KB_FLAG_1]           ; PUT IN LEFT ALŞT & CTL FLAGS
3193 00001070 D2E0              <1>       shl    al, cl              ; MOVE BACK TO BITS 3 & 2
3194 00001072 240C              <1>       and    al, ALT_SHIFT+CTL_SHIFT ; FILTER OUT OTHER GARBAGE
3195 00001074 0805[8D5E0000]    <1>       or     [KB_FLAG], al       ; PUT RESULT IN THE REAL FLAGS
3196 0000107A 88E0              <1>       mov    al, ah
3197                            <1> K23D:
3198 0000107C 3CB8              <1>       cmp    al, ALT_KEY+80h           ; IS THIS ALTERNATE SHIFT RELEASE
3199 0000107E 7536              <1>       jne    short K26           ; INTERRUPT RETURN
3200                            <1>       ;
3201                            <1>       ;----- ALTERNATE SHIFT KEY RELEASED, GET THE VALUE INTO BUFFER
3202 00001080 A0[915E0000]      <1>       mov    al, [ALT_INPUT]
3203 00001085 B400              <1>       mov    ah, 0               ; SCAN CODE OF 0
3204 00001087 8825[915E0000]    <1>       mov    [ALT_INPUT], ah     ; ZERO OUT THE FIELD
3205 0000108D 3C00              <1>       cmp    al, 0               ; WAS THE INPUT = 0?
3206 0000108F 7425              <1>       je     short K26           ; INTERRUPT_RETURN
3207                            <1>         ; 29/01/2016
3208                            <1>       ;jmp     K61                   ; IT WASN'T, SO PUT IN BUFFER
3209 00001091 E9D0020000        <1>       jmp    _K60
3210                            <1>       ;
3211                            <1> K24:                           ; BREAK-TOGGLE
3212 00001096 2025[8E5E0000]    <1>       and    [KB_FLAG_1], ah     ; INDICATE NO LONGER DEPRESSED
3213 0000109C EB18              <1>       jmp    short K26           ; INTERRUPT_RETURN
3214                            <1>       ;
3215                            <1>       ;----- TEST FOR HOLD STATE
3216                            <1>                                  ; AL, AH = SCAN CODE
3217                            <1> K25:                           ; NO-SHIFT-FOUND
3218 0000109E 3C80              <1>       cmp    al, 80h                   ; TEST FOR BREAK KEY
3219 000010A0 7314              <1>       jae    short K26           ; NOTHING FOR BREAK CHARS FROM HERE ON
3220 000010A2 F605[8E5E0000]08  <1>       test   byte [KB_FLAG_1], HOLD_STATE ; ARE WE IN HOLD STATE
3221 000010A9 7428              <1>       jz     short K28           ; BRANCH AROUND TEST IF NOT
3222 000010AB 3C45              <1>       cmp    al, NUM_KEY
3223 000010AD 7407              <1>       je     short K26           ; CAN'T END HOLD ON NUM_LOCK
3224 000010AF 8025[8E5E0000]F7  <1>       and    byte [KB_FLAG_1], ~HOLD_STATE ; TURN OFF THE HOLD STATE BIT
3225                            <1>       ;
3226                            <1> K26:
3227 000010B6 8025[905E0000]FC  <1>       and    byte [KB_FLAG_3], ~(LC_E0+LC_E1) ; RESET LAST CHAR H.C. FLAG
3228                            <1> K26A:                          ; INTERRUPT-RETURN
3229 000010BD FA                <1>       cli                        ; TURN OFF INTERRUPTS
3230 000010BE B020              <1>       mov    al, EOI                   ; END OF INTERRUPT COMMAND
3231 000010C0 E620              <1>       out    20h, al      ;out INTA00, al     ; SEND COMMAND TO INTERRUPT CONTROL PORT
3232                            <1> K27:                           ; INTERRUPT-RETURN-NO-EOI
3233 000010C2 B0AE              <1>       mov    al, ENA_KBD         ; INSURE KEYBOARD IS ENABLED
3234 000010C4 E8F8020000        <1>       call   SHIP_IT                   ; EXECUTE ENABLE
3235                            <1> K27A:
3236 000010C9 FA                <1>       cli                        ; DISABLE INTERRUPTS
3237                            <1>       ;;mov  byte [intflg], 0 ; 07/01/2017 ;; 15/01/2017
3238 000010CA 07                <1>       pop    es                  ; RESTORE REGISTERS
3239 000010CB 1F                <1>       pop    ds
3240 000010CC 5F                <1>       pop    edi
3241 000010CD 5E                <1>       pop    esi
3242 000010CE 5A                <1>       pop    edx
3243 000010CF 59                <1>       pop    ecx
3244 000010D0 5B                <1>       pop    ebx
3245 000010D1 58                <1>       pop    eax
3246                            <1>       ;pop   ebp
3247 000010D2 CF                <1>       iretd                      ; RETURN
3248                            <1>
3249                            <1>       ;----- NOT IN HOLD STATE
3250                            <1> K28:                           ; NO-HOLD-STATE
3251 000010D3 3C58              <1>       cmp    al, 88              ; TEST FOR OUT-OF-RANGE SCAN CODES
3252 000010D5 77DF              <1>       ja     short K26           ; IGNORE IF OUT-OF-RANGE
3253                            <1>       ;
3254 000010D7 F6C308            <1>       test   bl, ALT_SHIFT             ; ARE WE IN ALTERNATE SHIFT
3255                            <1>         ;jz  short K28A          ; IF NOT ALTERNATE
3256 000010DA 0F84F1000000      <1>         jz      K38
3257                            <1>       ;
3258 000010E0 F6C710            <1>       test   bh, KBX                   ; IS THIS THE ENCHANCED KEYBOARD?
3259 000010E3 740D              <1>       jz     short K29           ; NO, ALT STATE IS REAL
3260                            <1>        ;28/02/2015
3261 000010E5 F605[8E5E0000]04  <1>       test   byte [KB_FLAG_1], SYS_SHIFT ; YES, IS SYSREQ KEY DOWN?
3262                            <1>       ;jz    short K29           ;  NO, ALT STATE IS REAL
```

```
3263 000010EC 0F85DF000000        <1>       jnz    K38                ; YES, THIS IS PHONY ALT STATE
3264                              <1>         ;                        ; DUE TO PRESSING SYSREQ
3265                              <1> ;K28A:      jmp    short K38
3266                              <1>       ;
3267                              <1>       ;----- TEST FOR RESET KEY SEQUENCE (CTL ALT DEL)
3268                              <1> K29:                             ; TEST-RESET
3269 000010F2 F6C304              <1>       test   bl, CTL_SHIFT             ; ARE WE IN CONTROL SHIFT ALSO?
3270 000010F5 740B                <1>       jz     short K31          ; NO_RESET
3271 000010F7 3C53                <1>       cmp    al, DEL_KEY        ; CTL-ALT STATE, TEST FOR DELETE KEY
3272 000010F9 7507                <1>       jne    short K31          ; NO_RESET, IGNORE
3273                              <1>       ;
3274                              <1>       ;----- CTL-ALT-DEL HAS BEEN FOUND
3275                              <1>       ; 26/08/2014
3276                              <1> cpu_reset:
3277                              <1>       ; IBM PC/AT ROM BIOS source code - 10/06/85 (TEST4.ASM - PROC_SHUTDOWN)
3278                              <1>       ; Send FEh (system reset command) to the keyboard controller.
3279 000010FB B0FE                <1>       mov    al, SHUT_CMD       ; SHUTDOWN COMMAND
3280 000010FD E664                <1>       out    STATUS_PORT, al           ; SEND TO KEYBOARD CONTROL PORT
3281                              <1> khere:
3282 000010FF F4                  <1>       hlt                       ; WAIT FOR 80286 RESET
3283 00001100 EBFD                <1>       jmp    short khere        ; INSURE HALT
3284                              <1>
3285                              <1>       ;
3286                              <1>       ;----- IN ALTERNATE SHIFT, RESET NOT FOUND
3287                              <1> K31:                             ; NO-RESET
3288 00001102 3C39                <1>       cmp    al, 57             ; TEST FOR SPACE KEY
3289 00001104 7507                <1>       jne    short K311         ; NOT THERE
3290 00001106 B020                <1>       mov    al, ' '                   ; SET SPACE CHAR
3291 00001108 E948020000          <1>         jmp     K57                     ; BUFFER_FILL
3292                              <1> K311:
3293 0000110D 3C0F                <1>       cmp    al, 15             ; TEST FOR TAB KEY
3294 0000110F 7509                <1>       jne    short K312         ; NOT THERE
3295 00001111 66B800A5            <1>       mov    ax, 0A500h         ; SET SPECIAL CODE FOR ALT-TAB
3296 00001115 E93B020000          <1>         jmp     K57                     ; BUFFER_FILL
3297                              <1> K312:
3298 0000111A 3C4A                <1>       cmp    al, 74             ; TEST FOR KEY PAD -
3299 0000111C 0F84A2000000        <1>         je      K37B                    ; GO PROCESS
3300 00001122 3C4E                <1>       cmp    al, 78             ; TEST FOR KEY PAD +
3301 00001124 0F849A000000        <1>         je      K37B                    ; GO PROCESS
3302                              <1>       ;
3303                              <1>       ;----- LOOK FOR KEY PAD ENTRY
3304                              <1> K32:                             ; ALT-KEY-PAD
3305 0000112A BF[505D0000]        <1>       mov    edi, K30           ; ALT-INPUT-TABLE offset
3306 0000112F B90A000000          <1>       mov    ecx, 10                  ; LOOK FOR ENTRY USING KEYPAD
3307 00001134 F2AE                <1>       repne  scasb              ; LOOK FOR MATCH
3308 00001136 7525                <1>       jne    short K33          ; NO_ALT_KEYPAD
3309 00001138 F6C702              <1>       test   bh, LC_E0          ; IS THIS ONE OF THE NEW KEYS?
3310 0000113B 0F858A000000        <1>         jnz     K37C                    ; YES, JUMP, NOT NUMPAD KEY
3311 00001141 81EF[515D0000]      <1>       sub    edi, K30+1         ; DI NOW HAS ENTRY VALUE
3312 00001147 A0[915E0000]        <1>       mov    al, [ALT_INPUT]    ; GET THE CURRENT BYTE
3313 0000114C B40A                <1>       mov    ah, 10             ; MULTIPLY BY 10
3314 0000114E F6E4                <1>       mul    ah
3315 00001150 6601F8              <1>       add    ax, di             ; ADD IN THE LATEST ENTRY
3316 00001153 A2[915E0000]        <1>       mov    [ALT_INPUT], al    ; STORE IT AWAY
3317                              <1> ;K32A:
3318 00001158 E959FFFFFF          <1>         jmp     K26                     ; THROW AWAY THAT KEYSTROKE
3319                              <1>       ;
3320                              <1>       ;----- LOOK FOR SUPERSHIFT ENTRY
3321                              <1> K33:                             ; NO-ALT-KEYPAD
3322 0000115D C605[915E0000]00    <1>         mov     byte [ALT_INPUT], 0     ; ZERO ANY PREVIOUS ENTRY INTO INPUT
3323 00001164 B91A000000          <1>       mov    ecx, 26                  ; (DI),(ES) ALREADY POINTING
3324 00001169 F2AE                <1>       repne  scasb              ; LOOK FOR MATCH IN ALPHABET
3325 0000116B 7450                <1>       je     short K37A         ; MATCH FOUND, GO FILLL THE BUFFER
3326                              <1>       ;
3327                              <1>       ;----- LOOK FOR TOP ROW OF ALTERNATE SHIFT
3328                              <1> K34:                             ; ALT-TOP-ROW
3329 0000116D 3C02                <1>       cmp    al, 2              ; KEY WITH '1' ON IT
3330 0000116F 7253                <1>       jb     short K37B         ; MUST BE ESCAPE
3331 00001171 3C0D                <1>       cmp    al, 13             ; IS IT IN THE REGION
3332 00001173 7705                <1>       ja     short K35          ; NO, ALT SOMETHING ELSE
3333 00001175 80C476              <1>       add    ah, 118                  ; CONVERT PSEUDO SCAN CODE TO RANGE
3334 00001178 EB43                <1>       jmp    short K37A         ; GO FILL THE BUFFER
3335                              <1>       ;
3336                              <1>       ;----- TRANSLATE ALTERNATE SHIFT PSEUDO SCAN CODES
3337                              <1> K35:                             ; ALT-FUNCTION
3338 0000117A 3C57                <1>       cmp    al, F11_M          ; IS IT F11?
3339 0000117C 7209                <1>       jb     short K35A ; 20/02/2015   ; NO, BRANCH
3340 0000117E 3C58                <1>       cmp    al, F12_M          ; IS IT F12?
3341 00001180 7705                <1>       ja     short K35A ; 20/02/2015   ; NO, BRANCH
3342 00001182 80C434              <1>       add    ah, 52             ; CONVERT TO PSEUDO SCAN CODE
3343 00001185 EB36                <1>       jmp    short K37A         ; GO FILL THE BUFFER
3344                              <1> K35A:
3345 00001187 F6C702              <1>       test   bh, LC_E0          ; DO WE HAVE ONE OF THE NEW KEYS?
3346 0000118A 7422                <1>       jz     short K37          ; NO, JUMP
3347 0000118C 3C1C                <1>       cmp    al, 28             ; TEST FOR KEYPAD ENTER
3348 0000118E 7509                <1>         jne     short K35B              ; NOT THERE
3349 00001190 66B800A6            <1>       mov    ax, 0A600h         ; SPECIAL CODE
3350 00001194 E9BC010000          <1>       jmp    K57                ; BUFFER FILL
3351                              <1> K35B:
3352 00001199 3C53                <1>       cmp    al, 83             ; TEST FOR DELETE KEY
3353 0000119B 742E                <1>       je     short K37C         ; HANDLE WITH OTHER EDIT KEYS
3354 0000119D 3C35                <1>       cmp    al, 53             ; TEST FOR KEYPAD /
3355                              <1>       ;jne   short K32A         ; NOT THERE, NO OTHER E0 SPECIALS
3356 0000119F 0F8511FFFFFF        <1>         jne     K26
3357 000011A5 66B800A4            <1>       mov    ax, 0A400h         ; SPECIAL CODE
3358 000011A9 E9A7010000          <1>       jmp    K57                ; BUFFER FILL
3359                              <1> K37:
3360 000011AE 3C3B                <1>       cmp    al, 59             ; TEST FOR FUNCTION KEYS (F1)
3361 000011B0 7212                <1>         jb      short K37B              ; NO FN, HANDLE W/OTHER EXTENDED
3362 000011B2 3C44                <1>       cmp    al, 68             ; IN KEYPAD REGION?
3363                              <1>         ;ja   short K32A         ; IF SO, IGNORE
3364 000011B4 0F87FCFEFFFF        <1>         ja      K26
```

```
3365 000011BA 80C42D              <1>       add    ah, 45              ; CONVERT TO PSEUDO SCAN CODE
3366                              <1> K37A:
3367 000011BD B000                <1>       mov    al, 0               ; ASCII CODE OF ZERO
3368 000011BF E991010000          <1>         jmp     K57                     ; PUT IT IN THE BUFFER
3369                              <1> K37B:
3370 000011C4 B0F0                <1>       mov    al, 0F0h            ; USE SPECIAL ASCII CODE
3371 000011C6 E98A010000          <1>       jmp     K57                   ; PUT IT IN THE BUFFER
3372                              <1> K37C:
3373 000011CB 0450                <1>       add    al, 80              ; CONVERT SCAN CODE (EDIT KEYS)
3374 000011CD 88C4                <1>       mov    ah, al              ; (SCAN CODE NOT IN AH FOR INSERT)
3375 000011CF EBEC                <1>       jmp     short K37A             ; PUT IT IN THE BUFFER
3376                              <1>       ;
3377                              <1>       ;----- NOT IN ALTERNATE SHIFT
3378                              <1> K38:                              ; NOT-ALT-SHIFT
3379                              <1>                                   ; BL STILL HAS SHIFT FLAGS
3380 000011D1 F6C304              <1>       test   bl, CTL_SHIFT              ; ARE WE IN CONTROL SHIFT?
3381                              <1>       ;jnz   short K38A           ; YES, START PROCESSING
3382 000011D4 0F84B0000000        <1>         jz      K44                     ; NOT-CTL-SHIFT
3383                              <1>       ;
3384                              <1>       ;----- CONTROL SHIFT, TEST SPECIAL CHARACTERS
3385                              <1>       ;----- TEST FOR BREAK
3386                              <1> K38A:
3387 000011DA 3C46                <1>       cmp    al, SCROLL_KEY             ; TEST FOR BREAK
3388 000011DC 7531                <1>       jne    short K39           ; JUMP, NO-BREAK
3389 000011DE F6C710              <1>       test   bh, KBX                    ; IS THIS THE ENHANCED KEYBOARD?
3390 000011E1 7405                <1>       jz     short K38B          ; NO, BREAK IS VALID
3391 000011E3 F6C702              <1>       test   bh, LC_E0           ; YES, WAS LAST CODE AN E0?
3392 000011E6 7427                <1>       jz     short K39           ; NO-BREAK, TEST FOR PAUSE
3393                              <1> K38B:
3394 000011E8 8B1D[9A5E0000]      <1>       mov    ebx, [BUFFER_HEAD]  ; RESET BUFFER TO EMPTY
3395 000011EE 891D[9E5E0000]      <1>       mov    [BUFFER_TAIL], ebx
3396 000011F4 C605[8C5E0000]80    <1>       mov    byte [BIOS_BREAK], 80h  ; TURN ON BIOS_BREAK BIT
3397                              <1>       ;
3398                              <1>       ;----- ENABLE KEYBOARD
3399 000011FB B0AE                <1>       mov    al, ENA_KBD         ; ENABLE KEYBOARD
3400 000011FD E8BF010000          <1>       call   SHIP_IT                    ; EXECUTE ENABLE
3401                              <1>       ;
3402                              <1>       ; CTRL+BREAK code here !!!
3403                              <1>       ;INT   1BH                 ; BREAK INTERRUPT VECTOR
3404                              <1>       ; 17/10/2015
3405 00001202 E8CF510000          <1>       call   ctrlbrk ; control+break subroutine
3406                              <1>       ;
3407 00001207 6629C0              <1>       sub    ax, ax              ; PUT OUT DUMMY CHARACTER
3408 0000120A E946010000          <1>         jmp     K57                     ; BUFFER_FILL
3409                              <1>       ;
3410                              <1>       ;----- TEST FOR PAUSE
3411                              <1> K39:                              ; NO_BREAK
3412 0000120F F6C710              <1>       test   bh, KBX                    ; IS THIS THE ENHANCED KEYBOARD?
3413 00001212 7537                <1>       jnz    short K41           ; YES, THEN THIS CAN'T BE PAUSE
3414 00001214 3C45                <1>       cmp    al, NUM_KEY         ; LOOK FOR PAUSE KEY
3415 00001216 7533                <1>       jne    short K41           ; NO-PAUSE
3416                              <1> K39P:
3417 00001218 800D[8E5E0000]08    <1>       or     byte [KB_FLAG_1], HOLD_STATE ; TURN ON THE HOLD FLAG
3418                              <1>       ;
3419                              <1>       ;----- ENABLE KEYBOARD
3420 0000121F B0AE                <1>       mov    al, ENA_KBD         ; ENABLE KEYBOARD
3421 00001221 E89B010000          <1>       call   SHIP_IT                    ; EXECUTE ENABLE
3422                              <1> K39A:
3423 00001226 B020                <1>       mov    al, EOI                    ; END OF INTERRUPT TO CONTROL PORT
3424 00001228 E620                <1>       out    20h, al ;out INTA00, al    ; ALLOW FURTHER KEYSTROKE INTERRUPTS
3425                              <1>       ;
3426                              <1>       ;----- DURING PAUSE INTERVAL, TURN COLOR CRT BACK ON
3427 0000122A 803D[C25E0000]07    <1>         cmp     byte [CRT_MODE], 7      ; IS THIS BLACK AND WHITE CARD
3428 00001231 740A                <1>         je      short K40                   ; YES, NOTHING TO DO
3429 00001233 66BAD803            <1>       mov    dx, 03D8h           ; PORT FOR COLOR CARD
3430 00001237 A0[C35E0000]        <1>         mov     al, [CRT_MODE_SET]     ; GET THE VALUE OF THE CURRENT MODE
3431 0000123C EE                  <1>       out    dx, al              ; SET THE CRT MODE, SO THAT CRT IS ON
3432                              <1>       ;
3433                              <1> K40:                              ; PAUSE-LOOP
3434 0000123D F605[8E5E0000]08    <1>         test    byte [KB_FLAG_1], HOLD_STATE ; CHECK HOLD STATE FLAG
3435 00001244 75F7                <1>       jnz    short K40           ; LOOP UNTIL FLAG TURNED OFF
3436                              <1>       ;
3437 00001246 E977FEFFFF          <1>         jmp     K27                     ; INTERRUPT_RETURN_NO_EOI
3438                              <1>         ;
3439                              <1>       ;----- TEST SPECIAL CASE KEY 55
3440                              <1> K41:                              ; NO-PAUSE
3441 0000124B 3C37                <1>       cmp    al, 55              ; TEST FOR */PRTSC KEY
3442 0000124D 7513                <1>       jne    short K42           ; NOT-KEY-55
3443 0000124F F6C710              <1>       test   bh, KBX                    ; IS THIS THE ENHANCED KEYBOARD?
3444 00001252 7405                <1>       jz     short K41A          ; NO, CTL-PRTSC IS VALID
3445 00001254 F6C702              <1>       test   bh, LC_E0           ; YES, WAS LAST CODE AN E0?
3446 00001257 7421                <1>       jz     short K42B          ; NO, TRANSLATE TO A FUNCTION
3447                              <1> K41A:
3448 00001259 66B80072            <1>       mov    ax, 114*256         ; START/STOP PRINTING SWITCH
3449 0000125D E9F3000000          <1>         jmp     K57                     ; BUFFER_FILL
3450                              <1>       ;
3451                              <1>       ;----- SET UP TO TRANSLATE CONTROL SHIFT
3452                              <1> K42:                              ; NOT-KEY-55
3453 00001262 3C0F                <1>       cmp    al, 15              ; IS IT THE TAB KEY?
3454 00001264 7414                <1>       je     short K42B          ; YES, XLATE TO FUNCTION CODE
3455 00001266 3C35                <1>       cmp    al, 53              ; IS IT THE / KEY?
3456 00001268 750E                <1>       jne    short K42A          ; NO, NO MORE SPECIAL CASES
3457 0000126A F6C702              <1>       test   bh, LC_E0           ; YES, IS IT FROM THE KEY PAD?
3458 0000126D 7409                <1>       jz     short K42A          ; NO, JUST TRANSLATE
3459 0000126F 66B80095            <1>       mov    ax, 9500h           ; YES, SPECIAL CODE FOR THIS ONE
3460 00001273 E9DD000000          <1>       jmp    K57                 ; BUFFER FILL
3461                              <1> K42A:
3462                              <1>       ;;mov  ebx, _K8            ; SET UP TO TRANSLATE CTL
3463 00001278 3C3B                <1>       cmp    al, 59              ; IS IT IN CHARACTER TABLE?
3464                              <1>         ;jb  short K45F              ; YES, GO TRANSLATE CHAR
3465                              <1>       ;;jb   K56 ; 20/02/2015
3466                              <1>       ;;jmp  K64 ; 20/02/2015
```

```
3467                                 <1> K42B:
3468 0000127A BB[845D0000]           <1>       mov    ebx, _K8            ; SET UP TO TRANSLATE CTL
3469 0000127F 0F82AE000000           <1>       jb     K56 ;; 20/02/2015
3470 00001285 E9B9000000             <1>       jmp    K64
3471                                 <1>         ;
3472                                 <1>       ;----- NOT IN CONTROL SHIFT
3473                                 <1> K44:                              ; NOT-CTL-SHIFT
3474 0000128A 3C37                   <1>       cmp    al, 55               ; PRINT SCREEN KEY?
3475 0000128C 7528                   <1>       jne    short K45            ; NOT PRINT SCREEN
3476 0000128E F6C710                 <1>       test   bh, KBX                    ; IS THIS ENHANCED KEYBOARD?
3477 00001291 7407                   <1>       jz     short K44A           ; NO, TEST FOR SHIFT STATE
3478 00001293 F6C702                 <1>       test   bh, LC_E0            ; YES, LAST CODE A MARKER?
3479 00001296 7507                   <1>       jnz    short K44B           ; YES, IS PRINT SCREEN
3480 00001298 EB41                   <1>       jmp    short K45C           ; NO, TRANSLATE TO '*' CHARACTER
3481                                 <1> K44A:
3482 0000129A F6C303                 <1>       test   bl, LEFT_SHIFT+RIGHT_SHIFT ; NOT 101 KBD, SHIFT KEY DOWN?
3483 0000129D 743C                   <1>       jz     short K45C           ; NO, TRANSLATE TO '*' CHARACTER
3484                                 <1>       ;
3485                                 <1>       ;----- ISSUE INTERRUPT TO INDICATE PRINT SCREEN FUNCTION
3486                                 <1> K44B:
3487 0000129F B0AE                   <1>       mov    al, ENA_KBD          ; INSURE KEYBOARD IS ENABLED
3488 000012A1 E81B010000             <1>       call   SHIP_IT                    ; EXECUTE ENABLE
3489 000012A6 B020                   <1>       mov    al, EOI                    ; END OF CURRENT INTERRUPT
3490 000012A8 E620                   <1>       out    20h, al ;out INTA00, al    ; SO FURTHER THINGS CAN HAPPEN
3491                                 <1>       ; Print Screen !!!           ; ISSUE PRINT SCREEN INTERRUPT (INT 05h)
3492                                 <1>       ;PUSH  BP                    ; SAVE POINTER
3493                                 <1>       ;INT   5H                    ; ISSUE PRINT SCREEN INTERRUPT
3494                                 <1>       ;POP   BP                    ; RESTORE POINTER
3495 000012AA 8025[905E0000]FC       <1>         and     byte [KB_FLAG_3], ~(LC_E0+LC_E1) ; ZERO OUT THESE FLAGS
3496 000012B1 E90CFEFFFF             <1>         jmp     K27                      ; GO BACK WITHOUT EOI OCCURRING
3497                                 <1>       ;
3498                                 <1>       ;----- HANDLE IN-CORE KEYS
3499                                 <1> K45:                              ; NOT-PRINT-SCREEN
3500 000012B6 3C3A                   <1>       cmp    al, 58               ; TEST FOR IN-CORE AREA
3501 000012B8 7734                   <1>       ja     short K46            ; JUMP IF NOT
3502 000012BA 3C35                   <1>       cmp    al, 53               ; IS THIS THE '/' KEY?
3503 000012BC 7505                   <1>       jne    short K45A           ; NO, JUMP
3504 000012BE F6C702                 <1>       test   bh, LC_E0            ; WAS THE LAST CODE THE MARKER?
3505 000012C1 7518                   <1>       jnz    short K45C           ; YES, TRANSLATE TO CHARACTER
3506                                 <1> K45A:
3507 000012C3 B91A000000             <1>       mov    ecx, 26                    ; LENGHT OF SEARCH
3508 000012C8 BF[5A5D0000]           <1>       mov    edi, K30+10          ; POINT TO TABLE OF A-Z CHARS
3509 000012CD F2AE                   <1>       repne  scasb                ; IS THIS A LETTER KEY?
3510                                 <1>              ; 20/02/2015
3511 000012CF 7505                   <1>       jne    short K45B              ; NO, SYMBOL KEY
3512                                 <1>       ;
3513 000012D1 F6C340                 <1>       test   bl, CAPS_STATE             ; ARE WE IN CAPS_LOCK?
3514 000012D4 750C                   <1>       jnz    short K45D           ; TEST FOR SURE
3515                                 <1> K45B:
3516 000012D6 F6C303                 <1>       test   bl, LEFT_SHIFT+RIGHT_SHIFT ; ARE WE IN SHIFT STATE?
3517 000012D9 750C                   <1>       jnz    short K45E           ; YES, UPPERCASE
3518                                 <1>                                   ; NO, LOWERCASE
3519                                 <1> K45C:
3520 000012DB BB[DC5D0000]           <1>       mov    ebx, K10             ; TRANSLATE TO LOWERCASE LETTERS
3521 000012E0 EB51                   <1>       jmp    short K56
3522                                 <1> K45D:                             ; ALMOST-CAPS-STATE
3523 000012E2 F6C303                 <1>       test   bl, LEFT_SHIFT+RIGHT_SHIFT ; CL ON. IS SHIFT ON, TOO?
3524 000012E5 75F4                   <1>       jnz    short K45C           ; SHIFTED TEMP OUT OF CAPS STATE
3525                                 <1> K45E:
3526 000012E7 BB[345E0000]           <1>       mov    ebx, K11             ; TRANSLATE TO UPPER CASE LETTERS
3527 000012EC EB45                   <1> K45F: jmp    short K56
3528                                 <1>       ;
3529                                 <1>       ;----- TEST FOR KEYS F1 - F10
3530                                 <1> K46:                              ; NOT IN-CORE AREA
3531 000012EE 3C44                   <1>       cmp    al, 68               ; TEST FOR F1 - F10
3532                                 <1>       ;ja    short K47            ; JUMP IF NOT
3533                                 <1>       ;jmp   short K53            ; YES, GO DO FN KEY PROCESS
3534 000012F0 7635                   <1>       jna    short K53
3535                                 <1>       ;
3536                                 <1>       ;----- HANDLE THE NUMERIC PAD KEYS
3537                                 <1> K47:                              ; NOT F1 - F10
3538 000012F2 3C53                   <1>       cmp    al, 83               ; TEST NUMPAD KEYS
3539 000012F4 772D                   <1>       ja     short K52            ; JUMP IF NOT
3540                                 <1>       ;
3541                                 <1>       ;----- KEYPAD KEYS, MUST TEST NUM LOCK FOR DETERMINATION
3542                                 <1> K48:
3543 000012F6 3C4A                   <1>       cmp    al , 74                    ; SPECIAL CASE FOR MINUS
3544 000012F8 74ED                   <1>       je     short K45E           ; GO TRANSLATE
3545 000012FA 3C4E                   <1>       cmp    al , 78                    ; SPECIAL CASE FOR PLUS
3546 000012FC 74E9                   <1>       je     short K45E           ; GO TRANSLATE
3547 000012FE F6C702                 <1>       test   bh, LC_E0            ; IS THIS ONE OFTHE NEW KEYS?
3548 00001301 750A                   <1>       jnz    short K49            ; YES, TRANSLATE TO BASE STATE
3549                                 <1>       ;
3550 00001303 F6C320                 <1>       test   bl, NUM_STATE        ; ARE WE IN NUM LOCK
3551 00001306 7514                   <1>       jnz    short K50            ; TEST FOR SURE
3552 00001308 F6C303                 <1>       test   bl, LEFT_SHIFT+RIGHT_SHIFT ; ARE WE IN SHIFT STATE?
3553                                 <1>       ;jnz   short K51            ; IF SHIFTED, REALLY NUM STATE
3554 0000130B 75DA                   <1>       jnz    short K45E
3555                                 <1>       ;
3556                                 <1>       ;----- BASE CASE FOR KEYPAD
3557                                 <1> K49:
3558 0000130D 3C4C                   <1>       cmp    al, 76               ; SPECIAL CASE FOR BASE STATE 5
3559 0000130F 7504                   <1>       jne    short K49A           ; CONTINUE IF NOT KEYPAD 5
3560 00001311 B0F0                   <1>       mov    al, 0F0h             ; SPECIAL ASCII CODE
3561 00001313 EB40                   <1>       jmp    short K57            ; BUFFER FILL
3562                                 <1> K49A:
3563 00001315 BB[DC5D0000]           <1>       mov    ebx, K10             ; BASE CASE TABLE
3564 0000131A EB27                   <1>       jmp    short K64            ; CONVERT TO PSEUDO SCAN
3565                                 <1>       ;
3566                                 <1>       ;----- MIGHT BE NUM LOCK, TEST SHIFT STATUS
3567                                 <1> K50:                              ; ALMOST-NUM-STATE
3568 0000131C F6C303                 <1>         test    bl, LEFT_SHIFT+RIGHT_SHIFT
```

```
3569 0000131F 75EC                <1>       jnz    short K49           ; SHIFTED TEMP OUT OF NUM STATE
3570 00001321 EBC4                <1> K51:  jmp    short K45E          ; REALLY NUM STATE
3571                              <1>       ;
3572                              <1>       ;----- TEST FOR THE NEW KEYS ON WT KEYBOARDS
3573                              <1> K52:                               ; NOT A NUMPAD KEY
3574 00001323 3C56                <1>       cmp    al, 86              ; IS IT THE NEW WT KEY?
3575                              <1>       ;jne   short K53           ; JUMP IF NOT
3576                              <1>       ;jmp   short K45B          ; HANDLE WITH REST OF LETTER KEYS
3577 00001325 74AF                <1>       je     short K45B
3578                              <1>       ;
3579                              <1>       ;----- MUST BE F11 OR F12
3580                              <1> K53:                               ; F1 - F10 COME HERE, TOO
3581 00001327 F6C303              <1>       test   bl, LEFT_SHIFT+RIGHT_SHIFT ; TEST SHIFT STATE
3582 0000132A 74E1                <1>       jz     short K49           ; JUMP, LOWER CASE PSEUDO SC'S
3583                              <1>              ; 20/02/2015
3584 0000132C BB[345E0000]        <1>       mov    ebx, K11            ; UPPER CASE PSEUDO SCAN CODES
3585 00001331 EB10                <1>       jmp    short K64           ; TRANSLATE SCAN
3586                              <1>       ;
3587                              <1>       ;----- TRANSLATE THE CHARACTER
3588                              <1> K56:                               ; TRANSLATE-CHAR
3589 00001333 FEC8                <1>       dec    al                  ; CONVERT ORIGIN
3590 00001335 D7                  <1>       xlat                       ; CONVERT THE SCAN CODE TO ASCII
3591 00001336 F605[905E0000]02    <1>       test   byte [KB_FLAG_3], LC_E0   ; IS THIS A NEW KEY?
3592 0000133D 7416                <1>       jz     short K57           ; NO, GO FILL BUFFER
3593 0000133F B4E0                <1>       mov    ah, MC_E0           ; YES, PUT SPECIAL MARKER IN AH
3594 00001341 EB12                <1>       jmp    short K57           ; PUT IT INTO THE BUFFER
3595                              <1>       ;
3596                              <1>       ;----- TRANSLATE SCAN FOR PSEUDO SCAN CODES
3597                              <1> K64:                               ; TRANSLATE-SCAN-ORGD
3598 00001343 FEC8                <1>       dec    al                  ; CONVERT ORIGIN
3599 00001345 D7                  <1>              xlat                         ; CTL TABLE SCAN
3600 00001346 88C4                <1>       mov    ah, al              ; PUT VALUE INTO AH
3601 00001348 B000                <1>       mov    al, 0               ; ZERO ASCII CODE
3602 0000134A F605[905E0000]02    <1>       test   byte [KB_FLAG_3], LC_E0   ; IS THIS A NEW KEY?
3603 00001351 7402                <1>       jz     short K57           ; NO, GO FILL BUFFER
3604 00001353 B0E0                <1>       mov    al, MC_E0           ; YES, PUT SPECIAL MARKER IN AL
3605                              <1>       ;
3606                              <1>       ;----- PUT CHARACTER INTO BUFFER
3607                              <1> K57:                               ; BUFFER_FILL
3608 00001355 3CFF                <1>       cmp    al, -1              ; IS THIS AN IGNORE CHAR
3609                              <1>         ;je  short K59           ; YES, DO NOTHING WITH IT
3610 00001357 0F8459FDFFFF        <1>       je      K26                ; YES, DO NOTHING WITH IT
3611 0000135D 80FCFF              <1>       cmp    ah, -1              ; LOOK FOR -1 PSEUDO SCAN
3612                              <1>         ;jne short K61           ; NEAR_INTERRUPT_RETURN
3613 00001360 0F8450FDFFFF        <1>       je      K26                ; INTERRUPT_RETURN
3614                              <1> ;K59:                              ; NEAR_INTERRUPT_RETURN
3615                              <1> ;     jmp    K26                 ; INTERRUPT_RETURN
3616                              <1>
3617                              <1> _K60: ; 29/01/2016
3618 00001366 80FC68              <1>       cmp    ah, 68h      ; ALT + F1 key
3619 00001369 721F                <1>       jb     short K61
3620 0000136B 80FC6F              <1>       cmp    ah, 6Fh ; ALT + F8 key
3621 0000136E 771A                <1>       ja     short K61
3622                              <1>       ;
3623 00001370 8A1D[F64D0100]      <1>       mov    bl, [ACTIVE_PAGE]
3624 00001376 80C368              <1>       add    bl, 68h
3625 00001379 38E3                <1>       cmp    bl, ah
3626 0000137B 740D                <1>       je     short K61
3627 0000137D 6650                <1>       push   ax
3628 0000137F 88E0                <1>       mov    al, ah
3629 00001381 2C68                <1>       sub    al, 68h
3630 00001383 E8F4050000          <1>       call   set_active_page
3631 00001388 6658                <1>       pop    ax
3632                              <1> K61:                               ; NOT-CAPS-STATE
3633 0000138A 8B1D[9E5E0000]      <1>       mov    ebx, [BUFFER_TAIL]  ; GET THE END POINTER TO THE BUFFER
3634 00001390 89DE                <1>       mov    esi, ebx            ; SAVE THE VALUE
3635 00001392 E857FAFFFF          <1>       call   _K4                 ; ADVANCE THE TAIL
3636 00001397 3B1D[9A5E0000]      <1>       cmp    ebx, [BUFFER_HEAD]  ; HAS THE BUFFER WRAPPED AROUND
3637 0000139D 740E                <1>       je     short K62           ; BUFFER_FULL_BEEP
3638 0000139F 668906              <1>       mov    [esi], ax           ; STORE THE VALUE
3639 000013A2 891D[9E5E0000]      <1>       mov    [BUFFER_TAIL], ebx  ; MOVE THE POINTER UP
3640 000013A8 E909FDFFFF          <1>       jmp    K26
3641                              <1>       ;;cli                      ; TURN OFF INTERRUPTS
3642                              <1>       ;;mov  al, EOI                   ; END OF INTERRUPT COMMAND
3643                              <1>       ;;out  INTA00, al          ; SEND COMMAND TO INTERRUPT CONTROL PORT
3644                              <1>       ;MOV   AL, ENA_KBD         ; INSURE KEYBOARD IS ENABLED
3645                              <1>       ;CALL  SHIP_IT                   ; EXECUTE ENABLE
3646                              <1>       ;MOV   AX, 9102H           ; MOVE IN POST CODE & TYPE
3647                              <1>       ;INT   15H                 ; PERFORM OTHER FUNCTION
3648                              <1>       ;;and  byte [KB_FLAG_3],~(LC_E0+LC_E1) ; RESET LAST CHAR H.C. FLAG
3649                              <1>       ;JMP   K27A                ; INTERRUPT_RETURN
3650                              <1>       ;;jmp   K27
3651                              <1>       ;
3652                              <1>       ;----- BUFFER IS FULL SOUND THE BEEPER
3653                              <1> K62:
3654 000013AD B020                <1>       mov    al, EOI                   ; ENABLE INTERRUPT CONTROLLER CHIP
3655 000013AF E620                <1>       out    INTA00, al
3656 000013B1 66B9A602            <1>       mov    cx, 678                   ; DIVISOR FOR 1760 HZ
3657 000013B5 B304                <1>       mov    bl, 4               ; SHORT BEEP COUNT (1/16 + 1/64 DELAY)
3658 000013B7 E8E5090000          <1>       call   beep                ; GO TO COMMON BEEP HANDLER
3659 000013BC E901FDFFFF          <1>       jmp     K27                ; EXIT
3660                              <1>
3661                              <1> SHIP_IT:
3662                              <1>       ;--------------------------------------------------------------------------------
3663                              <1>       ; SHIP_IT
3664                              <1>       ;     THIS ROUTINES HANDLES TRANSMISSION OF COMMAND AND DATA BYTES
3665                              <1>       ;     TO THE KEYBOARD CONTROLLER.
3666                              <1>       ;--------------------------------------------------------------------------------
3667                              <1>       ;
3668 000013C1 6650                <1>       push   ax                  ; SAVE DATA TO SEND
3669                              <1>
3670                              <1>       ;----- WAIT FOR COMMAND TO ACCEPTED
```

```
3671 000013C3 FA                <1>       cli                         ; DISABLE INTERRUPTS TILL DATA SENT
3672                            <1>       ; xor  ecx, ecx             ; CLEAR TIMEOUT COUNTER
3673 000013C4 B900000100        <1>       mov    ecx, 10000h
3674                            <1> S10:
3675 000013C9 E464              <1>       in     al, STATUS_PORT            ; READ KEYBOARD CONTROLLER STATUS
3676 000013CB A802              <1>       test   al, INPT_BUF_FULL   ; CHECK FOR ITS INPUT BUFFER BUSY
3677 000013CD E0FA              <1>       loopnz S10                 ; WAIT FOR COMMAND TO BE ACCEPTED
3678                            <1>
3679 000013CF 6658              <1>       pop    ax                  ; GET DATA TO SEND
3680 000013D1 E664              <1>       out    STATUS_PORT, al           ; SEND TO KEYBOARD CONTROLLER
3681 000013D3 FB                <1>       sti                        ; ENABLE INTERRUPTS AGAIN
3682 000013D4 C3                <1>       retn                       ; RETURN TO CALLER
3683                            <1>
3684                            <1> SND_DATA:
3685                            <1>       ; --------------------------------------------------------------------------------
3686                            <1>       ; SND_DATA
3687                            <1>       ;     THIS ROUTINES HANDLES TRANSMISSION OF COMMAND AND DATA BYTES
3688                            <1>       ;     TO THE KEYBOARD AND RECEIPT OF ACKNOWLEDGEMENTS. IT ALSO
3689                            <1>       ;     HANDLES ANY RETRIES IF REQUIRED
3690                            <1>       ; --------------------------------------------------------------------------------
3691                            <1>       ;
3692 000013D5 6650              <1>       push   ax                  ; SAVE REGISTERS
3693 000013D7 6653              <1>       push   bx
3694 000013D9 51                <1>       push   ecx
3695 000013DA 88C7              <1>       mov    bh, al              ; SAVE TRANSMITTED BYTE FOR RETRIES
3696 000013DC B303              <1>       mov    bl, 3               ; LOAD RETRY COUNT
3697                            <1> SD0:
3698 000013DE FA                <1>       cli                        ; DISABLE INTERRUPTS
3699 000013DF 8025[8F5E0000]CF  <1>       and    byte [KB_FLAG_2], ~(KB_FE+KB_FA) ; CLEAR ACK AND RESEND FLAGS
3700                            <1>       ;
3701                            <1>       ;-----WAIT FOR COMMAND TO BE ACCEPTED
3702 000013E6 B900000100        <1>       mov    ecx, 10000h         ; MAXIMUM WAIT COUNT
3703                            <1> SD5:
3704 000013EB E464              <1>       in     al, STATUS_PORT            ; READ KEYBOARD PROCESSOR STATUS PORT
3705 000013ED A802              <1>       test   al, INPT_BUF_FULL   ; CHECK FOR ANY PENDING COMMAND
3706 000013EF E0FA              <1>       loopnz SD5                 ; WAIT FOR COMMAND TO BE ACCEPTED
3707                            <1>       ;
3708 000013F1 88F8              <1>       mov    al, bh              ; REESTABLISH BYTE TO TRANSMIT
3709 000013F3 E660              <1>       out    PORT_A, al          ; SEND BYTE
3710 000013F5 FB                <1>       sti                        ; ENABLE INTERRUPTS
3711                            <1>       ;mov   cx, 01A00h          ; LOAD COUNT FOR 10 ms+
3712 000013F6 B9FFFF0000        <1>       mov    ecx, 0FFFFh
3713                            <1> SD1:
3714 000013FB F605[8F5E0000]30  <1>       test   byte [KB_FLAG_2], KB_FE+KB_FA ; SEE IF EITHER BIT SET
3715 00001402 750F              <1>       jnz    short SD3           ; IF SET, SOMETHING RECEIVED GO PROCESS
3716 00001404 E2F5              <1>       loop   SD1                 ; OTHERWISE WAIT
3717                            <1> SD2:
3718 00001406 FECB              <1>       dec    bl                  ; DECREMENT RETRY COUNT
3719 00001408 75D4              <1>       jnz    short SD0           ; RETRY TRANSMISSION
3720 0000140A 800D[8F5E0000]80  <1>       or     byte [KB_FLAG_2], KB_ERR ; TURN ON TRANSMIT ERROR FLAG
3721 00001411 EB09              <1>       jmp    short SD4           ; RETRIES EXHAUSTED FORGET TRANSMISSION
3722                            <1> SD3:
3723 00001413 F605[8F5E0000]10  <1>       test   byte [KB_FLAG_2], KB_FA ; SEE IF THIS IS AN ACKNOWLEDGE
3724 0000141A 74EA              <1>       jz     short SD2           ; IF NOT, GO RESEND
3725                            <1> SD4:
3726 0000141C 59                <1>       pop    ecx                 ; RESTORE REGISTERS
3727 0000141D 665B              <1>       pop    bx
3728 0000141F 6658              <1>       pop    ax
3729 00001421 C3                <1>       retn                       ; RETURN, GOOD TRANSMISSION
3730                            <1>
3731                            <1> SND_LED:
3732                            <1>       ; --------------------------------------------------------------------------------
3733                            <1>       ; SND_LED
3734                            <1>       ;     THIS ROUTINES TURNS ON THE MODE INDICATORS.
3735                            <1>       ;
3736                            <1>       ;--------------------------------------------------------------------------------
3737                            <1>       ;
3738 00001422 FA                <1>       cli                        ; TURN OFF INTERRUPTS
3739 00001423 F605[8F5E0000]40  <1>       test   byte [KB_FLAG_2], KB_PR_LED ; CHECK FOR MODE INDICATOR UPDATE
3740 0000142A 755F              <1>       jnz    short SL1           ; DON'T UPDATE AGAIN IF UPDATE UNDERWAY
3741                            <1>       ;
3742 0000142C 800D[8F5E0000]40  <1>       or     byte [KB_FLAG_2], KB_PR_LED ; TURN ON UPDATE IN PROCESS
3743 00001433 B020              <1>       mov    al, EOI                    ; END OF INTERRUPT COMMAND
3744 00001435 E620              <1>       out    20h, al ;out INTA00, al    ; SEND COMMAND TO INTERRUPT CONTROL PORT
3745 00001437 EB11              <1>       jmp    short SL0           ; GO SEND MODE INDICATOR COMMAND
3746                            <1> SND_LED1:
3747 00001439 FA                <1>       cli                        ; TURN OFF INTERRUPTS
3748 0000143A F605[8F5E0000]40  <1>       test   byte [KB_FLAG_2], KB_PR_LED ; CHECK FOR MODE INDICATOR UPDATE
3749 00001441 7548              <1>       jnz    short SL1           ; DON'T UPDATE AGAIN IF UPDATE UNDERWAY
3750                            <1>       ;
3751 00001443 800D[8F5E0000]40  <1>       or     byte [KB_FLAG_2], KB_PR_LED ; TURN ON UPDATE IN PROCESS
3752                            <1> SL0:
3753 0000144A B0ED              <1>       mov    al, LED_CMD         ; LED CMD BYTE
3754 0000144C E884FFFFFF        <1>       call   SND_DATA            ; SEND DATA TO KEYBOARD
3755 00001451 FA                <1>       cli
3756 00001452 E836000000        <1>       call   MAKE_LED            ; GO FORM INDICATOR DATA BYTE
3757 00001457 8025[8F5E0000]F8  <1>       and    byte [KB_FLAG_2], 0F8h    ; ~KB_LEDS ; CLEAR MODE INDICATOR BITS
3758 0000145E 0805[8F5E0000]    <1>       or     [KB_FLAG_2], al    ; SAVE PRESENT INDICATORS FOR NEXT TIME
3759 00001464 F605[8F5E0000]80  <1>       test   byte [KB_FLAG_2], KB_ERR ; TRANSMIT ERROR DETECTED
3760 0000146B 750F              <1>       jnz    short SL2           ; IF SO, BYPASS SECOND BYTE TRANSMISSION
3761                            <1>       ;
3762 0000146D E863FFFFFF        <1>       call   SND_DATA            ; SEND DATA TO KEYBOARD
3763 00001472 FA                <1>       cli                        ; TURN OFF INTERRUPTS
3764 00001473 F605[8F5E0000]80  <1>       test   byte [KB_FLAG_2], KB_ERR ; TRANSMIT ERROR DETECTED
3765 0000147A 7408              <1>       jz     short SL3           ; IF NOT, DON'T SEND AN ENABLE COMMAND
3766                            <1> SL2:
3767 0000147C B0F4              <1>       mov    al, KB_ENABLE       ; GET KEYBOARD CSA ENABLE COMMAND
3768 0000147E E852FFFFFF        <1>       call   SND_DATA            ; SEND DATA TO KEYBOARD
3769 00001483 FA                <1>       cli                        ; TURN OFF INTERRUPTS
3770                            <1> SL3:
3771 00001484 8025[8F5E0000]3F  <1>       and    byte [KB_FLAG_2], ~(KB_PR_LED+KB_ERR) ; TURN OFF MODE INDICATOR
3772                            <1> SL1:                               ; UPDATE AND TRANSMIT ERROR FLAG
```

```
3773 0000148B FB                <1>       sti                     ; ENABLE INTERRUPTS
3774 0000148C C3                <1>       retn                    ; RETURN TO CALLER
3775                            <1>
3776                            <1> MAKE_LED:
3777                            <1>       ;-----------------------------------------------------------------------------
3778                            <1>       ; MAKE_LED
3779                            <1>       ;     THIS ROUTINES FORMS THE DATA BYTE NECESSARY TO TURN ON/OFF
3780                            <1>       ;     THE MODE INDICATORS.
3781                            <1>       ;-----------------------------------------------------------------------------
3782                            <1>       ;
3783                            <1>       ;push  cx                 ; SAVE CX
3784 0000148D A0[8D5E0000]      <1>       mov    al, [KB_FLAG]      ; GET CAPS & NUM LOCK INDICATORS
3785 00001492 2470              <1>       and    al, CAPS_STATE+NUM_STATE+SCROLL_STATE ; ISOLATE INDICATORS
3786                            <1>       ;mov   cl, 4              ; SHIFT COUNT
3787                            <1>       ;rol   al, cl             ; SHIFT BITS OVER TO TURN ON INDICATORS
3788 00001494 C0C004            <1>       rol    al, 4 ; 20/02/2015
3789 00001497 2407              <1>       and    al, 07h                  ; MAKE SURE ONLY MODE BITS ON
3790                            <1>       ;pop   cx
3791 00001499 C3                <1>       retn                    ; RETURN TO CALLER
3792                            <1>
3793                            <1> ; % include 'kybdata.s'   ; KEYBOARD DATA
3794                            <1>
3795                            <1>
3796                            <1> ; /// End Of KEYBOARD FUNCTIONS ///
3797
3798                                %include 'video.s' ; 07/03/2015
3799                            <1> ; ********************************************************************************
3800                            <1> ; TRDOS386.ASM (TRDOS 386 Kernel) - v2.0.0 - video.s
3801                            <1> ; ------------------------------------------------------------------------------
3802                            <1> ; Last Update: 15/01/2017
3803                            <1> ; ------------------------------------------------------------------------------
3804                            <1> ; Beginning: 16/01/2016
3805                            <1> ; ------------------------------------------------------------------------------
3806                            <1> ; Assembler: NASM version 2.11 (trdos386.s)
3807                            <1> ; ------------------------------------------------------------------------------
3808                            <1> ; Turkish Rational DOS
3809                            <1> ; Operating System Project v2.0 by ERDOGAN TAN (Beginning: 04/01/2016)
3810                            <1> ;
3811                            <1> ; Derived from 'Retro UNIX 386 Kernel - v0.2.1.0' source code by Erdogan Tan
3812                            <1> ; video.inc (13/08/2015)
3813                            <1> ;
3814                            <1> ; Derived from 'IBM PC-AT' BIOS source code (1985)
3815                            <1> ; ********************************************************************************
3816                            <1>
3817                            <1> ; Retro UNIX 386 v1 Kernel - VIDEO.INC
3818                            <1> ; Last Modification: 13/08/2015
3819                            <1> ;             (Video Data is in 'VIDATA.INC')
3820                            <1> ;
3821                            <1> ; ///////// VIDEO (CGA) FUNCTIONS ///////////////
3822                            <1>
3823                            <1> ; 16/01/2016 (32 bit modifications, TRDOS386 - TRDOS v2.0, video.s)
3824                            <1> ; INT 31H (TRDOS 386) = INT 10H (IBM PC/AT REAL MODE)
3825                            <1>
3826                            <1> ; IBM PC-AT BIOS Source Code
3827                            <1> ; TITLE VIDEO1 --- 06/10/85  VIDEO DISPLAY BIOS
3828                            <1>
3829                            <1> _int10h:
3830                            <1>       ; 23/03/2016
3831                            <1>       ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
3832 0000149A 9C                <1>       pushfd
3833 0000149B 0E                <1>       push   cs
3834 0000149C E851000000        <1>       call   VIDEO_IO_1
3835 000014A1 C3                <1>       retn
3836                            <1>
3837                            <1> ;--- INT 10 H ----------------------------------------------------------------
3838                            <1> ; VIDEO_IO                                                  :
3839                            <1> ;     THESE ROUTINES PROVIDE THE CRT DISPLAY INTERFACE                :
3840                            <1> ;     THE FOLLOWING FUNCTIONS ARE PROVIDED:                           :
3841                            <1> ;                                                                     :
3842                            <1> ;    (AH)= 00H     SET MODE (AL) CONTAINS MODE VALUE                      :
3843                            <1> ;           (AL) = 00H  40X25 BW MODE (POWER ON DEFAULT)              :
3844                            <1> ;           (AL) = 01H  40X25 COLOR                                   :
3845                            <1> ;           (AL) = 02H  80X25 BW                                      :
3846                            <1> ;           (AL) = 03H  80X25 COLOR                                   :
3847                            <1> ;                       GRAPHICS MODES                                :
3848                            <1> ;           (AL) = 04H  320X200 COLOR                           :
3849                            <1> ;           (AL) = 05H  320X200 BW MODE                               :
3850                            <1> ;           (AL) = 06H  640X200 BW MODE                               :
3851                            <1> ;           (AL) = 07H   80X25 MONOCHROME (USED INTERNAL TO VIDEO ONLY)      :
3852                            <1> ;           *** NOTES -BW MODES OPERATE SAME AS COLOR MODES, BUT COLOR :
3853                            <1> ;                      BURST IS NOT ENABLED                           :
3854                            <1> ;                     -CURSOR IS NOT DISPLAYED IN GRAPHICS MODE       :
3855                            <1> ;    (AH)= 01H     SET CURSOR TYPE                                          :
3856                            <1> ;           (CH) = BITS 4-0 = START LINE FOR CURSOR                   :
3857                            <1> ;                  ** HARDWARE WILL ALWAYS CAUSE BLINK                :
3858                            <1> ;                  ** SETTING BIT 5 OR 6 WILL CAUSE ERRATIC BLINKING  :
3859                            <1> ;                     OR NO CURSOR AT ALL                             :
3860                            <1> ;           (CL) = BITS 4-0 = END LINE FOR CURSOR                     :
3861                            <1> ;    (AH)= 02H     SET CURSOR POSITION                                  :
3862                            <1> ;           (DH,DL) = ROW,COLUMN  (00H,00H) IS UPPER LEFT             :
3863                            <1> ;           (BH) = A PAGE NUMBER (MUST BE 00H FOR GRAPHICS MODES)           :
3864                            <1> ;    (AH)= 03H     READ CURSOR POSITION                                 :
3865                            <1> ;           (BH) = PAGE NUMBER (MUST BE 00H FOR GRAPHICS MODES)       :
3866                            <1> ;           ON EXIT (DH,DL) = ROW,COLUMN OF CURRENT CURSOR                  :
3867                            <1> ;                   (CH,CL) = CURSOR MODE CURRENTLY SET               :
3868                            <1> ;    (AH)= 04H     READ LIGHT PEN POSITION                              :
3869                            <1> ;           ON EXIT:                                            :
3870                            <1> ;           (AH) = 00H -- LIGHT PEN SWITCH NOT DOWN/NOT TRIGGERED           :
3871                            <1> ;           (AH) = 01H -- VALID LIGHT PEN VALUE IN REGISTERS          :
3872                            <1> ;                   (DH,DL) = ROW,COLUMN OF CHARACTER LP POSITION           :
3873                            <1> ;                   (CH) = RASTER LINE (0-199)                        :
3874                            <1> ;                   (BX) = PIXEL COLUMN (0-319,639)                   :
```

```
3875                                    <1> ;    (AH)= 05H     SELECT ACTIVE DISPLAY PAGE (VALID ONLY FOR ALPHA MODES)          :
3876                                    <1> ;            (AL) = NEW PAGE VALUE (0-7 FOR MODES 0&1, 0-3 FOR MODES 2&3)      :
3877                                    <1> ;    (AH)= 06H     SCROLL ACTIVE PAGE UP                                            :
3878                                    <1> ;            (AL) = NUMBER OF LINES. ( LINES BLANKED AT BOTTOM OF WINDOW )     :
3879                                    <1> ;                     (AL) = 00H MEANS BLANK ENTIRE WINDOW                  :
3880                                    <1> ;           (CH,CL) = ROW,COLUMN OF UPPER LEFT CORNER OF SCROLL             :
3881                                    <1> ;           (DH,DL) = ROW,COLUMN OF LOWER RIGHT CORNER OF SCROLL            :
3882                                    <1> ;           (BH) = ATTRIBUTE TO BE USED ON BLANK LINE                       :
3883                                    <1> ;    (AH)= 07H     SCROLL ACTIVE PAGE DOWN                                          :
3884                                    <1> ;           (AL) = NUMBER OF LINES, INPUT LINES BLANKED AT TOP OF WINDOW      :
3885                                    <1> ;                     (AL) = 00H MEANS BLANK ENTIRE WINDOW                  :
3886                                    <1> ;           (CH,CL) = ROW,COLUMN OF UPPER LEFT CORNER OF SCROLL             :
3887                                    <1> ;           (DH,DL) = ROW,COLUMN OF LOWER RIGHT CORNER OF SCROLL            :
3888                                    <1> ;           (BH) = ATTRIBUTE TO BE USED ON BLANK LINE                       :
3889                                    <1> ;                                                                         :
3890                                    <1> ;   CHARACTER HANDLING ROUTINES                                             :
3891                                    <1> ;                                                                         :
3892                                    <1> ;    (AH)= 08H     READ ATTRIBUTE/CHARACTER AT CURRENT CURSOR POSITION         :
3893                                    <1> ;           (BH) = DISPLAY PAGE (VALID FOR ALPHA MODES ONLY)                :
3894                                    <1> ;           ON EXIT:                                                        :
3895                                    <1> ;           (AL) = CHAR READ                                                :
3896                                    <1> ;           (AH) = ATTRIBUTE OF CHARACTER READ (ALPHA MODES ONLY)            :
3897                                    <1> ;    (AH)= 09H     WRITE ATTRIBUTE/CHARACTER AT CURRENT CURSOR POSITION        :
3898                                    <1> ;           (BH) = DISPLAY PAGE (VALID FOR ALPHA MODES ONLY)                :
3899                                    <1> ;           (CX) = COUNT OF CHARACTERS TO WRITE                             :
3900                                    <1> ;           (AL) = CHAR TO WRITE                                            :
3901                                    <1> ;           (BL) = ATTRIBUTE OF CHARACTER (ALPHA)/COLOR OF CHAR (GRAPHICS)  :
3902                                    <1> ;                     SEE NOTE ON WRITE DOT FOR BIT 7 OF BL = 1.            :
3903                                    <1> ;    (AH) = 0AH    WRITE CHARACTER ONLY AT CURRENT CURSOR POSITION                  :
3904                                    <1> ;           (BH) = DISPLAY PAGE (VALID FOR ALPHA MODES ONLY)                :
3905                                    <1> ;           (CX) = COUNT OF CHARACTERS TO WRITE                             :
3906                                    <1> ;           (AL) = CHAR TO WRITE                                            :
3907                                    <1> ;                  NOTE: USE FUNCTION (AH)= 09H IN GRAPHICS MODES           :
3908                                    <1> ;     FOR READ/WRITE CHARACTER INTERFACE WHILE IN GRAPHICS MODE, THE         :
3909                                    <1> ;           CHARACTERS ARE FORMED FROM A CHARACTER GENERATOR IMAGE           :
3910                                    <1> ;           MAINTAINED IN THE SYSTEM ROM. ONLY THE 1ST 128 CHARS          :
3911                                    <1> ;           ARE CONTAINED THERE. TO READ/WRITE THE SECOND 128 CHARS,       :
3912                                    <1> ;           THE USER MUST INITIALIZE THE POINTER AT INTERRUPT 1FH           :
3913                                    <1> ;           (LOCATION 0007CH) TO POINT TO THE 1K BYTE TABLE CONTAINING      :
3914                                    <1> ;           THE CODE POINTS FOR THE SECOND 128 CHARS (128-255).            :
3915                                    <1> ;     FOR WRITE CHARACTER INTERFACE IN GRAPHICS MODE, THE REPLICATION FACTOR :
3916                                    <1> ;           CONTAINED IN (CX) ON ENTRY WILL PRODUCE VALID RESULTS ONLY      :
3917                                    <1> ;           FOR CHARACTERS CONTAINED ON THE SAME ROW. CONTINUATION TO       :
3918                                    <1> ;           SUCCEEDING LINES WILL NOT PRODUCE CORRECTLY.                    :
3919                                    <1> ;                                                                         :
3920                                    <1> ;   GRAPHICS INTERFACE                                                      :
3921                                    <1> ;   (AH)= 0BH     SET COLOR PALETTE                                         :
3922                                    <1> ;           (BH) = PALETTE COLOR ID BEING SET (0-127)                       :
3923                                    <1> ;           (BL) = COLOR VALUE TO BE USED WITH THAT COLOR ID                :
3924                                    <1> ;                  NOTE: FOR THE CURRENT COLOR CARD, THIS ENTRY POINT HAS   :
3925                                    <1> ;                        MEANING ONLY FOR 320X200 GRAPHICS.               :
3926                                    <1> ;                  COLOR ID = 0 SELECTS THE BACKGROUND COLOR (0-15)          :
3927                                    <1> ;                  COLOR ID = 1 SELECTS THE PALETTE TO BE USED:              :
3928                                    <1> ;                        0 = GREEN(1)/RED(2)/YELLOW(3)                     :
3929                                    <1> ;                        1 = CYAN(1)/MAGENTA(2)/WHITE(3)                   :
3930                                    <1> ;                  IN 40X25 OR 80X25 ALPHA MODES, THE VALUE SET FOR         :
3931                                    <1> ;                        PALETTE COLOR 0 INDICATES THE BORDER COLOR        :
3932                                    <1> ;                        TO BE USED (VALUES 0-31, WHERE 16-31 SELECT       :
3933                                    <1> ;                        THE HIGH INTENSITY BACKGROUND SET.                :
3934                                    <1> ;   (AH)= 0CH     WRITE DOT                                                 :
3935                                    <1> ;           (DX) = ROW NUMBER                                               :
3936                                    <1> ;           (CX) = COLUMN NUMBER                                            :
3937                                    <1> ;           (AL) = COLOR VALUE                                              :
3938                                    <1> ;                   IF BIT 7 OF AL = 1, THEN THE COLOR VALUE IS EXCLUSIVE   :
3939                                    <1> ;                   ORed WITH THE CURRENT CONTENTS OF THE DOT                :
3940                                    <1> ;   (AH)= ODH     READ DOT                                                  :
3941                                    <1> ;           (DX) = ROW NUMBER                                               :
3942                                    <1> ;           (CX) = COLUMN NUMBER                                            :
3943                                    <1> ;           (AL) = RETURNS THE DOT READ                                     :
3944                                    <1> ;                                                                         :
3945                                    <1> ;   ASCII TELETYPE ROUTINE FOR OUTPUT                                       :
3946                                    <1> ;                                                                         :
3947                                    <1> ;    (AH)= 0EH     WRITE TELETYPE TO ACTIVE PAGE                               :
3948                                    <1> ;            (AL) = CHAR TO WRITE                                           :
3949                                    <1> ;            (BL) = FOREGROUND COLOR IN GRAPHICS MODE                       :
3950                                    <1> ;            NOTE -- SCREEN WIDTH IS CONTROLLED BY PREVIOUS MODE SET          :
3951                                    <1> ;    (AH)= 0FH     CURRENT VIDEO STATE                                       :
3952                                    <1> ;            RETURNS THE CURRENT VIDEO STATE                                :
3953                                    <1> ;            (AL) = MODE CURRENTLY SET ( SEE (AH)=00H FOR EXPLANATION)      :
3954                                    <1> ;            (AH) = NUMBER OR CHARACTER COLUMNS ON SCREEN                    :
3955                                    <1> ;            (BH) = CURRENT ACTIVE DISPLAY PAGE                             :
3956                                    <1> ;    (AH)= 10H     RESERVED                                                  :
3957                                    <1> ;    (AH)= 11H     RESERVED                                                  :
3958                                    <1> ;    (AH)= 12H     RESERVED                                                  :
3959                                    <1> ;    (AH)= 13H     WRITE STRING                                              :
3960                                    <1> ;                  ES:BP  -  POINTER T0 STRING TO BE WRITTEN                 :
3961                                    <1> ;                  CX     -  LENGTH OF CHARACTER STRING TO WRITTEN             :
3962                                    <1> ;                  DX     -  CURSOR POSITION FOR STRING TO BE WRITTEN          :
3963                                    <1> ;                  BH     -  PAGE NUMBER                                       :
3964                                    <1> ;            (AL)= 00H    WRITE CHARACTER STRING                             :
3965                                    <1> ;                  BL     -  ATTRIBUTE                                         :
3966                                    <1> ;                  STRING IS  <CHAR,CHAR, ... ,CHAR>                         :
3967                                    <1> ;                  CURSOR NOT MOVED                                         :
3968                                    <1> ;            (AL)= 01H    WRITE CHARACTER STRING AND MOVE CURSOR             :
3969                                    <1> ;                  BL     -  ATTRIBUTE                                         :
3970                                    <1> ;                  STRING IS  <CHAR,CHAR, ... ,CHAR>                         :
3971                                    <1> ;                  CURSOR MOVED                                             :
3972                                    <1> ;            (AL)= 02H    WRITE CHARACTER AND ATTRIBUTE STRING               :
3973                                    <1> ;                         (VALID FOR ALPHA MODES ONLY)                      :
3974                                    <1> ;                  STRING IS <CHAR,ATTR,CHAR,ATTR ..  ,CHAR,ATTR>             :
3975                                    <1> ;                  CURSOR IS NOT MOVED                                      :
3976                                    <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>
000014A2 [4F150000]             <1> M1:   dd    SET_MODE     ; TABLE OF ROUTINES WITHIN VIDEO I/O
000014A6 [B7180000]             <1>       dd    SET_CTYPE
000014AA [EB180000]             <1>       dd    SET_CPOS
000014AE [13190000]             <1>       dd    READ_CURSOR
                                <1>       ;dd   VIDEO_RETURN ; READ_LPEN
000014B2 [38150000]             <1>       dd    set_mode_ncm ; Set mode without clearing video memory
000014B6 [59190000]             <1>       dd    ACT_DISP_PAGE
000014BA [F0190000]             <1>       dd    SCROLL_UP
000014BE [141B0000]             <1>       dd    SCROLL_DOWN
000014C2 [951B0000]             <1>       dd    READ_AC_CURRENT
000014C6 [ED1B0000]             <1>       dd    WRITE_AC_CURRENT
000014CA [131C0000]             <1>       dd    WRITE_C_CURRENT
000014CE [39250000]             <1>       dd    SET_COLOR
000014D2 [A4250000]             <1>       dd    WRITE_DOT
000014D6 [6F250000]             <1>       dd    READ_DOT
000014DA [951C0000]             <1>       dd    WRITE_TTY
000014DE [20150000]             <1>       dd    VIDEO_STATE
000014E2 [EF2E0000]             <1>       dd    vga_pal_funcs ; 10/08/2016 (TRDOS 386)
000014E6 [A52A0000]             <1>       dd    font_setup    ; 10/07/2016 (TRDOS 386)
000014EA [54150000]             <1>       dd    VIDEO_RETURN ; RESERVED
000014EE [021E0000]             <1>       dd    WRITE_STRING  ; 23/06/2016 (TRDOS 386)
                                <1> M1L   EQU   $ - M1
                                <1>
                                <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
000014F2 FC                     <1>       cld                       ; SET DIRECTION FORWARD
000014F3 80FC14                 <1>       cmp   ah, M1L/4           ; TEST FOR WITHIN TABLE RANGE
000014F6 7327                   <1>       jnb   short M4            ; BRANCH TO EXIT IF NOT A VALID COMMAND
                                <1>
000014F8 06                     <1>       push  es
000014F9 1E                     <1>       push  ds                  ; SAVE WORK AND PARAMETER REGISTERS
000014FA 52                     <1>       push  edx
000014FB 51                     <1>       push  ecx
000014FC 53                     <1>       push  ebx
000014FD 56                     <1>       push  esi
000014FE 57                     <1>       push  edi
000014FF 55                     <1>       push  ebp
                                <1>
00001500 66BE1000               <1>       mov   si, KDATA           ; POINT DS: TO DATA SEGMENT
00001504 8EDE                   <1>       mov   ds, si
00001506 8EC6                   <1>       mov   es, si
00001508 BF00800B00             <1>       mov   edi, 0B8000h        ; GET offset FOR COLOR CARD
0000150D A3[545B0100]           <1>       mov   [video_eax], eax ; 12/05/2016
                                <1>       ; 23/03/2016
00001512 C0E402                 <1>       shl   ah, 2  ; dword            ; TIMES 2 FOR WORD TABLE LOOKUP
00001515 0FB6F4                 <1>       movzx esi, ah                   ; MOVE OFFSET INTO LOOK UP REGISTER (SI)
                                <1>       ;mov  ah, [CRT_MODE]            ; MOVE CURRENT MODE INTO (AH) REGISTER
                                <1>
                                <1>       ;;15/01/2017
                                <1>       ; 14/01/2017
                                <1>       ; 02/01/2017
                                <1>       ;;mov  byte [intflg], 31h  ; video interrupt
00001518 FB                     <1>       sti
                                <1>       ;
                                <1>
00001519 FFA6[A2140000]         <1>       JMP   dword [esi+M1]            ; GO TO SELECTED FUNCTION
                                <1>
                                <1> M4:                             ; COMMAND NOT VALID
0000151F CF                     <1>       iretd                     ; DO NOTHING IF NOT IN VALID RANGE
                                <1>
                                <1> VIDEO_STATE:
                                <1>       ; 26/06/2016
                                <1>       ; 12/05/2016
                                <1>       ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>
                                <1> ;-------------------------------------------------
                                <1> ; VIDEO STATE
                                <1> ;  RETURNS THE CURRENT VIDEO STATE IN AX
                                <1> ;  AH = NUMBER OF COLUMNS ON THE SCREEN
                                <1> ;  AL = CURRENT VIDEO MODE
                                <1> ;  BH = CURRENT ACTIVE PAGE
                                <1> ;-------------------------------------------------
                                <1>
00001520 8A25[C45E0000]         <1>       mov   ah, [CRT_COLS]      ; GET NUMBER OF COLUMNS
00001526 A0[C25E0000]           <1>       mov   al, [CRT_MODE]      ; CURRENT MODE
                                <1>       ;movzx esi, al
                                <1>       ;mov  ah, [esi+M6]
                                <1>       ; BH = active page
0000152B 8A3D[F64D0100]         <1>       mov   bh, [ACTIVE_PAGE] ; GET CURRENT ACTIVE PAGE
```

```
4079 00001531 FA                <1>       cli    ; 02/01/2017
4080 00001532 5D                <1>       pop    ebp          ; RECOVER REGISTERS
4081 00001533 5F                <1>       pop    edi
4082 00001534 5E                <1>       pop    esi
4083 00001535 59                <1>       pop    ecx          ; DISCARD SAVED BX
4084 00001536 EB26              <1>       jmp    short M15    ; RETURN TO CALLER
4085                            <1>
4086                            <1> set_mode_ncm:
4087                            <1>       ; 04/07/2016 - TRDOS 386 (TRDOS v2.0)
4088                            <1>       ; set mode without clearing the video memory
4089                            <1>       ; (ony for graphics modes)
4090 00001538 3C07              <1>       cmp    al, 7 ; IBM PC CGA modes
4091 0000153A 7613              <1>       jna    short SET_MODE ; normal function (clear)
4092                            <1>       ; do not clear memory
4093 0000153C A2[635B0100]      <1>       mov    [noclearmem], al ; > 0
4094 00001541 E81F000000        <1>       call   _set_mode
4095 00001546 C605[635B0100]00  <1>       mov    byte [noclearmem], 0
4096 0000154D EB05              <1>         jmp     short VIDEO_RETURN
4097                            <1>
4098                            <1>       ; 10/08/2016
4099                            <1>       ; 08/08/2016
4100                            <1>       ; 30/07/2016
4101                            <1>       ; 29/07/2016
4102                            <1>       ; 27/07/2016
4103                            <1>       ; 26/07/2016
4104                            <1>       ; 25/07/2016
4105                            <1>       ; 23/07/2016
4106                            <1>       ; 18/07/2016
4107                            <1>       ; 02/07/2016
4108                            <1>       ; 26/06/2016
4109                            <1>       ; 24/06/2016
4110                            <1>       ; 29/05/2016 - TRDOS 386 (TRDOS v2.0)
4111                            <1> SET_MODE:
4112                            <1>       ; For 32 bit TRDOS and Retro UNIX 386:
4113                            <1>       ;     valid video mode: 03h only!
4114                            <1>       ;     (VGA modes will be selected with another routine)
4115                            <1>       ;
4116                            <1>       ; set_txt_mode ; 80*25 (16 fore colors, 8 back colors)
4117                            <1>
4118                            <1> ;-------------------------------------------------
4119                            <1> ; SET MODE                                     :
4120                            <1> ;     THIS ROUTINE INITIALIZES THE ATTACHMENT TO    :
4121                            <1> ;     THE SELECTED MODE, THE SCREEN IS BLANKED.     :
4122                            <1> ; INPUT                                          :
4123                            <1> ;     (AL) - MODE SELECTED (RANGE 0-7)       :
4124                            <1> ; OUTPUT                                       :
4125                            <1> ;     NONE                                     :
4126                            <1> ;-------------------------------------------------
4127                            <1>
4128 0000154F E811000000        <1>       call   _set_mode ; 24/06/2016 (set_txt_mode)
4129                            <1>
4130                            <1> ; 12/05/2016
4131                            <1> ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
4132                            <1>
4133                            <1> ;-----      NORMAL RETURN FROM ALL VIDEO RETURNS
4134                            <1>
4135                            <1> VIDEO_RETURN:
4136 00001554 A1[545B0100]      <1>       mov    eax, [video_eax] ; 12/05/2016
4137                            <1> _video_return:
4138 00001559 FA                <1>       cli ; 02/01/2017
4139 0000155A 5D                <1>       pop    ebp
4140 0000155B 5F                <1>       pop    edi
4141 0000155C 5E                <1>       pop    esi
4142 0000155D 5B                <1>       pop    ebx
4143                            <1> M15:  ; VIDEO_RETURN_C
4144                            <1>       ;;15/01/2017
4145                            <1>       ; 02/01/2017
4146                            <1>       ;;mov  byte [intflg], 0
4147                            <1>       ;
4148 0000155E 59                <1>       pop    ecx
4149 0000155F 5A                <1>       pop    edx
4150 00001560 1F                <1>       pop    ds
4151 00001561 07                <1>       pop    es     ; RECOVER SEGMENTS
4152 00001562 CF                <1>       iretd         ; ALL DONE
4153                            <1>
4154                            <1> set_txt_mode:
4155                            <1>       ; 29/07/2016
4156                            <1>       ; 27/06/2016
4157 00001563 B003              <1>       mov    al, 3
4158                            <1>
4159                            <1> ; 10/08/2016
4160                            <1> ; 08/08/2016
4161                            <1> ; 30/07/2016
4162                            <1> ; 29/07/2016
4163                            <1> ; 27/07/2016
4164                            <1> ; 26/07/2016
4165                            <1> ; 25/07/2016
4166                            <1> ; 23/07/2016
4167                            <1> ; 18/07/2016
4168                            <1> ; 07/07/2016
4169                            <1> ; 04/07/2016
4170                            <1> ; 03/07/2016
4171                            <1> ; 02/07/2016
4172                            <1> ; 26/06/2016
4173                            <1> ; 24/06/2016 (set_txt_mode -> _set_mode)
4174                            <1> ; 17/06/2016
4175                            <1> ; 29/05/2016
4176                            <1> ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
4177                            <1> _set_mode:
4178                            <1>       ; 24/06/2016
4179 00001565 3805[C25E0000]    <1>       cmp    [CRT_MODE], al  ; current mode = requested mode ?
4180 0000156B 750D              <1>       jne    short _sm_0
```

```
4181 0000156D 3C03                <1>       cmp    al, 3        ; text, 80*25 color, default mode
4182                              <1>                           ; for TRDOS 386 MainProg
4183 0000156F 755F                <1>         jne     short _sm_2     ;  multiscreen is only for mode 3
4184                              <1>
4185                              <1>       ; If '_set_mode' procedure is called for video mode 3
4186                              <1>       ;     while video mode is 3, video page will be cleared
4187                              <1>       ;     and cursor position of video page will be reset.
4188                              <1>
4189                              <1>       ; 29/07/2016
4190 00001571 800D[615B0100]80    <1>       or     byte [p_crt_mode], 80h ; clear page indicator
4191 00001578 EB5B                <1>         jmp     short _sm_3
4192                              <1> _sm_0:
4193 0000157A 803D[C25E0000]03    <1>       cmp    byte [CRT_MODE], 3
4194 00001581 7534                <1>         jne     short _sm_1
4195                              <1>
4196                              <1>       ; If '_set_mode' procedure is called for a video mode
4197                              <1>       ;     except video mode 3, while current video mode
4198                              <1>       ;     is 3; all video pages of mode 3 will be copied
4199                              <1>       ;     to 98000h address as backup, before mode change.
4200                              <1>
4201                              <1> _sm_save_pm:
4202                              <1>       ; 03/07/2016
4203                              <1>       ; save video pages
4204 00001583 BE00800B00          <1>       mov    esi, 0B8000h
4205 00001588 BF00800900          <1>       mov    edi, 98000h ; 30/07/2016
4206 0000158D B900200000          <1>       mov    ecx, (0B8000h-0B0000h)/4
4207 00001592 F3A5                <1>       rep    movsd
4208                              <1>
4209 00001594 C605[615B0100]03    <1>       mov    byte [p_crt_mode], 3 ; previous mode, backup sign
4210                              <1>       ;mov   cl, [ACTIVE_PAGE]
4211                              <1>       ;mov   [p_crt_page], cl
4212                              <1>
4213                              <1>       ; save cursor positions
4214 0000159B BE[E64D0100]        <1>       mov    esi, CURSOR_POSN
4215 000015A0 BF[665B0100]        <1>       mov    edi, cursor_pposn    ; cursor positions backup
4216 000015A5 B104                <1>       mov    cl, 4
4217 000015A7 F3A5                <1>       rep    movsd
4218                              <1>
4219                              <1>       ; 29/07/2016
4220                              <1>       ;mov   [ACTIVE_PAGE], cl ; 0
4221 000015A9 860D[F64D0100]      <1>       xchg   cl, [ACTIVE_PAGE]
4222 000015AF 880D[625B0100]      <1>       mov    [p_crt_page], cl     ; previous page (for mode 3)
4223                              <1>       ; [ACTIVE_PAGE] = 0
4224 000015B5 EB19                <1>       jmp    short _sm_2
4225                              <1>
4226                              <1> _sm_1:
4227 000015B7 3C03                <1>       cmp    al, 3        ; text, 80*25 color, default mode
4228                              <1>                           ; for TRDOS 386 MainProg
4229 000015B9 7515                <1>       jne    short _sm_2  ;  multiscreen is only for mode 3
4230                              <1>
4231                              <1>       ; If '_set_mode' procedure is called for video mode 3
4232                              <1>       ;     while video mode is not 3 and if there is video
4233                              <1>       ;     page backup for video mode 3, all (of 8) mode 3
4234                              <1>       ;     video pages will be restored from 98000h.
4235                              <1>
4236 000015BB 803D[615B0100]03    <1>       cmp    byte [p_crt_mode], 3 ; previous mode, backup sign
4237 000015C2 750C                <1>       jne    short _sm_2 ; there is no (multiscreen) video pages
4238                              <1>                          ; to be restored
4239 000015C4 8A0D[625B0100]      <1>       mov    cl, [p_crt_page]
4240 000015CA 880D[F64D0100]      <1>       mov    [ACTIVE_PAGE], cl
4241                              <1>
4242                              <1> _sm_2:
4243 000015D0 A2[C25E0000]        <1>       mov    [CRT_MODE], al  ; save mode in global variable
4244                              <1> _sm_3:
4245                              <1>       ; 30/07/2016
4246                              <1>       ; 26/07/2016
4247                              <1>       ; 25/07/2016
4248                              <1>       ; set_mode_vga:
4249                              <1>       ; 18/07/2016
4250                              <1>       ; 14/07/2016
4251                              <1>       ; 09/07/2016
4252                              <1>       ; 04/07/2016
4253                              <1>       ; 03/07/2016 (TRDOS 386 = TRDOS v2.0)
4254                              <1>       ; /// video mode 13h ///
4255                              <1>       ; derived from 'Plex86/Bochs VGABios' source code
4256                              <1>       ; vgabios-0.7a (2011)
4257                              <1>       ; by the LGPL VGABios developers Team (2001-2008)
4258                              <1>       ; 'vgabios.c', 'vgatables.h'
4259                              <1>       ;
4260                              <1>       ; Oracle VirtualBox 5.0.24 VGABios Source Code
4261                              <1>       ; ('vgabios.c', 'vgatables.h', 'vgafonts.h', 'vgarom.asm')
4262                              <1>       ;
4263 000015D5 88C4                <1>       mov    ah, al
4264 000015D7 B910000000          <1>       mov    ecx, vga_mode_count
4265 000015DC BE[DE5E0000]        <1>       mov    esi, vga_modes
4266 000015E1 31DB                <1>       xor    ebx, ebx
4267                              <1> _sm_4:
4268 000015E3 AC                  <1>       lodsb
4269 000015E4 38C4                <1>       cmp    ah, al
4270 000015E6 740C                <1>       je     short _sm_5
4271 000015E8 FEC3                <1>       inc    bl
4272 000015EA E2F7                <1>       loop   _sm_4
4273                              <1>
4274                              <1>       ; UNIMPLEMENTED VIDEO MODE !
4275 000015EC 31C0                <1>       xor    eax, eax
4276 000015EE A3[545B0100]        <1>         mov     [video_eax], eax ; 0
4277 000015F3 C3                  <1>       retn
4278                              <1>
4279                              <1> ;-----      eBX POINTS TO CORRECT ROW OF INITIALIZATION TABLE
4280                              <1>
4281                              <1> _sm_5:      ; 25/07/2016
4282 000015F4 89DE                <1>       mov    esi, ebx
```

```
4283 000015F6 81C6[2E5F0000]      <1>       add   esi, vga_memmodel
4284 000015FC 8A06                <1>       mov   al, [esi]
4285 000015FE A2[7A5B0100]        <1>       mov   [VGA_MTYPE], al
4286                              <1>
4287 00001603 89DF                <1>       mov   edi, ebx
4288 00001605 81C7[3E5F0000]      <1>       add   edi, vga_dac_s
4289 0000160B C0E302              <1>       shl   bl, 2 ; byte -> dword
4290 0000160E 81C3[EE5E0000]      <1>       add   ebx, vga_mode_tbl_ptr
4291                              <1>
4292                              <1>       ;mov  dword [VGA_BASE], 0B8000h
4293                              <1>       ;cmp  ah, 0Dh ; [CRT_MODE]
4294                              <1>       ;jb   short M9
4295                              <1>       ;mov  dword [VGA_BASE], 0A0000h
4296                              <1> ;M9:
4297 00001614 8B33                <1>       mov   esi, [ebx]
4298 00001616 89F3                <1>       mov   ebx, esi
4299 00001618 83C614              <1>       add   esi, vga_p_cm_pos ; ebx + 20
4300 0000161B 668B06              <1>       mov   ax, [esi]       ; get the cursor mode from the table
4301 0000161E 66A3[DB5E0000]      <1>       mov   [CURSOR_MODE], ax ; save cursor mode (initial value)
4302                              <1>       ; al = 6, ah = 7
4303                              <1>       ; al = 0Dh, ah = 0Eh ; 25/07/2016
4304 00001624 E83B020000          <1>       call  cursor_shape_fix
4305                              <1>       ; al = 14, ah = 15  (If [CHAR_HEIGHT] = 16)
4306 00001629 668906              <1>       mov   [esi], ax
4307                              <1>
4308 0000162C 56                  <1>       push  esi ; *
4309                              <1>
4310 0000162D 8A25[C95E0000]      <1>       mov   ah, [VGA_MODESET_CTL]
4311 00001633 80E408              <1>       and   ah, 8 ; default palette loading ?
4312 00001636 7524                <1>       jnz   short _sm_6
4313 00001638 66BAC603            <1>       mov   dx, 3C6h ; VGAREG_PEL_MASK (DAC mask register)
4314 0000163C B0FF                <1>       mov   al, 0FFh ; PEL mask
4315 0000163E EE                  <1>       out   dx, al
4316 0000163F 8A27                <1>       mov   ah, [edi] ; DAC model (selection number)
4317 00001641 E8ED0F0000          <1>       call  load_dac_palette
4318                              <1>       ; ecx = 0
4319 00001646 F605[C95E0000]02    <1>       test  byte [VGA_MODESET_CTL], 2 ; gray scale summing
4320 0000164D 740D                <1>       jz    short _sm_6
4321 0000164F 53                  <1>       push  ebx
4322 00001650 29DB                <1>       sub   ebx, ebx ; sub bl, bl
4323 00001652 66B90001            <1>         mov     cx, 256
4324 00001656 E82B100000          <1>       call  gray_scale_summing
4325 0000165B 5B                  <1>       pop   ebx
4326                              <1> _sm_6:
4327                              <1>       ; Reset Attribute Ctl flip-flop
4328 0000165C 66BADA03            <1>       mov   dx, 3DAh ; VGAREG_ACTL_RESET
4329 00001660 EC                  <1>       in    al, dx
4330                              <1>       ; Set Attribute Ctl
4331 00001661 89DE                <1>       mov   esi, ebx ; addr of params tbl for selected mode
4332 00001663 83C623              <1>       add   esi, 35  ; actl regs
4333 00001666 30E4                <1>       xor   ah, ah ; 0
4334 00001668 66BAC003            <1>       mov   dx, 3C0h ; VGAREG_ACTL_ADDRESS
4335                              <1> _sm_7:
4336 0000166C 88E0                <1>       mov   al, ah
4337 0000166E EE                  <1>       out   dx, al ; index
4338 0000166F AC                  <1>       lodsb
4339                              <1>       ; DX = 3C0h = VGAREG_ACTL_WRITE_DATA
4340 00001670 EE                  <1>       out   dx, al ; value
4341 00001671 FEC4                <1>       inc   ah
4342 00001673 80FC14              <1>       cmp   ah, 20 ; number of actl registers
4343 00001676 72F4                <1>       jb    short _sm_7
4344                              <1>       ;
4345 00001678 88E0                <1>       mov   al, ah ; 20
4346 0000167A EE                  <1>       out   dx, al ; index
4347 0000167B 28C0                <1>       sub   al, al ; 0
4348 0000167D EE                  <1>       out   dx, al ; value
4349                              <1>       ;
4350                              <1>       ; Set Sequencer Ctl
4351 0000167E 89DE                <1>       mov   esi, ebx ; addr of params tbl for selected mode
4352 00001680 83C605              <1>       add   esi, 5 ; sequ regs
4353                              <1>       ;
4354 00001683 66BAC403            <1>       mov   dx, 3C4h  ; VGAREG_SEQU_ADDRESS
4355 00001687 EE                  <1>       out   dx, al ; 0
4356 00001688 6642                <1>       inc   dx ; 3C5h ; VGAREG_SEQU_DATA
4357 0000168A B003                <1>       mov   al, 3
4358 0000168C EE                  <1>       out   dx, al
4359 0000168D B401                <1>       mov   ah, 1
4360                              <1> _sm_8:
4361 0000168F 88E0                <1>       mov   al, ah
4362                              <1>       ;mov  dx, 3C4h ; VGAREG_SEQU_ADDRESS
4363 00001691 664A                <1>       dec   dx
4364 00001693 EE                  <1>       out   dx, al ; index
4365 00001694 AC                  <1>       lodsb
4366 00001695 6642                <1>       inc   dx ; 3C5h ; VGAREG_SEQU_DATA
4367 00001697 EE                  <1>       out   dx, al
4368 00001698 80FC04              <1>       cmp   ah, 4 ; number of sequ regs
4369 0000169B 7304                <1>       jnb   short _sm_9
4370 0000169D FEC4                <1>       inc   ah
4371 0000169F EBEE                <1>       jmp   short _sm_8
4372                              <1> _sm_9:
4373                              <1>       ; Set Grafx Ctl
4374 000016A1 89DE                <1>       mov   esi, ebx ; addr of params tbl for selected mode
4375 000016A3 83C637              <1>       add   esi, 55 ; grdc regs
4376 000016A6 30E4                <1>       xor   ah, ah ; 0
4377                              <1> _sm_10:
4378 000016A8 88E0                <1>       mov   al, ah
4379 000016AA 66BACE03            <1>       mov   dx, 3CEh ; VGAREG_GRDC_ADDRESS
4380 000016AE EE                  <1>       out   dx, al
4381 000016AF AC                  <1>       lodsb
4382 000016B0 6642                <1>       inc   dx ; 3CFh ; VGAREG_GRDC_DATA
4383 000016B2 EE                  <1>       out   dx, al
4384 000016B3 FEC4                <1>       inc   ah
```

```
4385 000016B5 80FC09              <1>       cmp    ah, 9 ; number of grdc regs
4386 000016B8 72EE                <1>       jb     short _sm_10
4387                              <1>       ;
4388                              <1>       ; Disable CRTC write protection
4389 000016BA 66BAD403            <1>       mov    dx, 3D4h  ; VGAREG_VGA_CRTC_ADDRESS
4390                              <1>       ;mov   al, 11h
4391                              <1>       ;our   dx, al
4392                              <1>       ;inc   dx
4393                              <1>       ;sub   al, al
4394                              <1>       ;out   dx, al
4395 000016BE 66B81100           <1>       mov    ax, 11h
4396 000016C2 66EF                <1>       out    dx, ax
4397 000016C4 89DE                <1>       mov    esi, ebx ; addr of params tbl for selected mode
4398 000016C6 83C60A              <1>       add    esi, 10 ; crtc regs
4399                              <1>       ; ah = 0
4400                              <1> _sm_11:
4401 000016C9 88E0                <1>       mov    al, ah
4402                              <1>       ; dx = 3D4h = VGAREG_VGA_CRTC_ADDRESS
4403 000016CB EE                  <1>       out    dx, al ; index
4404 000016CC AC                  <1>       lodsb
4405 000016CD 6642                <1>       inc    dx  ; VGAREG_VGA_CRTC_ADDRESS + 1
4406 000016CF EE                  <1>       out    dx, al ; value
4407 000016D0 80FC18              <1>       cmp    ah, 24 ; number of crtc registers - 1
4408 000016D3 7306                <1>       jnb    short _sm_12
4409 000016D5 FEC4                <1>       inc    ah
4410 000016D7 664A                <1>       dec    dx ; 3D4h
4411 000016D9 EBEE                <1>       jmp    short _sm_11
4412                              <1> _sm_12:
4413                              <1>       ; Set the misc register
4414 000016DB 66BACC03            <1>       mov    dx, 3CCh ; VGAREG_READ_MISC_OUTPUT
4415 000016DF 8A4309              <1>       mov    al, [ebx+9] ; misc reg
4416 000016E2 EE                  <1>       out    dx, al
4417                              <1>       ;
4418                              <1>       ; Enable video
4419 000016E3 66BAC003            <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
4420 000016E7 B020                <1>       mov    al, 20h
4421 000016E9 EE                  <1>         out    dx, al   ; set bit 5 to 1
4422 000016EA 66BADA03            <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
4423 000016EE EC                  <1>       in     al, dx
4424                              <1>       ;
4425 000016EF 803D[635B0100]00    <1>       cmp    byte [noclearmem], 0
4426 000016F6 7740                <1>         ja     short _sm_15
4427                              <1>
4428                              <1>       ; 29/07/2016
4429 000016F8 31C0                <1>       xor    eax, eax
4430 000016FA B900400000          <1>       mov    ecx, 4000h ; 16K words (32K)
4431 000016FF 803D[7A5B0100]02    <1>       cmp     byte [VGA_MTYPE], 2  ; CTEXT, MTEXT, CGA
4432 00001706 7715                <1>       ja     short _sm_14    ; no ? (0A0000h)
4433 00001708 BF00800B00          <1>       mov    edi, 0B8000h
4434 0000170D 7409                <1>       je     short _sm_13 ; CGA graphics mode
4435                              <1>       ; 08/08/2016
4436 0000170F A3[765B0100]        <1>       mov    [VGA_INT43H], eax ; 0 ; default font
4437 00001714 66B82007            <1>       mov    ax, 0720h    ; CGA text mode
4438                              <1> _sm_13:
4439 00001718 F366AB              <1>       rep    stosw
4440 0000171B EB1B                <1>       jmp    short _sm_15
4441                              <1>
4442                              <1> _sm_14:
4443 0000171D BF00000A00          <1>       mov    edi, 0A0000h
4444                              <1>       ; ecx = 16384 dwords  (64K)
4445 00001722 66BAC403            <1>       mov    dx, 3C4h ; VGAREG_SEQU_ADDRESS
4446 00001726 B002                <1>       mov    al, 2
4447 00001728 EE                  <1>       out    dx, al
4448                              <1>       ;mov   dx, 3C5h ; VGAREG_SEQU_DATA
4449 00001729 6642                <1>       inc    dx
4450 0000172B EC                  <1>       in     al, dx ; mmask
4451 0000172C 6650                <1>       push   ax
4452 0000172E B00F                <1>       mov    al, 0Fh ; all planes
4453 00001730 EE                  <1>       out    dx, al
4454 00001731 30C0                <1>       xor    al, al ; 0
4455 00001733 F3AB                <1>       rep    stosd  ; ecx = 163684 (64K)
4456 00001735 6658                <1>       pop    ax
4457 00001737 EE                  <1>       out    dx, al  ; mmask
4458                              <1> _sm_15:
4459                              <1>       ; ebx = addr of params tbl for selected mode
4460                              <1>       ; 10/08/2016
4461 00001738 668B03              <1>       mov    ax, [ebx] ; num of columns, 'twidth'
4462 0000173B A2[C45E0000]        <1>       mov    [CRT_COLS], al
4463                              <1>       ;; 26/07/2016
4464                              <1>       ;; CRTC_ADDRESS = 3D4h (always)
4465                              <1>       ;mov   ah, [ebx+1] ; num of rows, 'theightm1'
4466 00001740 FEC4                <1>       inc    ah ; 09/07/2016
4467 00001742 8825[CA5E0000]      <1>       mov    [VGA_ROWS], ah
4468                              <1>       ; 10/08/2016
4469 00001748 8A4302              <1>       mov    al, [ebx+2]
4470 0000174B A2[C65E0000]        <1>       mov    [CHAR_HEIGHT], al
4471                              <1>       ; 29/07/2016
4472                              <1>       ; length of regen buffer in bytes
4473 00001750 668B4B03            <1>       mov    cx, [ebx+3] ; 'slength_l'
4474 00001754 66890D[645B0100]    <1>       mov    [CRT_LEN], cx
4475                              <1>       ;
4476                              <1>       ; 27/07/2016
4477 0000175B 30E4                <1>       xor    ah, ah
4478 0000175D A0[F64D0100]        <1>       mov    al, [ACTIVE_PAGE] ; may be > 0 for mode 3
4479                              <1>       ;mul   word [CRT_LEN] ; 4096 for mode 3
4480 00001762 66F7E1              <1>       mul    cx ; 29/07/2016
4481 00001765 66A3[E44D0100]      <1>       mov    [CRT_START], ax
4482                              <1>       ;
4483 0000176B B060                <1>       mov    al, 60h
4484 0000176D 803D[635B0100]00    <1>       cmp    byte [noclearmem], 0
4485 00001774 7602                <1>       jna    short _sm_16
4486 00001776 0480                <1>       add    al, 80h
```

```
4487                                    <1> _sm_16:
4488 00001778 A2[C75E0000]              <1>         mov    [VGA_VIDEO_CTL], al
4489 0000177D C605[C85E0000]F9          <1>         mov    byte [VGA_SWITCHES], 0F9h
4490 00001784 8025[C95E0000]7F          <1>         and    byte [VGA_MODESET_CTL], 7Fh
4491                                    <1>
4492 0000178B 5E                        <1>         pop    esi ; *
4493                                    <1>
4494                                    <1>         ; 26/07/2016
4495                                    <1>         ; 07/07/2016
4496 0000178C 668B0D[DB5E0000]          <1>         mov    cx, [CURSOR_MODE] ; restore cursor mode (initial value)
4497 00001793 66870E                    <1>         xchg   cx, [esi] ; cl = start line, ch = end line
4498                                    <1>                          ; reset to initial value
4499 00001796 86E9                      <1>         xchg   ch, cl  ; ch = start line, cl = end line
4500 00001798 66890D[DB5E0000]          <1>         mov    [CURSOR_MODE], cx ; save (fixed) cursor mode
4501                                    <1>
4502                                    <1>         ; 27/07/2016
4503 0000179F 803D[7A5B0100]02          <1>         cmp    byte [VGA_MTYPE], 2 ; CTEXT, MTEXT
4504 000017A6 7317                      <1>         jnb    short _sm_17
4505                                    <1>
4506                                    <1>         ; Set cursor shape
4507                                    <1>         ;mov   cx, 0607h
4508                                    <1>         ;call  _set_ctype
4509                                    <1>
4510                                    <1>         ; 29/07/2016
4511 000017A8 B40A                      <1>         mov    ah, 10 ; 6845 register for cursor set
4512 000017AA E8C4050000                <1>         call   m16    ; output cx register
4513                                    <1>
4514                                    <1>         ; 25/07/2016
4515 000017AF 803D[C25E0000]03          <1>           cmp     byte [CRT_MODE], 03h
4516 000017B6 7507                      <1>         jne    short _sm_17
4517                                    <1>         ; 26/07/2016
4518                                    <1>
4519 000017B8 A0[F64D0100]              <1>         mov    al, [ACTIVE_PAGE]
4520 000017BD EB0C                      <1>         jmp    short _sm_18
4521                                    <1> _sm_17:
4522                                    <1>         ; Set cursor pos for page 0..7
4523 000017BF 6629C0                    <1>         sub    ax, ax ; eax = 0
4524 000017C2 BF[E64D0100]              <1>         mov    edi, CURSOR_POSN
4525 000017C7 AB                        <1>         stosd
4526 000017C8 AB                        <1>         stosd
4527 000017C9 AB                        <1>         stosd
4528 000017CA AB                        <1>         stosd
4529                                    <1>         ;; Set active page 0
4530                                    <1>         ;mov   [ACTIVE_PAGE], al ; 0
4531                                    <1> _sm_18:
4532                                    <1>         ; 29/07/2016
4533 000017CB 803D[7A5B0100]02          <1>         cmp    byte [VGA_MTYPE], 2 ; CTEXT, MTEXT
4534 000017D2 0F8386000000              <1>           jnb  _sm_23
4535                                    <1>
4536                                    <1>         ;cmp   byte [CHAR_HEIGHT], 16
4537                                    <1>         ;je    short _sm_19
4538                                    <1>
4539                                    <1>         ;; copy and activate 8x16 font
4540                                    <1>
4541                                    <1>         ; 26/07/2016
4542 000017D8 B004                      <1>         mov    al, 04h
4543                                    <1>         ;sub   bl, bl
4544                                    <1>         ; AX = 1104H ; Load ROM 8x16 Character Set
4545                                    <1>         ; (BL = font block to load (EGA: 0-3; VGA: 0-7))
4546 000017DA E83A150000                <1>         call   load_text_8_16_pat
4547                                    <1>
4548                                    <1>         ; video_func_1103h:
4549                                    <1>         ; biosfn_set_text_block_specifier:
4550                                    <1>         ; BL = font block selector code
4551                                    <1>         ; NOTE: TRDOS 386 only uses and sets font block 0
4552                                    <1>         ; (It is as BL = 0 for TRDOS 386)
4553 000017DF 66BAC403                  <1>         mov    dx, 3C4h ; VGAREG_SEQU_ADDRESS
4554                                    <1>         ;mov   ah, bl
4555 000017E3 28E4                      <1>         sub    ah, ah ; 0
4556 000017E5 B003                      <1>         mov    al, 03h
4557 000017E7 66EF                      <1>         out    dx, ax
4558                                    <1> _sm_19:
4559                                    <1>         ; 29/07/2016
4560                                    <1>         ; 26/07/2016
4561                                    <1>         ; 24/06/2016
4562                                    <1>         ;mov   edi, 0B8000h
4563                                    <1>         ;mov   cx, 4000h ; 16K words (32K)
4564                                    <1>         ;
4565 000017E9 30C0                      <1>         xor    al, al
4566 000017EB 3805[615B0100]            <1>           cmp     byte [p_crt_mode], al ; 0
4567 000017F1 7707                      <1>           ja      short _sm_20 ; 3h, 80h or 83h
4568                                    <1>
4569                                    <1>         ; 30/07/2016
4570                                    <1>         ; 24/06/2016
4571                                    <1>         ; TRDOS 386 (TRDOS v2) 'set mode' modification
4572                                    <1>         ; (for multiscreen feature):
4573                                    <1>         ; If '_set_mode' procedure is called for video mode 3
4574                                    <1>         ;     while video mode is 3, video page will be cleared
4575                                    <1>         ;     and cursor position of video page will be reset.
4576                                    <1>         ; If '_set_mode' procedure is called for a video mode
4577                                    <1>         ;     except video mode 3, while current video mode
4578                                    <1>         ;     is 3; all video pages of mode 3 will be copied
4579                                    <1>         ;     to 98000h address as backup, before mode change.
4580                                    <1>         ; If '_set_mode' procedure is called for video mode 3
4581                                    <1>         ;     while video mode is not 3 and if there is video
4582                                    <1>         ;     page backup for video mode 3, all (of 8) mode 3
4583                                    <1>         ;     video pages will be restored from 98000h.
4584                                    <1>
4585 000017F3 A2[F64D0100]              <1>         mov    [ACTIVE_PAGE], al ; 0
4586                                    <1>         ;mov   ax, 0720h
4587                                    <1>         ;;mov  cx, 4000h ; 16K words (32K)
4588                                    <1>         ;;mov  edi, 0B8000h
```

```
                                <1>       ;rep   stosw
                                <1>       ;sub   al, al
000017F8 EB64                   <1>       jmp    short _sm_23
                                <1> _sm_20:
                                <1>       ; Previous video mode is 3
                                <1>       ; New video mode is 3 while current video mode is not 3
                                <1>       ; (multi screen) video pages will be restored from 0B0000h
                                <1>
000017FA 0FB61D[F64D0100]       <1>       movzx  ebx, byte [ACTIVE_PAGE]
00001801 D0E3                   <1>       shl    bl, 1 ; * 2
00001803 81C3[E64D0100]         <1>       add    ebx, CURSOR_POSN
                                <1>
                                <1>       ; 29/07/2016
00001809 F605[615B0100]7F       <1>       test    byte [p_crt_mode], 7Fh ; 83h or 3h
00001810 7427                   <1>       jz     short _sm_21 ; do not restore video pages
                                <1>
                                <1>       ;; restore video pages
00001812 BE00800900             <1>       mov    esi, 98000h ; 30/07/2016
00001817 BF00800B00             <1>       mov    edi, 0B8000h
0000181C 66B90020               <1>       mov    cx, 2000h ; 8K dwords (32K)
00001820 F3A5                   <1>       rep    movsd
                                <1>
                                <1>       ; restore cursor positions
00001822 BE[665B0100]           <1>       mov    esi, cursor_pposn
00001827 BF[E64D0100]           <1>       mov    edi, CURSOR_POSN
                                <1>       ;mov   ecx, 4 ; restore all cursor positions (16 bytes)
0000182C B104                   <1>       mov    cl, 4
0000182E F3A5                   <1>       rep    movsd
                                <1>
00001830 F605[615B0100]80       <1>         test    byte [p_crt_mode], 80h
00001837 7420                   <1>       jz     short _sm_22 ; do not clear current video pages
                                <1> _sm_21:
                                <1>       ; clear video page
00001839 668B0D[645B0100]       <1>       mov    cx, [CRT_LEN] ; 4096
00001840 66D1E9                 <1>       shr    cx, 1 ; 2072
00001843 66B82007               <1>       mov    ax, 0720h
00001847 BF00800B00             <1>       mov    edi, 0B8000h ; [crt_base]
0000184C 66033D[E44D0100]       <1>       add    di, [CRT_START]
00001853 F366AB                 <1>       rep    stosw  ; FILL THE REGEN BUFFER WITH BLANKS
                                <1>       ;
00001856 66890B                 <1>       mov    [ebx], cx ; reset cursor position
                                <1> _sm_22:
00001859 A2[615B0100]           <1>       mov    [p_crt_mode], al ; 0
                                <1> _sm_23:
                                <1>       ; al = video page number
                                <1>       ; [CRT_LEN] = length of regen buffer in bytes
0000185E E81E010000             <1>       call   _set_active_page
                                <1>
                                <1> ;-----       NORMAL RETURN FROM ALL VIDEO RETURNS
00001863 C3                     <1>       retn
                                <1>
                                <1> cursor_shape_fix:
                                <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
00001864 803D[C65E0000]08       <1>       cmp    byte [CHAR_HEIGHT], 8
0000186B 7649                   <1>       jna    short csf_3
0000186D 80FC08                 <1>       cmp    ah, 8
00001870 7344                   <1>       jnb    short csf_3
00001872 3C20                   <1>       cmp    al, 20h
00001874 7340                   <1>       jnb    short csf_3
                                <1>       ;
00001876 6650                   <1>       push   ax
                                <1>       ; {
                                <1>       ; if(CL!=(CH+1))
00001878 FEC0                   <1>       inc    al
0000187A 38C4                   <1>       cmp    ah, al   ; ah != al + 1
0000187C 740F                   <1>         je      short csf_1
                                <1>       ; CH = ((CH+1) * cheight / 8) -1;
0000187E 8A25[C65E0000]         <1>       mov    ah, [CHAR_HEIGHT]
00001884 F6E4                   <1>       mul    ah
00001886 C0E803                 <1>       shr    al, 3 ; / 8
00001889 FEC8                   <1>       dec    al ; - 1
0000188B EB0E                   <1>       jmp    short csf_2
                                <1> csf_1:
                                <1>       ; }
                                <1>       ; else       ; ah = al + 1
                                <1>       ; {
0000188D FEC4                   <1>       inc    ah     ; ah = ah + 1
                                <1>       ; CH = ((CL+1) * cheight / 8) - 2;
0000188F A0[C65E0000]           <1>       mov    al, [CHAR_HEIGHT]
00001894 F6E4                   <1>       mul    ah
00001896 C0E803                 <1>       shr    al, 3 ; / 8
00001899 2C02                   <1>       sub    al, 2 ; - 2
                                <1>       ; al = 14 (if [CHAR_HEIGHT] = 16)
```

```
4691                                    <1> csf_2:
4692 0000189B 880424                    <1>         mov    [esp], al
4693 0000189E 8A642401                  <1>         mov    ah, [esp+1]
4694                                    <1>         ; CL = ((CL+1) * cheight / 8) - 1;
4695 000018A2 FEC4                      <1>         inc    ah
4696 000018A4 A0[C65E0000]              <1>         mov    al, [CHAR_HEIGHT]
4697 000018A9 F6E4                      <1>         mul    ah
4698 000018AB C0E803                    <1>         shr    al, 3 ; / 8
4699 000018AE FEC8                      <1>         dec    al ; - 1
4700 000018B0 88442401                  <1>         mov    [esp+1], al
4701                                    <1>         ; ah = 15 (if [CHAR_HEIGHT] = 16)
4702                                    <1>         ;
4703 000018B4 6658                      <1>         pop    ax
4704                                    <1> csf_3:
4705 000018B6 C3                        <1>         retn
4706                                    <1>
4707                                    <1> SET_CTYPE:
4708                                    <1>         ; 12/09/2016
4709                                    <1>         ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
4710 000018B7 803D[C25E0000]07          <1>         cmp    byte [CRT_MODE], 7
4711 000018BE 0F8790FCFFFF              <1>         ja     VIDEO_RETURN ; 12/09/2016
4712 000018C4 E805000000                <1>         call   _set_ctype
4713 000018C9 E986FCFFFF                <1>           jmp     VIDEO_RETURN
4714                                    <1>
4715                                    <1> _set_ctype:
4716                                    <1>         ; 02/09/2014 (Retro UNIX 386 v1)
4717                                    <1>         ;
4718                                    <1>         ; VIDEO.ASM - 06/10/85  VIDEO DISPLAY BIOS
4719                                    <1>
4720                                    <1>         ; (CH) = BITS 4-0 = START LINE FOR CURSOR
4721                                    <1>         ;  ** HARDWARE WILL ALWAYS CAUSE BLINK
4722                                    <1>         ;  ** SETTING BIT 5 OR 6 WILL CAUSE ERRATIC BLINKING
4723                                    <1>         ;     OR NO CURSOR AT ALL
4724                                    <1>         ; (CL) = BITS 4-0 = END LINE FOR CURSOR
4725                                    <1>
4726                                    <1> ;-----------------------------------------------
4727                                    <1> ; SET_CTYPE
4728                                    <1> ;     THIS ROUTINE SETS THE CURSOR VALUE
4729                                    <1> ; INPUT
4730                                    <1> ;     (CX) HAS CURSOR VALUE CH-START LINE, CL-STOP LINE
4731                                    <1> ; OUTPUT
4732                                    <1> ;     NONE
4733                                    <1> ;-----------------------------------------------
4734                                    <1>
4735                                    <1>         ; 07/07/2016
4736                                    <1>         ; Fixing cursor start and stop line depending on
4737                                    <1>         ; current character height (=16)
4738                                    <1>         ; (Note: Default/initial values are 6 and 7.
4739                                    <1>         ; If set values are 6 (start) & 7 (stop) and
4740                                    <1>         ; [CHAR_HEIGHT] = 16 :
4741                                    <1>         ; After fixing, start line will be 14, stop line
4742                                    <1>         ; will be 15.)
4743 000018CE 6689C8                    <1>         mov    ax, cx
4744 000018D1 86C4                      <1>         xchg   al, ah
4745                                    <1>         ; AL = start line, AH = stop line
4746 000018D3 E88CFFFFFF                <1>         call   cursor_shape_fix
4747                                    <1>         ; AL = start line (fixed), AH = stop line (fixed)
4748 000018D8 6689C1                    <1>         mov    cx, ax
4749 000018DB 86E9                      <1>         xchg   ch, cl
4750                                    <1>         ; CH = start line (fixed), CL = stop line (fixed)
4751                                    <1>         ;
4752 000018DD B40A                      <1>         mov    ah, 10 ; 6845 register for cursor set
4753 000018DF 66890D[DB5E0000]          <1>         mov    [CURSOR_MODE], cx ; save in data area
4754                                    <1>         ;call  m16    ; output cx register
4755                                    <1>         ;retn
4756 000018E6 E988040000                <1>           jmp     m16
4757                                    <1>
4758                                    <1> SET_CPOS:
4759                                    <1>         ; 12/09/2016
4760                                    <1>         ; 07/07/2016
4761                                    <1>         ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
4762 000018EB 80FF07                    <1>         cmp    bh, 7 ; video page > 7 ; 07/07/2016
4763 000018EE 0F8760FCFFFF              <1>         ja     VIDEO_RETURN
4764                                    <1>         ;
4765 000018F4 803D[C25E0000]07          <1>         cmp    byte [CRT_MODE], 7
4766 000018FB 770A                      <1>         ja     short vga_set_cpos ; 12/09/2016
4767 000018FD E846040000                <1>         call   _set_cpos
4768 00001902 E94DFCFFFF                <1>           jmp     VIDEO_RETURN
4769                                    <1>
4770                                    <1> vga_set_cpos:
4771                                    <1>         ; 12/09/2016
4772                                    <1>         ; 09/07/2016
4773                                    <1>         ; set cursor position
4774                                    <1>         ; NOTE: Hardware cursor position will not be set
4775                                    <1>         ;   in any VGA modes (>7)
4776                                    <1>         ;   But, cursor position will be saved into
4777                                    <1>         ;   [CURSOR_POSN].
4778                                    <1>         ;   TRDOS 386 (TRDOS v2.0) uses only one page
4779                                    <1>         ;   (page 0) for all graphics modes.
4780                                    <1>
4781 00001907 668915[E64D0100]          <1>         mov    [CURSOR_POSN], dx ; save cursor pos for pg 0
4782                                    <1>         ; 04/08/2016
4783                                    <1>         ;mov   bh, [ACTIVE_PAGE] ; = 0
4784                                    <1>         ;call  _set_cpos
4785 0000190E E941FCFFFF                <1>         jmp     VIDEO_RETURN
4786                                    <1>
4787                                    <1> READ_CURSOR:
4788                                    <1>         ; 12/09/2016
4789                                    <1>         ; 07/07/2016
4790                                    <1>         ; 12/05/2016
4791                                    <1>         ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
4792                                    <1>         ;
```

```
4793                                    <1>       ; VIDEO.ASM - 06/10/85  VIDEO DISPLAY BIOS
4794                                    <1>
4795                                    <1> ;--------------------------------------------------
4796                                    <1> ; READ_CURSOR
4797                                    <1> ;     THIS ROUTINE READS THE CURRENT CURSOR VALUE FROM THE
4798                                    <1> ;     845, FORMATS IT, AND SENDS IT BACK TO THE CALLER
4799                                    <1> ; INPUT
4800                                    <1> ;     BH - PAGE OF CURSOR
4801                                    <1> ; OUTPUT
4802                                    <1> ;     DX - ROW, COLUMN OF THE CURRENT CURSOR POSITION
4803                                    <1> ;     CX - CURRENT CURSOR MODE
4804                                    <1> ;--------------------------------------------------
4805                                    <1>
4806                                    <1>       ; BH = Video page number (0 to 7)
4807                                    <1>
4808                                    <1>       ; 07/07/2016
4809 00001913 80FF07                   <1>       cmp    bh, 7 ; video page > 7 (invalid)
4810 00001916 7606                     <1>       jna    short read_cursor_1
4811                                    <1>       ; invalid video page (input)
4812 00001918 31C9                     <1>       xor    ecx, ecx ; 0
4813 0000191A 31D2                     <1>       xor    edx, edx ; 0
4814 0000191C EB15                     <1>       jmp    short read_cursor_2
4815                                    <1> read_cursor_1:
4816                                    <1>       ; 12/09/2016
4817 0000191E 803D[C25E0000]07         <1>       cmp    byte [CRT_MODE], 7 ; vga mode
4818 00001925 7727                     <1>       ja     short vga_get_cpos
4819                                    <1>       ;
4820 00001927 E815000000               <1>       call   get_cpos
4821 0000192C 0FB70D[DB5E0000]         <1>       movzx  ecx, word [CURSOR_MODE]
4822                                    <1> read_cursor_2:
4823 00001933 5D                       <1>       pop    ebp
4824 00001934 5F                       <1>       pop    edi
4825 00001935 5E                       <1>       pop    esi
4826 00001936 5B                       <1>       pop    ebx
4827 00001937 58                       <1>       pop    eax  ; DISCARD SAVED CX AND DX
4828 00001938 58                       <1>       pop    eax
4829 00001939 A1[545B0100]             <1>       mov    eax, [video_eax] ; 12/05/2016
4830                                    <1>       ;;15/01/2017
4831                                    <1>       ;;mov  byte [intflg], 0 ; 07/01/2017
4832 0000193E 1F                       <1>       pop    ds
4833 0000193F 07                       <1>       pop    es
4834 00001940 CF                       <1>       iretd
4835                                    <1>
4836                                    <1> get_cpos:
4837                                    <1>       ; 12/05/2016
4838                                    <1>       ; 16/01/2016
4839                                    <1>       ; BH = Video page number (0 to 7)
4840                                    <1>       ;
4841 00001941 D0E7                     <1>       shl    bh, 1 ; WORD OFFSET
4842 00001943 0FB6F7                   <1>       movzx  esi, bh
4843 00001946 0FB796[E64D0100]         <1>       movzx  edx, word [esi+CURSOR_POSN]
4844 0000194D C3                       <1>       retn
4845                                    <1>
4846                                    <1> vga_get_cpos:
4847                                    <1>       ; 12/09/2016
4848                                    <1>       ; get cursor position (vga)
4849 0000194E 0FB715[E64D0100]         <1>       movzx  edx, word [CURSOR_POSN] ; cursor pos for pg 0
4850 00001955 31C9                     <1>       xor    ecx, ecx ; Cursor Mode = 0 (invalid)
4851 00001957 EBDA                     <1>       jmp     short read_cursor_2
4852                                    <1>
4853                                    <1> ACT_DISP_PAGE:
4854                                    <1>       ; 07/07/2016
4855                                    <1>       ; 26/06/2016
4856                                    <1>       ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
4857                                    <1>       ;
4858                                    <1>       ; VIDEO.ASM - 06/10/85  VIDEO DISPLAY BIOS
4859                                    <1>       ;
4860                                    <1> ;--------------------------------------------------
4861                                    <1> ; ACT_DISP_PAGE
4862                                    <1> ;     THIS ROUTINE SETS THE ACTIVE DISPLAY PAGE, ALLOWING
4863                                    <1> ;     THE FULL USE OF THE MEMORY SET ASIDE FOR THE VIDEO ATTACHMENT
4864                                    <1> ; INPUT
4865                                    <1> ;     AL HAS THE NEW ACTIVE DISPLAY PAGE
4866                                    <1> ; OUTPUT
4867                                    <1> ;     THE 6845 IS RESET TO DISPLAY THAT PAGE
4868                                    <1> ;--------------------------------------------------
4869                                    <1>       ; 07/07/2016
4870 00001959 3C07                     <1>       cmp    al, 7  ; > 7 = invalid video page number
4871 0000195B 0F87F3FBFFFF             <1>       ja     VIDEO_RETURN
4872 00001961 803D[C25E0000]03         <1>         cmp     byte [CRT_MODE], 3
4873 00001968 7408                     <1>       je     short adp_1
4874 0000196A 20C0                     <1>       and    al, al
4875 0000196C 0F85E2FBFFFF             <1>         jnz     VIDEO_RETURN
4876                                    <1>       ;sub   al, al ; 0 ; force to page 0
4877                                    <1> adp_1:
4878 00001972 E805000000               <1>       call   set_active_page
4879 00001977 E9D8FBFFFF               <1>         jmp     VIDEO_RETURN
4880                                    <1>
4881                                    <1> set_active_page:   ; tty_sw
4882                                    <1>       ; 26/07/2016
4883                                    <1>       ; 26/06/2016
4884                                    <1>       ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
4885                                    <1>       ; 30/06/2015
4886                                    <1>       ; 04/03/2014  (act_disp_page --> tty_sw)
4887                                    <1>       ; 10/12/2013
4888                                    <1>       ; 04/12/2013
4889                                    <1>       ;
4890 0000197C A2[F64D0100]             <1>       mov    [ACTIVE_PAGE], al ; save active page value ; [ptty]
4891                                    <1> _set_active_page:
4892                                    <1>       ; 27/06/2015
4893 00001981 0FB6D8                   <1>       movzx  ebx, al
4894                                    <1>       ;
```

```
4895 00001984 6698                <1>       cbw    ; 07/09/2014 (ah=0)
4896 00001986 66F725[645B0100]    <1>       mul    word [CRT_LEN]  ; get saved length of regen buffer
4897                              <1>                              ; display page times regen length
4898                              <1>       ; 10/12/2013
4899 0000198D 66A3[E44D0100]      <1>       mov    [CRT_START], ax ; save start address for later
4900 00001993 6689C1              <1>       mov    cx, ax ; start address to cx
4901                              <1> _M16:
4902                              <1>       ;sar   cx, 1
4903 00001996 66D1E9              <1>       shr    cx, 1  ; divide by 2 for 6845 handling
4904 00001999 B40C                <1>       mov    ah, 12 ; 6845 register for start address
4905 0000199B E8D3030000          <1>       call   m16
4906                              <1>       ;sal   bx, 1
4907                              <1>       ; 01/09/2014
4908 000019A0 D0E3                <1>       shl    bl, 1  ; *2 for word offset
4909 000019A2 81C3[E64D0100]      <1>       add    ebx, CURSOR_POSN
4910 000019A8 668B13              <1>       mov    dx, [ebx] ; get cursor for this page
4911                              <1>       ; 16/01/2016
4912                              <1>       ;call  m18
4913                              <1>       ;retn
4914 000019AB E9AF030000          <1>       jmp    m18
4915                              <1>
4916                              <1> position:
4917                              <1>       ; 24/06/2016
4918                              <1>       ; 12/05/2016 - TRDOS 386 (TRDOS v2.0)
4919                              <1>       ; 27/06/2015
4920                              <1>       ; 02/09/2014
4921                              <1>       ; 30/08/2014 (Retro UNIX 386 v1)
4922                              <1>       ; 04/12/2013 (Retro UNIX 8086 v1)
4923                              <1>       ;
4924                              <1>       ; VIDEO.ASM - 06/10/85  VIDEO DISPLAY BIOS
4925                              <1>       ;
4926                              <1> ;---------------------------------------
4927                              <1> ; POSITION
4928                              <1> ;     THIS SERVICE ROUTINE CALCULATES THE REGEN BUFFER ADDRESS
4929                              <1> ;     OF A CHARACTER IN THE ALPHA MODE
4930                              <1> ; INPUT
4931                              <1> ;     AX = ROW, COLUMN POSITION
4932                              <1> ; OUTPUT
4933                              <1> ;     AX = OFFSET OF CHAR POSITION IN REGEN BUFFER
4934                              <1> ;---------------------------------------
4935                              <1>
4936                              <1>       ; DX = ROW, COLUMN POSITION
4937 000019B0 0FB605[C45E0000]    <1>       movzx  eax, byte [CRT_COLS] ; 24/06/2016
4938 000019B7 F6E6                <1>       mul    dh      ; row value
4939 000019B9 30F6                <1>       xor    dh, dh   ; 0
4940 000019BB 6601D0              <1>       add    ax, dx  ; add column value to the result
4941 000019BE 66D1E0              <1>       shl    ax, 1  ; * 2 for attribute bytes
4942                              <1>              ; EAX = AX = OFFSET OF CHAR POSITION IN REGEN BUFFER
4943 000019C1 C3                  <1>       retn
4944                              <1>
4945                              <1> find_position:
4946                              <1>       ; 24/06/2016
4947                              <1>       ; 12/05/2016 - TRDOS 386 (TRDOS v2.0)
4948                              <1>       ; 27/06/2015
4949                              <1>       ; 07/09/2014
4950                              <1>       ; 02/09/2014
4951                              <1>       ; 30/08/2014 (Retro UNIX 386 v1)
4952                              <1>       ; VIDEO.ASM - 06/10/85  VIDEO DISPLAY BIOS
4953                              <1>
4954 000019C2 0FB6CF              <1>       movzx  ecx, bh ; video page number
4955 000019C5 89CE                <1>       mov    esi, ecx
4956 000019C7 66D1E6              <1>       shl    si, 1
4957 000019CA 668B96[E64D0100]    <1>       mov    dx, [esi+CURSOR_POSN]
4958 000019D1 740C                <1>       jz     short p21
4959 000019D3 6631F6              <1>       xor    si, si
4960                              <1> p20:
4961 000019D6 660335[645B0100]    <1>       add    si, [CRT_LEN] ; 24/06/2016
4962                              <1>       ;add   si, 80*25*2 ; add length of buffer for one page
4963 000019DD E2F7                <1>       loop   p20
4964                              <1> p21:
4965 000019DF 6621D2              <1>       and    dx, dx
4966 000019E2 7407                <1>       jz     short p22
4967 000019E4 E8C7FFFFFF          <1>       call   position ; determine location in regen in page
4968 000019E9 01C6                <1>       add    esi, eax ; add location to start of regen page
4969                              <1> p22:
4970                              <1>       ;mov   dx, [addr_6845] ; get base address of active display
4971                              <1>       ;mov   dx, 03D4h ; I/O address of color card
4972                              <1>       ;add   dx, 6  ; point at status port
4973 000019EB 66BADA03            <1>       mov    dx, 03DAh ; status port
4974                              <1>       ; cx = 0
4975 000019EF C3                  <1>       retn
4976                              <1>
4977                              <1> SCROLL_UP:
4978                              <1>       ; 07/07/2016
4979                              <1>       ; 26/06/2016
4980                              <1>       ; 12/05/2016
4981                              <1>       ; 30/01/2016
4982                              <1>       ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
4983                              <1>       ; 07/09/2014
4984                              <1>       ; 02/09/2014
4985                              <1>       ; 01/09/2014 (Retro UNIX 386 v1 - beginning)
4986                              <1>       ; 04/04/2014
4987                              <1>       ; 04/12/2013
4988                              <1>       ;
4989                              <1>       ; VIDEO.ASM - 06/10/85  VIDEO DISPLAY BIOS
4990                              <1>       ;
4991                              <1> ;---------------------------------------------
4992                              <1> ; SCROLL UP
4993                              <1> ;     THIS ROUTINE MOVES A BLOCK OF CHARACTERS UP
4994                              <1> ;     ON THE SCREEN
4995                              <1> ; INPUT
4996                              <1> ;     (AH) = CURRENT CRT MODE
```

```
                                <1> ;     (AL) = NUMBER OF ROWS TO SCROLL
                                <1> ;     (CX) = ROW/COLUMN OF UPPER LEFT CORNER
                                <1> ;     (DX) = ROW/COLUMN OF LOWER RIGHT CORNER
                                <1> ;     (BH) = ATTRIBUTE TO BE USED ON BLANKED LINE
                                <1> ;     (DS) = DATA SEGMENT
                                <1> ;     (ES) = REGEN BUFFER SEGMENT
                                <1> ; OUTPUT
                                <1> ;     NONE -- THE REGEN BUFFER IS MODIFIED
                                <1> ;-----------------------------------------
                                <1>
                                <1>       ; 07/07/2016
000019F0 38F5                   <1>       cmp   ch, dh
000019F2 0F875CFBFFFF           <1>       ja    VIDEO_RETURN
000019F8 38D1                   <1>       cmp   cl, dl
000019FA 0F8754FBFFFF           <1>       ja    VIDEO_RETURN
                                <1>       ;
00001A00 E805000000             <1>       call  _scroll_up
00001A05 E94AFBFFFF             <1>         jmp     VIDEO_RETURN
                                <1>
                                <1> _scroll_up:  ; from 'write_tty'
                                <1>       ;
                                <1>       ; cl = left upper column
                                <1>       ; ch = left upper row
                                <1>       ; dl = right lower column
                                <1>       ; dh = right lower row
                                <1>       ;
                                <1>       ; al = line count
                                <1>       ; bl = attribute to be used on blanked line
                                <1>       ; bh = video page number (0 to 7)
                                <1>
00001A0A E896000000             <1>       call  test_line_count ; 16/01/2016
                                <1>
00001A0F 8A25[C25E0000]         <1>       mov   ah, [CRT_MODE] ; current video mode
                                <1>       ;cmp  ah, 4
                                <1>       ;jb   short n0
                                <1>       ;cmp  byte [CRT_MODE], 4
00001A15 80FC04                 <1>         cmp ah, 4 ; 07/07/2016
00001A18 0F8320050000           <1>       jnb     GRAPHICS_UP ; 26/06/2016
                                <1>
                                <1>       ;cmp  ah, 7 ; TEST FOR BW CARD
                                <1>       ;jne  GRAPHICS_UP
                                <1> n0:
                                <1>       ; 07/07/2016
00001A1E 80FF07                 <1>       cmp   bh, 7 ; video page number
00001A21 7606                   <1>       jna   short n1
00001A23 8A3D[F64D0100]         <1>       mov   bh, [ACTIVE_PAGE]
                                <1> n1:
00001A29 88DC                   <1>       mov   ah, bl ; attribute
00001A2B 6650                   <1>       push  ax ; *
                                <1>       ;mov  esi, [CRT_BASE]
00001A2D BE00800B00             <1>         mov     esi, 0B8000h
00001A32 3A3D[F64D0100]         <1>         cmp     bh, [ACTIVE_PAGE]
00001A38 750B                   <1>       jne   short n2
                                <1>       ;
00001A3A 66A1[E44D0100]         <1>         mov     ax, [CRT_START]
00001A40 6601C6                 <1>         add     si, ax
00001A43 EB11                   <1>         jmp     short n4
                                <1> n2:
00001A45 20FF                   <1>         and     bh, bh
00001A47 740D                   <1>       jz    short n4
00001A49 88F8                   <1>       mov   al, bh
                                <1> n3:
00001A4B 660335[645B0100]       <1>         add si, [CRT_LEN]
00001A52 FEC8                   <1>         dec al
00001A54 75F5                   <1>       jnz   short n3
                                <1> n4:
00001A56 E85D000000             <1>       call  scroll_position ; 16/01/2016
00001A5B 7420                   <1>         jz      short n6
                                <1>
00001A5D 01CE                   <1>         add     esi, ecx ; from address for scroll
00001A5F 88F5                   <1>       mov   ch, dh  ; #rows in block
00001A61 28C5                   <1>       sub   ch, al ; #rows to be moved
                                <1> n5:
00001A63 E894000000             <1>       call  n10 ; 16/01/2016
                                <1>
00001A68 51                     <1>         push ecx
00001A69 0FB60D[C45E0000]       <1>       movzx ecx, byte [CRT_COLS]
00001A70 00C9                   <1>       add   cl, cl
00001A72 01CE                   <1>         add esi, ecx  ; next line
00001A74 01CF                   <1>         add edi, ecx
00001A76 59                     <1>       pop   ecx
                                <1>
00001A77 FECD                   <1>       dec   ch      ; count of lines to move
00001A79 75E8                   <1>       jnz   short n5 ; row loop
                                <1>       ; ch = 0
00001A7B 88C6                   <1>       mov   dh, al  ; #rows
                                <1> n6:
                                <1>       ; attribute in ah
00001A7D B020                   <1>       mov   al, ' '       ; fill with blanks
                                <1> n7:
00001A7F E885000000             <1>       call  n11 ; 16/01/2016
                                <1>
00001A84 8A0D[C45E0000]         <1>       mov   cl, [CRT_COLS]
00001A8A 00C9                   <1>       add   cl, cl
00001A8C 01CF                   <1>         add edi, ecx
                                <1>
00001A8E FECE                   <1>       dec   dh
00001A90 75ED                   <1>       jnz   short n7
                                <1> n16:
00001A92 3A3D[F64D0100]         <1>       cmp   bh, [ACTIVE_PAGE]
00001A98 750A                   <1>       jne   short n8
                                <1>
```

```
5099                                    <1>       ;cmp   byte [CRT_MODE], 7  ; is this the black and white card
5100                                    <1>       ;je    short n8            ; if so, skip the mode reset
5101                                    <1>
5102 00001A9A A0[C35E0000]              <1>       mov    al, [CRT_MODE_SET] ; get the value of mode set
5103 00001A9F 66BAD803                  <1>       mov    dx, 03D8h ; always set color card port
5104 00001AA3 EE                        <1>       out    dx, al
5105                                    <1> n8:
5106 00001AA4 C3                        <1>       retn
5107                                    <1>
5108                                    <1> test_line_count:
5109                                    <1>       ; 12/05/2016
5110                                    <1>       ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
5111                                    <1>       ; 07/09/2014 (scroll_up)
5112 00001AA5 08C0                      <1>       or     al, al
5113 00001AA7 740E                      <1>       jz     short al_set2
5114 00001AA9 6652                      <1>       push   dx
5115 00001AAB 28EE                      <1>       sub    dh, ch  ; subtract upper row from lower row number
5116 00001AAD FEC6                      <1>       inc    dh     ; adjust difference by 1
5117 00001AAF 38C6                      <1>       cmp    dh, al       ; line count = amount of rows in window?
5118 00001AB1 7502                      <1>       jne    short al_set1 ; if not the we're all set
5119 00001AB3 30C0                      <1>       xor    al, al ; otherwise set al to zero
5120                                    <1> al_set1:
5121 00001AB5 665A                      <1>       pop    dx
5122                                    <1> al_set2:
5123 00001AB7 C3                        <1>       retn
5124                                    <1>
5125                                    <1> scroll_position:
5126                                    <1>       ; 26/06/2016
5127                                    <1>       ; 30/01/2016
5128                                    <1>         ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
5129                                    <1>       ; 07/09/2014 (scroll_up)
5130                                    <1>
5131 00001AB8 6652                      <1>       push   dx
5132 00001ABA 6689CA                    <1>       mov    dx, cx ; now, upper left position in DX
5133 00001ABD E8EEFEFFFF                <1>       call   position
5134 00001AC2 01C6                      <1>       add    esi, eax
5135 00001AC4 89F7                      <1>       mov    edi, esi
5136 00001AC6 665A                      <1>       pop    dx     ; lower right position in DX
5137 00001AC8 6629CA                    <1>       sub    dx, cx
5138 00001ACB FEC6                      <1>       inc    dh     ; dh = #rows
5139 00001ACD FEC2                      <1>       inc    dl     ; dl = #cols in block
5140 00001ACF 59                        <1>       pop    ecx    ; return address
5141 00001AD0 6658                      <1>       pop    ax     ; * ; al = line count, ah = attribute
5142 00001AD2 51                        <1>       push   ecx    ; return address
5143 00001AD3 0FB7C8                    <1>       movzx  ecx, ax
5144 00001AD6 8A25[C45E0000]            <1>       mov    ah, [CRT_COLS]
5145 00001ADC F6E4                      <1>       mul    ah     ; determine offset to from address
5146 00001ADE 6601C0                    <1>       add    ax, ax  ; *2 for attribute byte
5147                                    <1>       ;
5148 00001AE1 6650                      <1>       push   ax     ; offset
5149 00001AE3 6652                      <1>       push   dx
5150                                    <1>       ;
5151                                    <1>       ; 04/04/2014
5152 00001AE5 66BADA03                  <1>       mov    dx, 3DAh ; guaranteed to be color card here
5153                                    <1> n9:                   ; wait_display_enable
5154 00001AE9 EC                        <1>         in      al, dx   ; get port
5155 00001AEA A808                      <1>       test   al, RVRT ; wait for vertical retrace
5156 00001AEC 74FB                      <1>       jz     short n9 ; wait_display_enable
5157 00001AEE B025                      <1>       mov    al, 25h
5158 00001AF0 B2D8                      <1>       mov    dl, 0D8h ; address control port
5159 00001AF2 EE                        <1>       out    dx, al ; turn off video during vertical retrace
5160 00001AF3 665A                      <1>       pop    dx     ; #rows, #cols
5161 00001AF5 6658                      <1>              pop    ax     ; offset
5162 00001AF7 6691                      <1>       xchg   ax, cx ;
5163                                    <1>       ; ecx = offset, al = line count, ah = attribute
5164                                    <1>       ;
5165 00001AF9 08C0                      <1>       or     al, al
5166 00001AFB C3                        <1>       retn
5167                                    <1> n10:
5168                                    <1>       ; Move rows
5169 00001AFC 88D1                      <1>       mov    cl, dl ; get # of cols to move
5170 00001AFE 56                        <1>       push   esi
5171 00001AFF 57                        <1>       push   edi    ; save start address
5172                                    <1> n10r:
5173 00001B00 66A5                      <1>       movsw         ; move that line on screen
5174 00001B02 FEC9                      <1>       dec    cl
5175 00001B04 75FA                      <1>         jnz     short n10r
5176 00001B06 5F                        <1>       pop    edi
5177 00001B07 5E                        <1>       pop    esi    ; recover addresses
5178 00001B08 C3                        <1>       retn
5179                                    <1> n11:
5180                                    <1>       ; Clear rows
5181                                    <1>                     ; dh =  #rows
5182 00001B09 88D1                      <1>         mov  cl, dl ; get # of cols to clear
5183 00001B0B 57                        <1>         push    edi     ; save address
5184                                    <1> n11r:
5185 00001B0C 66AB                      <1>         stosw           ; store fill character
5186 00001B0E FEC9                      <1>       dec    cl
5187 00001B10 75FA                      <1>         jnz     short n11r
5188 00001B12 5F                        <1>         pop     edi     ; recover address
5189 00001B13 C3                        <1>       retn
5190                                    <1>
5191                                    <1> SCROLL_DOWN:
5192                                    <1>       ; 07/07/2016
5193                                    <1>       ; 27/06/2016
5194                                    <1>       ; 26/06/2016
5195                                    <1>       ; 12/05/2016
5196                                    <1>       ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
5197                                    <1>       ;
5198                                    <1>       ; VIDEO.ASM - 06/10/85  VIDEO DISPLAY BIOS
5199                                    <1>
5200                                    <1> ;----------------------------------------
```

```
                                <1> ; SCROLL DOWN
                                <1> ;     THIS ROUTINE MOVES THE CHARACTERS WITHIN A DEFINED
                                <1> ;     BLOCK DOWN ON THE SCREEN, FILLING THE TOP LINES
                                <1> ;     WITH A DEFINED CHARACTER
                                <1> ; INPUT
                                <1> ;     (AH) = CURRENT CRT MODE
                                <1> ;     (AL) = NUMBER OF LINES TO SCROLL
                                <1> ;     (CX) = UPPER LEFT CORNER OF RECION
                                <1> ;     (DX) = LOWER RIGHT CORNER OF REGION
                                <1> ;     (BH) = FILL CHARACTER
                                <1> ;     (DS) = DATA SEGMENT
                                <1> ;     (ES) = REGEN SEGMENT
                                <1> ; OUTPUT
                                <1> ;     NONE -- SCREEN IS SCROLLED
                                <1> ;----------------------------------------
                                <1>
                                <1>       ; 07/07/2016
00001B14 38F5                   <1>       cmp    ch, dh
00001B16 0F8738FAFFFF           <1>       ja     VIDEO_RETURN
00001B1C 38D1                   <1>       cmp    cl, dl
00001B1E 0F8730FAFFFF           <1>       ja     VIDEO_RETURN
                                <1>       ;
00001B24 E805000000             <1>       call   _scroll_down
00001B29 E926FAFFFF             <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>       ; !!!!
00001B2E FD                     <1>       std          ; DIRECTION FOR SCROLL DOWN
                                <1>       ; !!!!
00001B2F E871FFFFFF             <1>       call   test_line_count ; 16/01/2016
                                <1>
00001B34 8A25[C25E0000]         <1>       mov    ah, [CRT_MODE] ; current video mode
                                <1>       ;cmp   ah, 4
                                <1>       ;jb    short _n0
                                <1>       ;cmp   byte [CRT_MODE], 4
00001B3A 80FC04                 <1>         cmp  ah, 4 ; 07/07/2016
00001B3D 0F83DF070000           <1>       jnb     GRAPHICS_DOWN ; 26/06/2016
                                <1>
                                <1>       ;cmp   ah, 7 ; TEST FOR BW CARD
                                <1>       ;jne   GRAPHICS_DOWN
                                <1> _n0:
                                <1>       ; 07/07/2016
00001B43 80FF07                 <1>       cmp    bh, 7 ; video page number
00001B46 7606                   <1>       jna    short n12
00001B48 8A3D[F64D0100]         <1>       mov    bh, [ACTIVE_PAGE]
                                <1>       ;
                                <1> n12:                ; CONTINUE_DOWN
00001B4E 88DC                   <1>       mov    ah, bl
00001B50 6650                   <1>       push   ax     ; * ; save attribute in ah
00001B52 6689D0                 <1>       mov    ax, dx ; LOWER RIGHT CORNER
00001B55 E85EFFFFFF             <1>       call   scroll_position     ; GET REGEN LOCATION
00001B5A 741F                   <1>       jz     short n14
00001B5C 29CE                   <1>       sub    esi, ecx  ; SI IS FROM ADDRESS
00001B5E 88F5                   <1>       mov    ch, dh  ; #rows in block
00001B60 28C5                   <1>       sub    ch, al ; #rows to be moved
                                <1> n13:
00001B62 E895FFFFFF             <1>       call   n10    ; MOVE ONE ROW
                                <1>
00001B67 51                     <1>       push   ecx
00001B68 8A0D[C45E0000]         <1>       mov    cl, [CRT_COLS]
00001B6E 00C9                   <1>       add    cl, cl
00001B70 29CE                   <1>         sub  esi, ecx  ; next line
00001B72 29CF                   <1>         sub  edi, ecx
00001B74 59                     <1>         pop  ecx
                                <1>
00001B75 FECD                   <1>       dec    ch      ; count of lines to move
00001B77 75E9                   <1>       jnz    short n13 ; row loop
                                <1>       ; ch = 0
00001B79 88C6                   <1>       mov    dh, al  ; #rows
                                <1> n14:
                                <1>       ; attribute in ah
00001B7B B020                   <1>       mov    al, ' '       ; fill with blanks
                                <1> n15:
00001B7D E887FFFFFF             <1>       call   n11 ; 16/01/2016
                                <1>
00001B82 8A0D[C45E0000]         <1>       mov    cl, [CRT_COLS]
00001B88 00C9                   <1>       add    cl, cl
00001B8A 29CF                   <1>         sub  edi, ecx
                                <1>
00001B8C FECE                   <1>       dec    dh
00001B8E 75ED                   <1>       jnz    short n15
                                <1>       ;
00001B90 E9FDFEFFFF             <1>       jmp    n16 ; 27/06/2016
                                <1>
                                <1> ;     cmp    bh, [ACTIVE_PAGE]
                                <1> ;     jne    short n16
                                <1> ;
                                <1> ;     ;cmp   byte [CRT_MODE], 7  ; is this the black and white card
                                <1> ;     ;je    short n16           ; if so, skip the mode reset
                                <1> ;
                                <1> ;     mov    al, [CRT_MODE_SET] ; get the value of mode set
                                <1> ;     mov    dx, 03D8h ; always set color card port
```

```
                        <1> ;     out    dx, al
                        <1> ;n16:
                        <1> ;     ; !!!!
                        <1> ;     cld          ; Clear direction flag !
                        <1> ;     ; !!!!
                        <1> ;     retn
                        <1>
                        <1> READ_AC_CURRENT:
                        <1>       ; 08/07/2016
                        <1>       ; 26/06/2016
                        <1>       ; 12/05/2016
                        <1>       ; 18/01/2016
                        <1>       ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
                        <1>       ;
                        <1>       ; VIDEO.ASM - 06/10/85  VIDEO DISPLAY BIOS
                        <1>       ;
                        <1>       ; 08/07/2016
00001B95 803D[C25E0000]07 <1>       cmp     byte [CRT_MODE], 7 ; 6!?
00001B9C 7607           <1>       jna    short read_ac_c
00001B9E 31C0           <1>       xor    eax, eax
00001BA0 E9B4F9FFFF     <1>       jmp     _video_return
                        <1> read_ac_c:
00001BA5 E805000000     <1>       call   _read_ac_current
                        <1>       ; 12/05/2016
                        <1>       ;jmp     VIDEO_RETURN
00001BAA E9AAF9FFFF     <1>       jmp    _video_return
                        <1>
                        <1> ;--------------------------------------------------------------------
                        <1> ; READ_AC_CURRENT                                          :
                        <1> ;     THIS ROUTINE READS THE ATTRIBUTE AND CHARACTER AT THE CURRENT     :
                        <1> ;     CURSOR POSITION AND RETURNS THEM TO THE CALLER                    :
                        <1> ; INPUT                                                          :
                        <1> ;     (AH) = CURRENT CRT MODE                                    :
                        <1> ;     (BH) = DISPLAY PAGE ( ALPHA MODES ONLY )                   :
                        <1> ;     (DS) = DATA SEGMENT                                  :
                        <1> ;     (ES) = REGEN SEGMENT                                       :
                        <1> ; OUTPUT                                                         :
                        <1> ;     (AL) = CHARACTER READ                                      :
                        <1> ;     (AH) = ATTRIBUTE READ                                      :
                        <1> ;--------------------------------------------------------------------
                        <1>
                        <1> _read_ac_current:
                        <1>       ; 26/06/2016
                        <1>       ; 12/05/2016
                        <1>       ; 18/01/2016
                        <1>
                        <1>       ;mov   ah, [CRT_MODE] ; current video mode
                        <1>       ;cmp   ah, 4
                        <1>       ;jb    short p10
00001BAF 803D[C25E0000]04 <1>       cmp    byte [CRT_MODE], 4
00001BB6 0F83BB080000   <1>       jnb     GRAPHICS_READ ; 26/06/2016
                        <1>
                        <1>       ;cmp   ah, 7 ; TEST FOR BW CARD
                        <1>       ;jne   GRAPHICS_READ
                        <1> p10:
00001BBC E801FEFFFF     <1>       call   find_position; GET REGEN LOCATION AND PORT ADDRESS
                        <1>       ;
                        <1>       ; esi = regen location
                        <1>       ; dx = status port
                        <1>       ;
00001BC1 8A25[C25E0000]   <1>       mov    ah, [CRT_MODE]
00001BC7 80EC02         <1>       sub    ah, 2
00001BCA D0EC           <1>       shr    ah, 1
00001BCC 7515           <1>       jnz    short p13
                        <1>
                        <1>       ; WAIT FOR HORIZONTAL RETRACE OR VERTICAL RETRACE IF COLOR 80
                        <1> p11:
00001BCE FB             <1>       sti          ; enable interrupts first
00001BCF 3A3D[F64D0100]   <1>       cmp     bh, [ACTIVE_PAGE]
00001BD5 750C           <1>       jne    short p13
00001BD7 FA             <1>       cli          ; block interrupts for single loop
00001BD8 EC             <1>       in     al, dx ; get status from the adapter
00001BD9 A801           <1>       test   al, RHRZ ; is horizontal retrace low
00001BDB 75F1           <1>       jnz    short p11 ; wait until it is
                        <1> p12:                ;  wait for either retrace high
00001BDD EC             <1>       in     al, dx ; get status again
00001BDE A809           <1>       test   al, RVRT+RHRZ ; is horizontal or vertical retrace high
00001BE0 74FB           <1>       jz     short p12 ; wait until either retrace active
00001BE2 FB             <1>       sti
                        <1> p13:
00001BE3 81C600800B00     <1>       add    esi, 0B8000h
00001BE9 668B06         <1>       mov    ax, [esi]
                        <1>
00001BEC C3             <1>       retn   ; 18/01/2016
                        <1>
                        <1> WRITE_AC_CURRENT:
                        <1>       ; 08/07/2016
                        <1>       ; 26/06/2016
                        <1>       ; 24/06/2016
                        <1>       ; 12/05/2016
                        <1>       ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
                        <1>       ;
                        <1>       ; VIDEO.ASM - 06/10/85  VIDEO DISPLAY BIOS
                        <1>       ;
                        <1> ;--------------------------------------------------------------
                        <1> ; WRITE_AC_CURRENT                                    :
                        <1> ;     THTS ROUTINE WRITES THE ATTRIBUTE AND CHARACTER           :
                        <1> ;     AT THE CURRENT CURSOR POSITION                       :
                        <1> ; INPUT                                                    :
                        <1> ;     (AH) = CURRENT CRT MODE                              :
                        <1> ;     (BH) = DISPLAY PAGE                            :
                        <1> ;     (CX) = COUNT OF CHARACTERS TO WRITE                  :
```

```
                                <1> ;     (AL) = CHAR TO WRITE                                  :
                                <1> ;     (BL) = ATTRIBUTE OF CHAR TO WRITE                     :
                                <1> ;     (DS) = DATA SEGMENT                                :
                                <1> ;     (ES) = REGEN SEGMENT                                  :
                                <1> ; OUTPUT                                                 :
                                <1> ;     DISPLAY REGEN BUFFER UPDATED                          :
                                <1> ;-----------------------------------------------------------------
                                <1>
                                <1>       ; 08/07/2016
00001BED 803D[C25E0000]07       <1>       cmp    byte [CRT_MODE], 7 ; 6!?
00001BF4 760A                   <1>       jna    short write_ac_c
                                <1>
00001BF6 E8F20A0000             <1>       call   vga_write_char_attr
00001BFB E954F9FFFF             <1>       jmp     VIDEO_RETURN
                                <1>
                                <1> write_ac_c:
00001C00 E834000000             <1>       call   _write_c_current
                                <1>
00001C05 0FB6F7                 <1>       movzx  esi, bh ; video page number (0 to 7)
00001C08 889E[CB5E0000]         <1>       mov    [esi+chr_attrib], bl ; color/attribute
                                <1>
00001C0E E941F9FFFF             <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
00001C13 803D[C25E0000]07       <1>       cmp    byte [CRT_MODE], 7 ; 6!?
00001C1A 760A                   <1>       jna    short write_c_c
                                <1>
00001C1C E8CC0A0000             <1>       call   vga_write_char_only
00001C21 E92EF9FFFF             <1>       jmp     VIDEO_RETURN
                                <1>
                                <1> write_c_c:
                                <1>       ;and   bh, 7 ; video page number (<= 7)
00001C26 0FB6F7                 <1>       movzx  esi, bh
00001C29 8A9E[CB5E0000]         <1>       mov    bl, [esi+chr_attrib]
                                <1>
00001C2F E805000000             <1>       call   _write_c_current
00001C34 E91BF9FFFF             <1>         jmp     VIDEO_RETURN
                                <1>
                                <1> _write_c_current:  ; from 'write_tty'
                                <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>       ;mov   ah, [CRT_MODE] ; current video mode
                                <1>       ;cmp   ah, 4
                                <1>       ;jb    short p40
00001C39 803D[C25E0000]04       <1>       cmp    byte [CRT_MODE], 4
00001C40 0F8381070000           <1>         jnb     GRAPHICS_WRITE ; 26/06/2016
                                <1>
                                <1>       ;cmp   ah, 7 ; TEST FOR BW CARD
                                <1>       ;jne   GRAPHICS_WRITE
                                <1> p40:
                                <1>       ; al = character
                                <1>       ; bl = color/attribute
                                <1>       ; bh = video page
                                <1>       ; cx = count of characters to write
00001C46 6652                   <1>       push   dx
00001C48 88DC                   <1>       mov    ah, bl  ; color/attribute (12/05/2016)
00001C4A 6650                   <1>       push   ax     ; save character & attribute/color
00001C4C 6651                   <1>       push   cx
00001C4E E86FFDFFFF             <1>       call   find_position  ; get regen location and port address
00001C53 6659                   <1>       pop    cx
                                <1>       ; esi = regen location
                                <1>       ; dx = status port
                                <1>       ;
00001C55 81C600800B00           <1>       add    esi, 0B8000h ; 30/08/2014 (crt_base)
                                <1>       ;
00001C5B 8A25[C25E0000]         <1>       mov    ah, [CRT_MODE]
00001C61 80EC02                 <1>       sub    ah, 2
00001C64 D0EC                   <1>       shr    ah, 1
00001C66 7519                   <1>       jnz    short p44    ; 26/06/2016
                                <1>
                                <1>       ; WAIT FOR HORIZONTAL RETRACE OR VERTICAL RETRACE IF COLOR 80
```

```
5507                                    <1> p41:
5508 00001C68 FB                        <1>         sti             ; enable interrupts first
5509 00001C69 3A3D[F64D0100]            <1>         cmp     bh, [ACTIVE_PAGE]
5510 00001C6F 7510                      <1>         jne     short p44
5511 00001C71 FA                        <1>         cli             ; block interrupts for single loop
5512 00001C72 EC                        <1>         in      al, dx ; get status from the adapter
5513 00001C73 A808                      <1>         test    al, RVRT ; check for vertical retrace first
5514 00001C75 7509                      <1>         jnz     short p43 ; Do fast write now if vertical retrace
5515 00001C77 A801                      <1>         test    al, RHRZ ; is horizontal retrace low
5516 00001C79 75ED                      <1>         jnz     short p41 ; wait until it is
5517                                    <1> p42:                ;  wait for either retrace high
5518 00001C7B EC                        <1>         in      al, dx ; get status again
5519 00001C7C A809                      <1>         test    al, RVRT+RHRZ ; is horizontal or vertical retrace high
5520 00001C7E 74FB                      <1>         jz      short p42 ; wait until either retrace active
5521                                    <1> p43:
5522 00001C80 FB                        <1>         sti
5523                                    <1> p44:
5524 00001C81 668B0424                  <1>         mov     ax, [esp] ; restore the character (al) & attribute (ah)
5525 00001C85 668906                    <1>         mov     [esi], ax
5526                                    <1>
5527 00001C88 6649                      <1>         dec     cx
5528 00001C8A 7404                      <1>         jz      short p45
5529                                    <1>
5530 00001C8C 46                        <1>         inc     esi
5531 00001C8D 46                        <1>         inc     esi
5532 00001C8E EBD8                      <1>         jmp     short p41
5533                                    <1> p45:
5534 00001C90 6658                      <1>         pop     ax
5535 00001C92 665A                      <1>         pop     dx
5536 00001C94 C3                        <1>         retn
5537                                    <1>
5538                                    <1> ; 09/07/2016
5539                                    <1> ; 26/06/2016
5540                                    <1> ; 24/06/2016
5541                                    <1> ; 12/05/2016
5542                                    <1> ; 18/01/2016
5543                                    <1> ; 16/01/2016 - TRDOS 386 (TRDOS v2.0)
5544                                    <1> ; 30/06/2015
5545                                    <1> ; 27/06/2015
5546                                    <1> ; 11/03/2015
5547                                    <1> ; 02/09/2014
5548                                    <1> ; 30/08/2014
5549                                    <1> ; VIDEO FUNCTIONS
5550                                    <1> ; (write_tty - Retro UNIX 8086 v1 - U9.ASM, 01/02/2014)
5551                                    <1>
5552                                    <1> WRITE_TTY:
5553                                    <1>         ; 09/07/2016
5554                                    <1>         ; 01/07/2016
5555                                    <1>         ; 26/06/2016
5556                                    <1>         ; 24/06/2016
5557                                    <1>         ; 13/05/2016
5558                                    <1>         ; 12/05/2016
5559                                    <1>         ; 30/01/2016
5560                                    <1>         ; 18/01/2016
5561                                    <1>         ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
5562                                    <1>         ; 13/08/2015
5563                                    <1>         ; 02/09/2014
5564                                    <1>         ; 30/08/2014 (Retro UNIX 386 v1 - beginning)
5565                                    <1>         ; 01/02/2014 (Retro UNIX 8086 v1 - last update)
5566                                    <1>         ; 03/12/2013 (Retro UNIX 8086 v1 - beginning)
5567                                    <1>         ; (Modified registers: EAX, EBX, ECX, EDX, ESI, EDI)
5568                                    <1>         ;
5569                                    <1>         ; INPUT -> AL = Character to be written
5570                                    <1>         ;          BL = Color (Forecolor, Backcolor)
5571                                    <1>         ;          BH = Video Page (0 to 7)
5572                                    <1>
5573                                    <1>         ; 09/07/2016
5574 00001C95 803D[C25E0000]07          <1>           cmp     byte [CRT_MODE], 7
5575 00001C9C 760A                      <1>         jna     short write_tty_cga
5576                                    <1>
5577 00001C9E E8290D0000                <1>         call    vga_write_teletype
5578 00001CA3 E9ACF8FFFF                <1>         jmp     VIDEO_RETURN
5579                                    <1>
5580                                    <1> write_tty_cga:
5581                                    <1>         ; 13/05/2016
5582                                    <1>         ;call _write_tty
5583                                    <1>         ; 01/07/2016
5584 00001CA8 E818000000                <1>         call    _write_tty_m3
5585 00001CAD E9A2F8FFFF                <1>         jmp     VIDEO_RETURN
5586                                    <1>
5587                                    <1> RVRT  equ   00001000b    ; VIDEO VERTICAL RETRACE BIT
5588                                    <1> RHRZ  equ   00000001b    ; VIDEO HORIZONTAL RETRACE BIT
5589                                    <1>
5590                                    <1> ; Derived from "WRITE_TTY" procedure of IBM "pc-at" rombios source code
5591                                    <1> ; (06/10/1985), 'video.asm', INT 10H, VIDEO_IO
5592                                    <1> ;
5593                                    <1> ; 06/10/85  VIDEO DISPLAY BIOS
5594                                    <1> ;
5595                                    <1> ;--- WRITE_TTY ----------------------------------------------------------------
5596                                    <1> ;                                                                :
5597                                    <1> ;   THIS INTERFACE PROVIDES A TELETYPE LIKE INTERFACE TO THE                 :
5598                                    <1> ;   VIDEO CARDS. THE INPUT CHARACTER IS WRITTEN TO THE CURRENT               :
5599                                    <1> ;   CURSOR POSITION, AND THE CURSOR IS MOVED TO THE NEXT POSITION.          :
5600                                    <1> ;   IF THE CURSOR LEAVES THE LAST COLUMN OF THE FIELD, THE COLUMN           :
5601                                    <1> ;   IS SET TO ZERO, AND THE ROW VALUE IS INCREMENTED. IF THE ROW       :
5602                                    <1> ;   ROW VALUE LEAVES THE FIELD, THE CURSOR IS PLACED ON THE LAST ROW,        :
5603                                    <1> ;   FIRST COLUMN, AND THE ENTIRE SCREEN IS SCROLLED UP ONE LINE.       :
5604                                    <1> ;   WHEN THE SCREEN IS SCROLLED UP, THE ATTRIBUTE FOR FILLING THE            :
5605                                    <1> ;   NEWLY BLANKED LINE IS READ FROM THE CURSOR POSITION ON THE PREVIOUS      :
5606                                    <1> ;   LINE BEFORE THE SCROLL, IN CHARACTER MODE. IN GRAPHICS MODE,       :
5607                                    <1> ;   THE 0 COLOR IS USED.                                               :
5608                                    <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> ; 08/07/2016
                        <1> ; 26/06/2016
                        <1> ; 24/06/2016
                        <1> _write_tty:  ; 13/05/2016
00001CB2 FA             <1>       cli
                        <1>       ;
                        <1>       ; 01/09/2014
00001CB3 803D[C25E0000]03 <1>     cmp    byte [CRT_MODE], 3
00001CBA 7409           <1>       je     short _write_tty_m3
                        <1>       ;
                        <1> set_mode_3:
00001CBC 53             <1>       push   ebx
00001CBD 50             <1>       push   eax
00001CBE E8A2F8FFFF     <1>       call   _set_mode
00001CC3 58             <1>       pop    eax
00001CC4 5B             <1>       pop    ebx
                        <1>       ;
                        <1> _write_tty_m3: ; 24/06/2016 (m3 -> _write_tty_m3)
00001CC5 0FB6F7         <1>       movzx  esi, bh ; 12/05/2016
00001CC8 66D1E6         <1>       shl    si, 1
00001CCB 81C6[E64D0100] <1>       add    esi, CURSOR_POSN
00001CD1 668B16         <1>       mov    dx, [esi]
                        <1>       ;
                        <1>       ; dx now has the current cursor position
                        <1>       ;
00001CD4 3C0D           <1>       cmp    al, 0Dh      ; CR   ; is it carriage return or control character
00001CD6 7636           <1>       jbe    short u8
                        <1>       ;
                        <1>       ; write the char to the screen
                        <1> u0:
                        <1>       ; al = character
                        <1>       ; bl = attribute/color
                        <1>       ; bh = video page number (0 to 7)
                        <1>       ;
00001CD8 66B90100       <1>       mov    cx, 1  ; 24/06/2016
                        <1>       ; cx = count of characters to write
                        <1>       ;
00001CDC E858FFFFFF     <1>       call   _write_c_current ; 16/01/2015
                        <1>       ;
                        <1>       ; position the cursor for next char
00001CE1 FEC2           <1>       inc    dl           ; next column
00001CE3 3A15[C45E0000] <1>       cmp    dl, [CRT_COLS]  ; test for column overflow
00001CE9 755D           <1>         jne     _set_cpos
00001CEB B200           <1>       mov    dl, 0        ; column = 0
                        <1> u10:                      ; (line feed found)
00001CED 80FE18         <1>       cmp    dh, 25-1     ; check for last row
00001CF0 7218           <1>       jb     short u6
                        <1>       ;
                        <1>       ; scroll required
                        <1> u1:
                        <1>       ; SET CURSOR POSITION (04/12/2013)
00001CF2 E851000000     <1>       call   _set_cpos
                        <1>       ;
                        <1>       ; determine value to fill with during scroll
                        <1> u2:
                        <1>       ; bh = video page number
                        <1>       ;
00001CF7 E8B3FEFFFF     <1>       call   _read_ac_current ; 18/01/2016
                        <1>       ;
                        <1>       ; al = character, ah = attribute
                        <1>       ; bh = video page number
                        <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
00001CFC 6629C9         <1>       sub    cx, cx
00001CFF 66BA4F18       <1>       mov    dx, 184Fh ; dl = 79 (column), dh = 24 (row)
                        <1>       ;
00001D03 B001           <1>       mov    al, 1        ; scroll 1 line up
                        <1>              ; ah = attribute
                        <1>       ;mov   bl, al ; 12/05/2016
00001D05 E900FDFFFF     <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
00001D0A 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
00001D0C EB3A           <1>       jmp     _set_cpos
                        <1>
                        <1>       ; check for control characters
                        <1> u8:
00001D0E 7436           <1>       je     short u9
00001D10 3C0A           <1>       cmp    al, 0Ah            ; is it a line feed (0Ah)
00001D12 74D9           <1>       je     short u10
00001D14 3C07           <1>       cmp    al, 07h      ; is it a bell
00001D16 747A           <1>       je     short u11
00001D18 3C08           <1>       cmp    al, 08h            ; is it a backspace
                        <1>       ;jne   short u0
00001D1A 7422           <1>       je     short bs     ; 12/12/2013
                        <1>       ; 12/12/2013 (tab stop)
00001D1C 3C09           <1>       cmp    al, 09h            ; is it a tab stop
00001D1E 75B8           <1>       jne    short u0
00001D20 88D0           <1>       mov    al, dl
00001D22 6698           <1>       cbw
00001D24 B108           <1>       mov    cl, 8
00001D26 F6F1           <1>       div    cl
00001D28 28E1           <1>       sub    cl, ah
                        <1> ts:
                        <1>       ; 02/09/2014
                        <1>       ; 01/09/2014
00001D2A B020           <1>       mov    al, 20h
                        <1> tsloop:
00001D2C 6651           <1>       push   cx
00001D2E 6650           <1>       push   ax
                        <1>       ;mov   bh, [ACTIVE_PAGE]
00001D30 E890FFFFFF     <1>       call   _write_tty_m3 ; 24/06/2016 (m3 -> _write_tty_m3)
00001D35 6658           <1>       pop    ax  ; ah = attribute/color
00001D37 6659           <1>       pop    cx
00001D39 FEC9           <1>       dec    cl
00001D3B 75EF           <1>       jnz    short tsloop
00001D3D C3             <1>       retn
                        <1> bs:
                        <1>       ; back space found
                        <1>
00001D3E 08D2           <1>       or     dl, dl             ; is it already at start of line
                        <1>       ;je    short u7     ; set_cursor
00001D40 7406           <1>       jz     short _set_cpos
00001D42 664A           <1>       dec    dx                 ; no -- just move it back
                        <1>       ;jmp   short u7
00001D44 EB02           <1>       jmp    short _set_cpos
                        <1>
                        <1>       ; carriage return found
                        <1> u9:
00001D46 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>       ; 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>       ;
00001D48 BE[E64D0100]   <1>       mov    esi, CURSOR_POSN
00001D4D 0FB6C7         <1>         movzx   eax, bh   ; BH = video page number
                        <1> ;     or     al, al
                        <1> ;     jz     short _set_cpos_0
00001D50 D0E0           <1>         shl     al, 1   ; word offset
00001D52 01C6           <1>         add     esi, eax
                        <1> ;_set_cpos_0:
00001D54 668916         <1>       mov    [esi], dx ; save the pointer
00001D57 383D[F64D0100] <1>       cmp    [ACTIVE_PAGE], bh
00001D5D 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:
00001D5F E84CFCFFFF     <1>       call   position ; determine location in regen buffer
00001D64 668B0D[E44D0100] <1>     mov    cx, [CRT_START]
00001D6B 6601C1         <1>       add    cx, ax  ; add char position in regen buffer
                        <1>                     ; to the start address (offset) for this page
```

```
5813 00001D6E 66D1E9              <1>       shr    cx, 1  ; divide by 2 for char only count
5814 00001D71 B40E                <1>       mov    ah, 14 ; register number for cursor
5815                              <1>       ;call  m16    ; output value to the 6845
5816                              <1>       ;retn
5817                              <1>
5818                              <1>       ;-----THIS ROUTINE OUTPUTS THE CX REGISTER
5819                              <1>       ;     TO THE 6845 REGISTERS NAMED IN (AH)
5820                              <1> m16:
5821 00001D73 FA                  <1>       cli
5822                              <1>       ;mov   dx, [addr_6845] ; address register
5823 00001D74 66BAD403            <1>       mov    dx, 03D4h ; I/O address of color card
5824 00001D78 88E0                <1>       mov    al, ah ; get value
5825 00001D7A EE                  <1>       out    dx, al ; register set
5826 00001D7B 6642                <1>       inc    dx     ; data register
5827 00001D7D EB00                <1>       jmp    $+2    ; i/o delay
5828 00001D7F 88E8                <1>       mov    al, ch ; data
5829 00001D81 EE                  <1>       out    dx, al
5830 00001D82 664A                <1>       dec    dx
5831 00001D84 88E0                <1>       mov    al, ah
5832 00001D86 FEC0                <1>       inc    al     ; point to other data register
5833 00001D88 EE                  <1>       out    dx, al ; set for second register
5834 00001D89 6642                <1>       inc    dx
5835 00001D8B EB00                <1>       jmp    $+2    ; i/o delay
5836 00001D8D 88C8                <1>       mov    al, cl ; second data value
5837 00001D8F EE                  <1>       out    dx, al
5838 00001D90 FB                  <1>       sti
5839                              <1> m17:
5840 00001D91 C3                  <1>       retn
5841                              <1>
5842                              <1> beeper:
5843                              <1>       ; 04/08/2016
5844                              <1>       ; 12/05/2016 - TRDOS 386 (TRDOS v2.0)
5845                              <1>       ; 30/08/2014 (Retro UNIX 386 v1)
5846                              <1>       ; 18/01/2014
5847                              <1>       ; 03/12/2013
5848                              <1>       ; bell found
5849                              <1> u11:
5850 00001D92 FB                  <1>       sti
5851 00001D93 3A3D[F64D0100]      <1>       cmp    bh, [ACTIVE_PAGE]
5852 00001D99 7551                <1>       jne    short u12    ; Do not sound the beep
5853                              <1>                           ; if it is not written on the active page
5854                              <1> beeper_gfx: ; 04/08/2016
5855 00001D9B 66B93305            <1>       mov    cx, 1331     ; divisor for 896 hz tone
5856 00001D9F B31F                <1>       mov    bl, 31       ; set count for 31/64 second for beep
5857                              <1>       ;call  beep         ; sound the pod bell
5858                              <1>       ;jmp   short u5     ; tty_return
5859                              <1>       ;retn
5860                              <1>
5861                              <1> TIMER equ    040h               ; 8254 TIMER - BASE ADDRESS
5862                              <1> PORT_B       equ    061h        ; PORT B READ/WRITE DIAGNOSTIC REGISTER
5863                              <1> GATE2 equ    00000001b    ; TIMER 2 INPUT CATE CLOCK BIT
5864                              <1> SPK2  equ    00000010b    ; SPEAKER OUTPUT DATA ENABLE BIT
5865                              <1>
5866                              <1> beep:
5867                              <1>       ; 07/02/2015
5868                              <1>       ; 30/08/2014 (Retro UNIX 386 v1)
5869                              <1>       ; 18/01/2014
5870                              <1>       ; 03/12/2013
5871                              <1>       ;
5872                              <1>       ; TEST4.ASM - 06/10/85  POST AND BIOS UTILITY ROUTINES
5873                              <1>       ;
5874                              <1>       ; ROUTINE TO SOUND THE BEEPER USING TIMER 2 FOR TONE
5875                              <1>       ;
5876                              <1>       ; ENTRY:
5877                              <1>       ;    (BL) = DURATION COUNTER ( 1 FOR 1/64 SECOND )
5878                              <1>       ;    (CX) = FREQUENCY DIVISOR (1193180/FREQUENCY) (1331 FOR 886 HZ)
5879                              <1>       ; EXIT:                          :
5880                              <1>       ;    (AX),(BL),(CX) MODIFIED.
5881                              <1>
5882 00001DA1 9C                  <1>       pushf  ; 18/01/2014 ; save interrupt status
5883 00001DA2 FA                  <1>       cli                 ; block interrupts during update
5884 00001DA3 B0B6                <1>       mov    al, 10110110b; select timer 2, lsb, msb binary
5885 00001DA5 E643                <1>       out    TIMER+3, al  ; write timer mode register
5886 00001DA7 EB00                <1>       jmp    $+2          ; I/O delay
5887 00001DA9 88C8                <1>       mov    al, cl       ; divisor for hz (low)
5888 00001DAB E642                <1>       out    TIMER+2,AL   ; write timer 2 count - lsb
5889 00001DAD EB00                <1>       jmp    $+2          ; I/O delay
5890 00001DAF 88E8                <1>       mov    al, ch       ; divisor for hz (high)
5891 00001DB1 E642                <1>       out    TIMER+2, al  ; write timer 2 count - msb
5892 00001DB3 E461                <1>       in     al, PORT_B   ; get current setting of port
5893 00001DB5 88C4                <1>       mov    ah, al       ; save that setting
5894 00001DB7 0C03                <1>       or     al, GATE2+SPK2      ; gate timer 2 and turn speaker on
5895 00001DB9 E661                <1>       out    PORT_B, al   ; and restore interrupt status
5896                              <1>       ;popf  ; 18/01/2014
5897 00001DBB FB                  <1>       sti
5898                              <1> g7:                        ; 1/64 second per count (bl)
5899 00001DBC B90B040000          <1>       mov    ecx, 1035    ; delay count for 1/64 of a second
5900 00001DC1 E827000000          <1>       call   waitf        ; go to beep delay 1/64 count
5901 00001DC6 FECB                <1>       dec    bl           ; (bl) length count expired?
5902 00001DC8 75F2                <1>       jnz    short g7     ; no - continue beeping speaker
5903                              <1>       ;
5904                              <1>       ;pushf              ; save interrupt status
5905 00001DCA FA                  <1>       cli    ; 18/01/2014 ; block interrupts during update
5906 00001DCB E461                <1>       in     al, PORT_B   ; get current port value
5907                              <1>         ;or      al, not (GATE2+SPK2) ; isolate current speaker bits in case
5908 00001DCD 0CFC                <1>         or      al, ~(GATE2+SPK2)
5909 00001DCF 20C4                <1>         and  ah, al       ; someone turned them off during beep
5910 00001DD1 88E0                <1>       mov    al, ah       ; recover value of port
5911                              <1>         ;or      al, not (GATE2+SPK2) ; force speaker data off
5912 00001DD3 0CFC                <1>       or     al, ~(GATE2+SPK2) ; isolate current speaker bits in case
5913 00001DD5 E661                <1>       out    PORT_B, al   ; and stop speaker timer
5914                              <1>       ;popf               ; restore interrupt flag state
```

```
5915 00001DD7 FB                <1>       sti
5916 00001DD8 B90B040000        <1>       mov    ecx, 1035    ; force 1/64 second delay (short)
5917 00001DDD E80B000000        <1>       call   waitf        ; minimum delay between all beeps
5918                            <1>       ;pushf              ; save interrupt status
5919 00001DE2 FA                <1>       cli                 ; block interrupts during update
5920 00001DE3 E461              <1>       in     al, PORT_B   ; get current port value in case
5921 00001DE5 2403              <1>       and    al, GATE2+SPK2      ; someone turned them on
5922 00001DE7 08E0              <1>       or     al, ah       ; recover value of port_b
5923 00001DE9 E661              <1>       out    PORT_B, al   ; restore speaker status
5924 00001DEB 9D                <1>       popf                ; restore interrupt flag state
5925                            <1> u12:
5926 00001DEC C3                <1>       retn
5927                            <1>
5928                            <1> REFRESH_BIT equ    00010000b    ; REFRESH TEST BIT
5929                            <1>
5930                            <1> WAITF:
5931                            <1> waitf:
5932                            <1>       ; 30/08/2014 (Retro UNIX 386 v1)
5933                            <1>       ; 03/12/2013
5934                            <1>       ;
5935                            <1> ;     push ax                   ; save work register (ah)
5936                            <1> ;waitf1:
5937                            <1>                                 ; use timer 1 output bits
5938                            <1> ;     in     al, PORT_B   ; read current counter output status
5939                            <1> ;     and    al, REFRESH_BIT     ; mask for refresh determine bit
5940                            <1> ;     cmp    al, ah       ; did it just change
5941                            <1> ;     je     short waitf1 ; wait for a change in output line
5942                            <1> ;     ;
5943                            <1> ;     mov    ah, al       ; save new lflag state
5944                            <1> ;     loop   waitf1       ; decrement half cycles till count end
5945                            <1> ;     ;
5946                            <1> ;     pop    ax           ; restore (ah)
5947                            <1> ;     retn                ; return (cx)=0
5948                            <1>
5949                            <1> ; 06/02/2015 (unix386.s <-- dsectrm2.s)
5950                            <1> ; 17/12/2014 (dsectrm2.s)
5951                            <1> ; WAITF
5952                            <1> ; /// IBM PC-XT Model 286 System BIOS Source Code - Test 4 - 06/10/85 ///
5953                            <1> ;
5954                            <1> ;---WAITF--------------------------------------------------------------------------
5955                            <1> ;     FIXED TIME WAIT ROUTINE (HARDWARE CONTROLLED - NOT PROCESSOR)
5956                            <1> ; ENTRY:
5957                            <1> ;     (CX) = COUNT OF 15.085737 MICROSECOND INTERVALS TO WAIT
5958                            <1> ;            MEMORY REFRESH TIMER 1 OUTPUT USED AS REFERENCE
5959                            <1> ; EXIT:
5960                            <1> ;                    AFTER (CX) TIME COUNT (PLUS OR MINUS 16 MICROSECONDS)
5961                            <1> ;     (CX) = 0
5962                            <1> ;-------------------------------------------------------------------------------
5963                            <1>
5964                            <1> ; Refresh period: 30 micro seconds (15-80 us)
5965                            <1> ; (16/12/2014 - AWARDBIOS 1999 - ATORGS.ASM, WAIT_REFRESH)
5966                            <1>
5967                            <1> ;WAITF:                               ; DELAY FOR (CX)*15.085737 US
5968 00001DED 6650              <1>       PUSH   AX                 ; SAVE WORK REGISTER (AH)
5969                            <1>       ; 16/12/2014
5970                            <1>       ;shr   cx, 1              ; convert to count of 30 micro seconds
5971 00001DEF D1E9              <1>       shr    ecx, 1 ; 21/02/2015
5972                            <1> ;17/12/2014
5973                            <1> ;WAITF1:
5974                            <1> ;     IN     AL, PORT_B   ;061h  ; READ CURRENT COUNTER OUTPUT STATUS
5975                            <1> ;     AND    AL, REFRESH_BIT     ;00010000b ; MASK FOR REFRESH DETERMINE BIT
5976                            <1> ;     CMP    AL, AH              ; DID IT JUST CHANGE
5977                            <1> ;     JE     short WAITF1        ; WAIT FOR A CHANGE IN OUTPUT LINE
5978                            <1> ;     MOV    AH, AL              ; SAVE NEW FLAG STATE
5979                            <1> ;     LOOP   WAITF1              ; DECREMENT HALF CYCLES TILL COUNT END
5980                            <1>       ;
5981                            <1>       ; 17/12/2014
5982                            <1>       ;
5983                            <1>       ; Modification from 'WAIT_REFRESH' procedure of AWARD BIOS - 1999
5984                            <1>       ;
5985                            <1> ;WAIT_REFRESH:  Uses port 61, bit 4 to have CPU speed independent waiting.
5986                            <1> ;     INPUT:  CX = number of refresh periods to wait
5987                            <1> ;                  (refresh periods = 1 per 30 microseconds on most machines)
5988                            <1> WR_STATE_0:
5989 00001DF1 E461              <1>       IN     AL,PORT_B          ; IN AL,SYS1
5990 00001DF3 A810              <1>       TEST   AL,010H
5991 00001DF5 74FA              <1>       JZ     SHORT WR_STATE_0
5992                            <1> WR_STATE_1:
5993 00001DF7 E461              <1>       IN     AL,PORT_B          ; IN AL,SYS1
5994 00001DF9 A810              <1>       TEST   AL,010H
5995 00001DFB 75FA              <1>       JNZ    SHORT WR_STATE_1
5996 00001DFD E2F2              <1>         LOOP    WR_STATE_0
5997                            <1>       ;
5998 00001DFF 6658              <1>       POP    AX                 ; RESTORE (AH)
5999 00001E01 C3                <1>       RETn                      ; (CX) = 0
6000                            <1>
6001                            <1> ; 09/07/2016
6002                            <1> ; 01/07/2016
6003                            <1> ; 24/06/2016
6004                            <1> ; 23/06/2016 - TRDOS 386 (TRDOS v2.0)
6005                            <1> ; VIDEO1.ASM - 24/03/1985 (IBM PC-AT BIOS source code)
6006                            <1> ;-------------------------------------------------------------------------------
6007                            <1> ; WRITE_STRING                                                    :
6008                            <1> ;     THIS ROUTINE WRITES A STRING OF CHARACTERS TO THE CRT.                  :
6009                            <1> ; INPUT                                                  :
6010                            <1> ;     (AL) = WRITE STRING COMMAND  0 - 3                          :
6011                            <1> ;     (BH) = DISPLAY PAGE (ACTIVE PAGE)                           :
6012                            <1> ;     (CX) = COUNT OF CHARACTERS TO WRITE, IF (CX) = 0 THEN RETURN            :
6013                            <1> ;     (DX) = CURSOR POSITION FOR START OF STRING WRITE             :
6014                            <1> ;     (BL) = ATTRIBUTE OF CHARACTER TO WRITE IF (AL) = 0  OR     (AL) = 1       :
6015                            <1> ;     (eBP) = SOURCE STRING OFFSET                                :
6016                            <1> ; OUTPUT                                                 :
```

```
                        <1> ;     NONE                                                          :
                        <1> ;--------------------------------------------------------------------------------
                        <1>
                        <1> ; AL = 00h: Assign all characters the attribute in BL; do not update cursor
                        <1> ; AL = 01h: Assign all characters the attribute in BL; update cursor
                        <1> ; AL = 02h: Use attributes in string; do not update cursor
                        <1> ; AL = 03h: Use attributes in string; update cursor
                        <1>
                        <1> WRITE_STRING:
                        <1>       ; 12/09/2016
                        <1>       ; 09/07/2016
                        <1>       ;cmp   byte [CRT_MODE], 7 ; 6?!
                        <1>       ;ja    VIDEO_RETURN       ; not a valid function for VGA modes
                        <1>       ;
00001E02 A2[605B0100]   <1>       mov    [w_str_cmd], al           ; save (AL) command
00001E07 3C04           <1>       CMP    AL, 4              ; TEST FOR INVALID WRITE STRING OPTION
00001E09 0F8345F7FFFF   <1>       JNB    VIDEO_RETURN       ; IF OPTION INVALID THEN RETURN
                        <1>
                        <1>         ;JCXZ   VIDEO_RETURN            ; IF ZERO LENGTH STRING THEN RETURN
                        <1>
00001E0F 67E35E         <1>         jcxz    P55                   ; 01/07/2016
                        <1>
                        <1>
                        <1>       ; 01/07/2016
                        <1>       ;and   ecx, 0FFFFh
                        <1>       ; ECX = byte count
                        <1>       ;push  ecx
00001E12 89EE           <1>       mov    esi, ebp ; user buffer
00001E14 BF00000700     <1>       mov    edi, Cluster_Buffer  ; system buffer
00001E19 E8AECA0000     <1>       call   transfer_from_user_buffer
                        <1>       ;pop   ecx
00001E1E 0F8230F7FFFF   <1>       jc     VIDEO_RETURN
                        <1>       ; ecx = transfer (byte) count = character count
00001E24 BD00000700     <1>       mov    ebp, Cluster_Buffer
                        <1>       ; 12/09/2016
00001E29 803D[C25E0000]07   <1>       cmp    byte [CRT_MODE], 7 ; 6?!
00001E30 0F879F000000   <1>       ja     vga_write_string
                        <1>       ;
00001E36 0FB6F7         <1>       movzx  esi, bh                   ; GET CURRENT CURSOR PAGE
00001E39 66D1E6         <1>       SAL    SI,1               ; CONVERT TO PAGE OFFSET  (SI= PAGE)
                        <1>       ; *****
00001E3C 66FFB6[E64D0100]   <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:
00001E43 53             <1>       push   ebx ; ****
00001E44 51             <1>       push   ecx ; ***
00001E45 56             <1>       push   esi ; **
00001E46 52             <1>       push   edx ; *
00001E47 E8FCFEFFFF     <1>       call   _set_cpos
                        <1> P50:
00001E4C 8A4500         <1>       MOV    AL, [eBP]          ; GET CHARACTER FROM INPUT STRING
00001E4F 45             <1>       INC    eBP                ; BUMP POINTER TO CHARACTER
                        <1>
                        <1> ;-----      TEST FOR SPECIAL CHARACTER'S
                        <1>
00001E50 3C08           <1>       CMP    AL, 08H                  ; IS IT A BACKSPACE
00001E52 740C           <1>       JE     short P51          ; BACK_SPACE
00001E54 3C0D           <1>       CMP    AL, 0Dh ; CR       ; IS IT CARRIAGE RETURN
00001E56 7408           <1>       JE     short P51          ; CAR_RET
00001E58 3C0A           <1>       CMP    AL, 0Ah ; LF       ; IS IT A LINE FEED
00001E5A 7404           <1>       JE     short P51          ; LINE_FEED
00001E5C 3C07           <1>       CMP    AL, 07h                  ; IS IT A BELL
00001E5E 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>
00001E60 E860FEFFFF     <1>       call   _write_tty_m3
                        <1>
00001E65 5A             <1>       pop    edx ; *
00001E66 5E             <1>       pop    esi ; **
                        <1>
00001E67 668B96[E64D0100]   <1>       MOV    DX, [eSI+CURSOR_POSN]     ; GET CURRENT CURSOR POSITION
00001E6E EB46           <1>       JMP    SHORT P54          ; SET CURSOR POSITION AND CONTINUE
                        <1> P55:
00001E70 E9DFF6FFFF     <1>       JMP    VIDEO_RETURN
                        <1> P52:
00001E75 66B90100       <1>       MOV    CX, 1              ; SET CHARACTER WRITE AMOUNT TO ONE
00001E79 803D[605B0100]02   <1>       CMP    byte [w_str_cmd], 2 ; IS THE ATTRIBUTE IN THE STRING
00001E80 7204           <1>       JB     short P53          ; IF NOT THEN SKIP
00001E82 8A5D00         <1>       MOV    BL, [eBP]          ; ELSE GET NEW ATTRIBUTE
00001E85 45             <1>       INC    eBP                ; BUMP STRING POINTER
                        <1> P53:
                        <1>       ;MOV   AH,09H             ; GOT_CHARACTER
                        <1>       ;INT   10H                ; WRITE CHARACTER TO THE CRT
                        <1>
00001E86 E8AEFDFFFF     <1>       call   _write_c_current
                        <1>
00001E8B 5A             <1>       pop    edx ; *
                        <1>
00001E8C 0FB6F7         <1>       movzx  esi, bh ; video page number (0 to 7)
00001E8F 889E[CB5E0000]     <1>       mov    [esi+chr_attrib], bl ; color/attribute
                        <1>
00001E95 FEC2           <1>       INC    DL                 ; INCREMENT COLUMN COUNTER
00001E97 3A15[C45E0000]     <1>       CMP    DL, [CRT_COLS]           ; IF COLS ARE WITHIN RANGE FOR THIS MODE
00001E9D 7217           <1>       JB     short P54          ;    THEN GO TO COLUMNS SET
00001E9F FEC6           <1>       INC    DH                 ; BUMP ROW COUNTER BY ONE
00001EA1 28D2           <1>       SUB    DL, DL             ; SET COLUMN COUNTER TO ZERO
00001EA3 80FE19         <1>       CMP    DH, 25             ; IF ROWS ARE LESS THAN 25 THEN
00001EA6 720E           <1>       JB     short P54          ; GO TO ROWS_COLUMNS_SET
```

```
6119                                    <1>
6120 00001EA8 66B80A0E                  <1>       MOV    AX,0E0AH           ; ELSE SCROLL SCREEN
6121                                    <1>       ;INT   10H                ; RESET ROW COUNTER TO 24
6122                                    <1>
6123 00001EAC E814FEFFFF                <1>       call   _write_tty_m3
6124                                    <1>
6125 00001EB1 66BA0018                  <1>       mov    dx, 1800h          ; Column = 0, Row = 24
6126 00001EB5 5E                        <1>       pop    esi ; **
6127                                    <1> P54:
6128                                    <1>                                 ; ROW_COLUMNS_SET
6129                                    <1>       ;MOV   AX,0200H           ; SET NEW CURSOR POSITION COMMAND
6130                                    <1>       ;INT   10H                ; ESTABLISH NEW CURSOR POSITION
6131                                    <1>
6132 00001EB6 59                        <1>       pop    ecx ; ***
6133 00001EB7 5B                        <1>       pop    ebx ; ****
6134                                    <1>
6135                                    <1>       ;LOOP  P50                ; DO IT ONCE MORE UNTIL (CX) = ZERO
6136 00001EB8 6649                      <1>       dec    cx
6137 00001EBA 7587                      <1>       jnz    short P50next
6138                                    <1>
6139 00001EBC 665A                      <1>       POP    DX  ; *****        ; RESTORE OLD CURSOR COORDINATES
6140                                    <1>
6141 00001EBE F605[605B0100]01          <1>       test   byte [w_str_cmd], 1 ; IF CURSOR WAS NOT TO BE MOVED
6142 00001EC5 0F8589F6FFFF              <1>       JNZ    VIDEO_RETURN        ; THEN EXIT WITHOUT RESETTING OLD VALUE
6143                                    <1>
6144                                    <1>       ;MOV   AX,0200H           ; ELSE RESTORE OLD CURSOR POSITION
6145                                    <1>       ;INT   10H
6146                                    <1>                                 ; DONE - EXIT WRITE STRING
6147 00001ECB E878FEFFFF                <1>       call   _set_cpos
6148 00001ED0 E97FF6FFFF                <1>       JMP    VIDEO_RETURN        ; RETURN TO CALLER
6149                                    <1>
6150                                    <1> vga_write_string:
6151                                    <1>       ; 12/09/2016 - TRDOS 386 (TRDOS v2.0)
6152                                    <1>       ;
6153                                    <1>       ; derived from 'Plex86/Bochs VGABios' source code
6154                                    <1>       ; vgabios-0.7a (2011)
6155                                    <1>       ; by the LGPL VGABios developers Team (2001-2008)
6156                                    <1>       ; 'vgabios.c', ' biosfn_write_string'
6157                                    <1>
6158                                    <1>       ; INPUT                                                 :
6159                                    <1>       ;     (AL) = WRITE STRING COMMAND  0 - 3                  :
6160                                    <1>       ;     (BH) = DISPLAY PAGE (ACTIVE PAGE)                   :
6161                                    <1>       ;     (CX) = COUNT OF CHARACTERS TO WRITE, IF (CX) = 0 THEN RETURN       :
6162                                    <1>       ;     (DX) = CURSOR POSITION FOR START OF STRING WRITE     :
6163                                    <1>       ;     (BL) = ATTRIBUTE OF CHARACTER TO WRITE IF (AL) = 0  OR     (AL) = 1  :
6164                                    <1>       ;     (eBP) = SOURCE STRING OFFSET                        :
6165                                    <1>       ; OUTPUT                                                :
6166                                    <1>       ;     NONE                                              :
6167                                    <1>       ;-------------------------------------------------------------------------;
6168                                    <1>
6169                                    <1>       ; AL = 00h: Assign all characters the attribute in BL; do not update cursor
6170                                    <1>       ; AL = 01h: Assign all characters the attribute in BL; update cursor
6171                                    <1>       ; AL = 02h: Use attributes in string; do not update cursor
6172                                    <1>       ; AL = 03h: Use attributes in string; update cursor
6173                                    <1>
6174                                    <1>       ; biosfn_write_string(GET_AL(),GET_BH(),GET_BL(),CX,GET_DH(),GET_DL(),ES,BP);
6175                                    <1>       ; static void biosfn_write_string (flag,page,attr,count,row,col,seg,offset)
6176                                    <1>
6177                                    <1>       ; // Read curs info for the page
6178                                    <1>       ; biosfn_get_cursor_pos(page,&dummy,&oldcurs);
6179                                    <1>       ; bh = video page = 0
6180                                    <1>       ;movzx esi, word [CURSOR_POSN] ; current cursor position for video page 0
6181                                    <1>
6182                                    <1>       ; // if row=0xff special case : use current cursor position
6183                                    <1>       ; if(row==0xff)
6184                                    <1>       ;  {col=oldcurs&0x00ff;
6185                                    <1>       ;   row=(oldcurs&0xff00)>>8;
6186                                    <1>       ;  }
6187                                    <1>
6188                                    <1>       ;mov   al, [w_str_cmd]
6189                                    <1>
6190 00001ED5 80FEFF                    <1>       cmp    dh, 0FFh
6191 00001ED8 7407                      <1>       je     short vga_wstr_1 ; user current cursor position
6192                                    <1> vga_wstr_0:
6193                                    <1>       ; set cursor position
6194 00001EDA 668915[E64D0100]          <1>       mov    [CURSOR_POSN], dx ; save cursor pos for pg 0
6195                                    <1> vga_wstr_1:
6196 00001EE1 66FF35[E64D0100]          <1>       push   word [CURSOR_POSN] ; *
6197                                    <1>
6198                                    <1>       ; ebp = string offset in system buffer (user buffer was copied to)
6199                                    <1>
6200                                    <1>       ; while(count--!=0)
6201                                    <1>       ;  {
6202                                    <1>       ;   car=read_byte(seg,offset++);
6203                                    <1>       ;   if((flag&0x02)!=0)
6204                                    <1>       ;    attr=read_byte(seg,offset++);
6205                                    <1>       ;    biosfn_write_teletype(car,page,attr,WITH_ATTR);
6206                                    <1>       ;  }
6207                                    <1>
6208                                    <1>       ;push  eax ; **
6209                                    <1>       ;test  al, 2
6210 00001EE8 F605[605B0100]02          <1>       test   byte [w_str_cmd], 2
6211 00001EEF 751D                      <1>       jnz    short vga_wstr_3
6212 00001EF1 881D[F74D0100]            <1>       mov    [ccolor], bl
6213                                    <1> vga_wstr_2:
6214 00001EF7 51                        <1>       push   ecx
6215 00001EF8 8A4500                    <1>       mov    al, [ebp]
6216 00001EFB E8CC0A0000                <1>       call   vga_write_teletype
6217 00001F00 59                        <1>       pop    ecx
6218 00001F01 6649                      <1>       dec    cx
6219 00001F03 741E                      <1>       jz     short vga_wstr_4
6220 00001F05 45                        <1>       inc    ebp
```

```
6221 00001F06 8A1D[F74D0100]      <1>       mov   bl, [ccolor]
6222 00001F0C EBE9                <1>       jmp   short vga_wstr_2
6223                              <1> vga_wstr_3:
6224 00001F0E 51                  <1>       push  ecx
6225 00001F0F 8A4500              <1>       mov   al, [ebp]
6226 00001F12 45                  <1>       inc   ebp
6227 00001F13 8A5D00              <1>       mov   bl, [ebp]
6228 00001F16 E8B10A0000          <1>       call  vga_write_teletype
6229 00001F1B 59                  <1>       pop   ecx
6230 00001F1C 6649                <1>       dec   cx
6231 00001F1E 7403                <1>       jz    short vga_wstr_4
6232 00001F20 45                  <1>       inc   ebp
6233 00001F21 EBEB                <1>       jmp   short vga_wstr_3
6234                              <1> vga_wstr_4:
6235                              <1>       ; // Set back curs pos
6236                              <1>       ; if((flag&0x01)==0)
6237                              <1>       ;  biosfn_set_cursor_pos(page,oldcurs);
6238                              <1>       ; }
6239                              <1>       ;pop  eax ; **
6240 00001F23 665A                <1>       pop   dx ; word [CURSOR_POSN] ; *
6241                              <1>       ;test al, 1
6242 00001F25 F605[605B0100]01    <1>       test  byte [w_str_cmd], 1
6243 00001F2C 0F8522F6FFFF        <1>       jnz   VIDEO_RETURN
6244 00001F32 668915[E64D0100]    <1>       mov   [CURSOR_POSN], dx
6245 00001F39 E916F6FFFF          <1>       JMP   VIDEO_RETURN
6246                              <1>
6247                              <1> ; 07/07/2016
6248                              <1> ; 27/06/2016 - TRDOS 386 (TRDOS v2.0)
6249                              <1> ; VIDEO1.ASM - 24/03/1985 (IBM PC-AT BIOS source code)
6250                              <1> ;--------------------------------------------------
6251                              <1> ;  SCROLL UP
6252                              <1> ;   THIS ROUTINE SCROLLS UP THE INFORMATION ON THE CRT
6253                              <1> ; ENTRY ---
6254                              <1> ;  CH,CL = UPPER LEFT CORNER OF REGION TO SCROLL
6255                              <1> ;  DH,DL = LOWER RIGHT CORNER OF REGION TO SCROLL
6256                              <1> ;   BOTH OF THE ABOVE ARE IN CHARACTER POSITIONS
6257                              <1> ;  BH = FILL VALUE FOR BLANKED LINES
6258                              <1> ;  AL = # LINES TO SCROLL (AL=0 MEANS BLANK THE ENTIRE FIELD)
6259                              <1> ;  DS = DATA SEGMENT
6260                              <1> ;  ES = REGEN SEGMENT
6261                              <1> ; EXIT --
6262                              <1> ;  NOTHING, THE SCREEN IS SCROLLED
6263                              <1> ;--------------------------------------------------
6264                              <1>
6265                              <1>       ; cl = upper left column
6266                              <1>       ; ch = upper left row
6267                              <1>       ; dl = lower rigth column
6268                              <1>       ; dh = lower right row
6269                              <1>       ;
6270                              <1>       ; al = line count (AL=0 means blank entire fields)
6271                              <1>       ; bl = fill value for blanked lines
6272                              <1>       ; bh = unused
6273                              <1>
6274                              <1> GRAPHICS_UP:
6275                              <1>       ; 07/07/2016
6276                              <1>       ;AH = Current video mode, [CRT_MODE]
6277 00001F3E 80FC07              <1>       cmp   ah, 7
6278 00001F41 7766                <1>       ja    short vga_graphics_up
6279                              <1>       ;je   n0
6280                              <1>
6281 00001F43 88C7                <1>       MOV   bh, al              ; save line count in BH
6282 00001F45 6689C8              <1>       MOV   AX, CX              ; GET UPPER LEFT POSITION INTO AX REG
6283                              <1>
6284                              <1> ;-----      USE CHARACTER SUBROUTINE FOR POSITIONING
6285                              <1> ;-----      ADDRESS RETURNED IS MULTIPLIED BY 2 FROM CORRECT VALUE
6286                              <1>
6287 00001F48 E8D9050000          <1>       CALL  GRAPH_POSN
6288 00001F4D 0FB7F8              <1>       MOVzx eDI, AX                   ; SAVE RESULT AS DESTINATION ADDRESS
6289                              <1>
6290                              <1> ;-----      DETERMINE SIZE OF WINDOW
6291                              <1>
6292 00001F50 6629CA              <1>       SUB   DX, CX
6293 00001F53 6681C20101          <1>         ADD     DX, 101h              ; ADJUST VALUES
6294 00001F58 C0E602              <1>       SAL   DH, 2               ; MULTIPLY ROWS BY 4 AT 8 VERT DOTS/CHAR
6295                              <1>                                 ; AND EVEN/ODD ROWS
6296                              <1> ;-----      DETERMINE CRT MODE
6297                              <1>
6298 00001F5B 803D[C25E0000]06    <1>       CMP   byte [CRT_MODE], 6  ; TEST FOR MEDIUM RES
6299 00001F62 7305                <1>         JNC     short _R7_              ; FIND_SOURCE
6300                              <1>
6301                              <1> ;-----      MEDIUM RES UP
6302 00001F64 D0E2                <1>       SAL   DL, 1               ; # COLUMNS * 2, SINCE 2 BYTES/CHAR
6303 00001F66 66D1E7              <1>       SAL   DI, 1               ; OFFSET *2 SINCE 2 BYTES/CHAR
6304                              <1>
6305                              <1> ;-----      DETERMINE THE SOURCE ADDRESS IN THE BUFFER
6306                              <1> _R7_:                                 ; FIND_SOURCE
6307 00001F69 81C700800B00        <1>       add   edi, 0B8000h
6308 00001F6F C0E702              <1>       sal   bh, 2               ; multiply number of lines by 4
6309 00001F72 7431                <1>         JZ      short _R11             ; IF ZERO, THEN BLANK ENTIRE FIELD
6310 00001F74 B050                <1>       MOV   AL, 80              ; 80 BYTES/ROW
6311 00001F76 F6E7                <1>       mul   bh                  ; determine offset to source
6312 00001F78 0FB7F0              <1>       movzx esi, ax                   ; offset to source
6313 00001F7B 01FE                <1>       add   eSI, eDI            ; SET UP SOURCE
6314 00001F7D 88F4                <1>       MOV   AH, DH              ; NUMBER OF ROWS IN FIELD
6315 00001F7F 28FC                <1>       sub   ah, bh              ; determine number to move
6316                              <1>
6317                              <1> ;-----      LOOP THROUGH, MOVING ONE ROW AT A TIME, BOTH EVEN AND ODD FIELDS
6318                              <1> _R8:                                  ; ROW_LOOP
6319 00001F81 E812040000          <1>         CALL    _R17                    ; MOVE ONE ROW
6320 00001F86 6681EEB01F          <1>       SUB   SI, 2000h-80        ; MOVE TO NEXT ROW
6321 00001F8B 6681EFB01F          <1>       SUB   DI, 2000h-80
6322 00001F90 FECC                <1>       DEC   AH                  ; NUMBER OF ROWS TO MOVE
```

```
6323 00001F92 75ED              <1>         JNZ     short _R8               ; CONTINUE TILL ALL MOVED
6324                            <1>
6325                            <1> ;-----      FILL IN THE VACATED LINE(S)
6326                            <1> _R9:                                    ; CLEAR ENTRY
6327 00001F94 88D8              <1>       mov   al, bl              ; attribute to fill with
6328                            <1> _R10_:
6329 00001F96 E819040000        <1>         CALL    _R18                    ; CLEAR THAT ROW
6330 00001F9B 6681EFB01F        <1>       SUB   DI, 2000h-80        ; POINT TO NEXT LINE
6331 00001FA0 FECF              <1>       dec   bh                  ; number of lines to fill
6332 00001FA2 75F2              <1>         JNZ     short _R10_             ; CLEAR LOOP
6333 00001FA4 C3                <1>       retn                      ; EVERYYHING DONE
6334                            <1>
6335                            <1> _R11:                                   ; BLANK_FIELD
6336 00001FA5 88F7              <1>       mov   bh, dh              ; set blank count to everything in field
6337 00001FA7 EBEB              <1>         JMP     short _R9               ; CLEAR THE FIELD
6338                            <1>
6339                            <1> vga_graphics_up:
6340                            <1>       ; 08/08/2016
6341                            <1>       ; 07/08/2016
6342                            <1>       ; 04/08/2016
6343                            <1>       ; 01/08/2016
6344                            <1>       ; 31/07/2016
6345                            <1>       ; 07/07/2016 - TRDOS 386 (TRDOS v2.0)
6346                            <1>       ;
6347                            <1>       ; derived from 'Plex86/Bochs VGABios' source code
6348                            <1>       ; vgabios-0.7a (2011)
6349                            <1>       ; by the LGPL VGABios developers Team (2001-2008)
6350                            <1>       ; 'vgabios.c', 'biosfn_scroll'
6351                            <1>       ;
6352                            <1>
6353                            <1>       ; cl = upper left column
6354                            <1>       ; ch = upper left row
6355                            <1>       ; dl = lower rigth column
6356                            <1>       ; dh = lower right row
6357                            <1>       ;
6358                            <1>       ; al = line count (AL=0 means blank entire fields)
6359                            <1>       ; bl = fill value for blanked lines
6360                            <1>       ; bh = unused
6361                            <1>       ;
6362                            <1>       ; ah = [CRT_MODE], current video mode
6363                            <1>
6364 00001FA9 88C7              <1>       mov   bh, al ; 31/07/2016
6365 00001FAB BE[E65E0000]      <1>       mov   esi, vga_g_modes
6366 00001FB0 89F7              <1>       mov   edi, esi
6367 00001FB2 83C708            <1>       add   edi, vga_g_mode_count
6368                            <1> vga_g_up_0:
6369 00001FB5 AC                <1>       lodsb
6370 00001FB6 38E0              <1>       cmp   al, ah ; [CRT_MODE]
6371 00001FB8 7405              <1>       je    short vga_g_up_1
6372 00001FBA 39FE              <1>       cmp   esi, edi
6373 00001FBC 72F7              <1>       jb    short vga_g_up_0
6374                            <1>       ;xor  bh, bh ; 31/07/2016)
6375 00001FBE C3                <1>       retn  ; nothing to do
6376                            <1> vga_g_up_1:
6377 00001FBF 88F8              <1>       mov   al, bh ; 31/07/2016
6378 00001FC1 83C64F            <1>       add   esi, vga_g_memmodel - (vga_g_modes + 1)
6379                            <1>       ; [ESI] = VGA memory model number (LINEAR8, PLANAR4, PLANAR1)
6380                            <1>
6381                            <1>       ; if(rlr>=nbrows)rlr=nbrows-1;
6382                            <1>       ; if(clr>=nbcols)clr=nbcols-1;
6383                            <1>       ; if(nblines>nbrows)nblines=0;
6384                            <1>       ; cols=clr-cul+1;
6385                            <1>
6386 00001FC4 3A35[CA5E0000]      <1>       cmp   dh, [VGA_ROWS]
6387 00001FCA 7208              <1>       jb    short vga_g_up_2
6388 00001FCC 8A35[CA5E0000]    <1>       mov   dh, [VGA_ROWS]
6389 00001FD2 FECE              <1>       dec   dh
6390                            <1> vga_g_up_2:
6391 00001FD4 3A15[C45E0000]    <1>       cmp   dl, [CRT_COLS]  ; = [VGA_COLS]
6392 00001FDA 7208              <1>       jb    short vga_g_up_3
6393 00001FDC 8A15[C45E0000]    <1>       mov   dl, [CRT_COLS]
6394 00001FE2 FECA              <1>       dec   dl
6395                            <1> vga_g_up_3:
6396 00001FE4 3A05[CA5E0000]    <1>       cmp   al, [VGA_ROWS]
6397 00001FEA 7602              <1>       jna   short vga_g_up_4
6398 00001FEC 28C0              <1>       sub   al, al ; 0
6399                            <1> vga_g_up_4:
6400 00001FEE 88D7              <1>       mov   bh, dl ; clr
6401 00001FF0 28CF              <1>       sub   bh, cl ; cul
6402 00001FF2 FEC7              <1>       inc   bh ; cols = clr-cul+1
6403                            <1>
6404 00001FF4 20C0              <1>       and   al, al ; nblines = 0
6405 00001FF6 755D              <1>       jnz   short vga_g_up_6
6406 00001FF8 20ED              <1>       and   ch, ch ; rul = 0
6407 00001FFA 7559              <1>       jnz   short vga_g_up_6
6408 00001FFC 20C9              <1>       and   cl, cl ; cul = 0
6409 00001FFE 7555              <1>       jnz   short vga_g_up_6
6410                            <1>
6411 00002000 6650              <1>       push  ax
6412 00002002 A0[CA5E0000]      <1>       mov   al, [VGA_ROWS]
6413 00002007 FEC8              <1>       dec   al
6414 00002009 38C6              <1>       cmp   dh, al ; rlr = nbrows-1
6415 0000200B 7546              <1>       jne   short vga_g_up_5
6416 0000200D A0[C45E0000]      <1>         mov     al, [CRT_COLS]  ; = VGA_COLS
6417 00002012 FEC8              <1>       dec   al
6418 00002014 38C2              <1>       cmp   dl, al ; clr = nbcols-1
6419 00002016 753B              <1>       jne   short vga_g_up_5
6420 00002018 6658              <1>       pop   ax
6421                            <1>
6422 0000201A 66B80502          <1>       mov   ax, 0205h
6423 0000201E 66BACE03          <1>       mov   dx, 3CEh ; VGAREG_GRDC_ADDRESS
6424 00002022 66EF              <1>       out   dx, ax
```

```
6425 00002024 A0[CA5E0000]        <1>       mov    al, [VGA_ROWS]
6426 00002029 8A25[C45E0000]      <1>       mov    ah, [CRT_COLS] ; = [VGA_COLS]
6427 0000202F F6E4                <1>       mul    ah
6428 00002031 0FB7D0              <1>       movzx  edx, ax
6429                              <1>       ; 08/08/2016
6430 00002034 0FB605[C65E0000]    <1>       movzx  eax, byte [CHAR_HEIGHT]
6431 0000203B F7E2                <1>       mul    edx
6432                              <1>       ; eax = byte count
6433 0000203D 89C1                <1>       mov    ecx, eax
6434                              <1>       ;; 07/08/2016
6435                              <1>       ;shl   dx, 3 ; * 8 ; * [CHAR_HEIGHT]
6436                              <1>       ;mov   ecx, edx
6437 0000203F 88D8                <1>       mov    al, bl ; fill value for blanked lines
6438 00002041 BF00000A00          <1>       mov    edi, 0A0000h
6439 00002046 F3AA                <1>       rep    stosb
6440                              <1>
6441 00002048 66B80500            <1>       mov    ax, 5
6442 0000204C 66BACE03            <1>       mov    dx, 3CEh ; VGAREG_GRDC_ADDRESS
6443 00002050 66EF                <1>       out    dx, ax ; 0005h
6444                              <1>
6445 00002052 C3                  <1>       retn
6446                              <1>
6447                              <1> vga_g_up_5:
6448 00002053 6658                <1>       pop    ax
6449                              <1>
6450                              <1> vga_g_up_6:
6451                              <1>       ; [ESI] = VGA memory model number for current video mode
6452                              <1>         ;
6453                              <1>         ; LINEAR8 equ 5
6454                              <1>         ; PLANAR4 equ 4
6455                              <1>         ; PLANAR1 equ 3
6456                              <1>
6457 00002055 803E04              <1>       cmp    byte [esi], PLANAR4
6458 00002058 7424                <1>       je     short vga_g_up_planar
6459 0000205A 803E03              <1>       cmp    byte [esi], PLANAR1
6460 0000205D 741F                <1>       je     short vga_g_up_planar
6461                              <1> vga_g_up_linear8:
6462                              <1>       ; 07/07/2016 (TEMPORARY)
6463                              <1>       ;
6464                              <1>       ; cl = upper left column ; cul
6465                              <1>       ; ch = upper left row ; rul
6466                              <1>       ; dl = lower rigth column ; clr
6467                              <1>       ; dh = lower right row ; rlr
6468                              <1>
6469                              <1> vga_g_up_l0:
6470                              <1>       ;{for(i=rul;i<=rlr;i++)
6471                              <1>       ; if((i+nblines>rlr)||(nblines==0))
6472 0000205F 08C0                <1>       or     al, al
6473 00002061 7414                <1>       jz     short vga_g_up_l2
6474 00002063 88C4                <1>       mov    ah, al
6475 00002065 00EC                <1>       add    ah, ch ; i+nblines
6476                              <1>       ;jc    short vga_g_up_l2
6477 00002067 38F4                <1>       cmp    ah, dh
6478 00002069 770C                <1>       ja     short vga_g_up_l2
6479                              <1>       ; else
6480                              <1>       ;  vgamem_copy_pl4(cul,i+nblines,i,cols,nbcols,cheight);
6481 0000206B E8F2000000          <1>       call   vgamem_copy_l8
6482                              <1> vga_g_up_l1:
6483 00002070 FEC5                <1>       inc    ch
6484 00002072 38F5                <1>       cmp    ch, dh
6485 00002074 76E9                <1>       jna    short vga_g_up_l0
6486 00002076 C3                  <1>       retn
6487                              <1> vga_g_up_l2:
6488                              <1>       ; vgamem_fill_pl4(cul,i,cols,nbcols,cheight,attr);
6489 00002077 E850010000          <1>       call   vgamem_fill_l8
6490 0000207C EBF2                <1>       jmp    short vga_g_up_l1
6491                              <1>
6492                              <1> vga_g_up_planar:
6493                              <1>       ; cl = upper left column ; cul
6494                              <1>       ; ch = upper left row ; rul
6495                              <1>       ; dl = lower rigth column ; clr
6496                              <1>       ; dh = lower right row ; rlr
6497                              <1> vga_g_up_pl0:
6498                              <1>       ;{for(i=rul;i<=rlr;i++)
6499                              <1>       ; if((i+nblines>rlr)||(nblines==0))
6500 0000207E 20C0                <1>       and    al, al
6501 00002080 7414                <1>       jz     short vga_g_up_pl2
6502 00002082 88C4                <1>       mov    ah, al
6503 00002084 00EC                <1>       add    ah, ch ; i+nblines
6504                              <1>       ;jc    short vga_g_up_pl2
6505 00002086 38F4                <1>       cmp    ah, dh
6506 00002088 770C                <1>       ja     short vga_g_up_pl2
6507                              <1>       ; else
6508                              <1>       ;  vgamem_copy_pl4(cul,i+nblines,i,cols,nbcols,cheight);
6509 0000208A E80E000000          <1>       call   vgamem_copy_pl4
6510                              <1> vga_g_up_pl1:
6511 0000208F FEC5                <1>       inc    ch
6512 00002091 38F5                <1>       cmp    ch, dh
6513 00002093 76E9                <1>       jna    short vga_g_up_pl0
6514 00002095 C3                  <1>       retn
6515                              <1> vga_g_up_pl2:
6516                              <1>       ; vgamem_fill_pl4(cul,i,cols,nbcols,cheight,attr);
6517 00002096 E870000000          <1>       call   vgamem_fill_pl4
6518 0000209B EBF2                <1>       jmp    short vga_g_up_pl1
6519                              <1>
6520                              <1> vgamem_copy_pl4:
6521                              <1>       ; 08/08/2016
6522                              <1>       ; 07/08/2016
6523                              <1>       ; 07/07/2016 - TRDOS 386 (TRDOS v2.0)
6524                              <1>       ;
6525                              <1>       ; derived from 'Plex86/Bochs VGABios' source code
6526                              <1>       ; vgabios-0.7a (2011)
```

```
6527                                 <1>       ; by the LGPL VGABios developers Team (2001-2008)
6528                                 <1>       ; 'vgabios.c', 'vgamem_copy_pl4'
6529                                 <1>       ;
6530                                 <1>       ; vgamem_copy_pl4(xstart,ysrc,ydest,cols,nbcols,cheight)
6531                                 <1>       ; cl = xstart, ah = ysrc (i+nblines), ch = ydest (i),
6532                                 <1>       ; bh = cols, [CRT_COLS] = nbcols, [CHAR_HEIGHT] = cheight
6533                                 <1>
6534                                 <1>       ; src=ysrc*cheight*nbcols+xstart;
6535                                 <1>       ; dest=ydest*cheight*nbcols+xstart;
6536                                 <1>
6537 0000209D 52                    <1>       push   edx
6538 0000209E 50                    <1>       push   eax
6539                                 <1>
6540                                 <1>       ; outw(VGAREG_GRDC_ADDRESS, 0x0105)
6541 0000209F 66B80501              <1>       mov    ax, 0105h
6542 000020A3 66BACE03              <1>       mov    dx, 3CEh ; VGAREG_GRDC_ADDRESS
6543 000020A7 66EF                  <1>       out    dx, ax
6544                                 <1>
6545                                 <1>       ; 07/08/2016
6546                                 <1>       ;mov     ah, [esp+1]
6547                                 <1>       ;movzx edx, ah ; ysrc
6548 000020A9 0FB6542401            <1>       movzx  edx, byte [esp+1]
6549                                 <1>       ; 08/08/2016
6550 000020AE 0FB605[C65E0000]      <1>       movzx  eax, byte [CHAR_HEIGHT]
6551 000020B5 8A25[C45E0000]        <1>       mov    ah, [CRT_COLS] ; nbcols
6552 000020BB F6E4                  <1>       mul    ah
6553                                 <1>       ;; 07/08/2016
6554                                 <1>       ;movzx eax, byte [CRT_COLS]
6555                                 <1>       ;shl   ax, 3 ; * 8 ; * [CHAR_HEIGHT]
6556 000020BD 50                    <1>       push   eax ; cheight * nbcols
6557 000020BE F7E2                  <1>       mul    edx ; * ysrc
6558                                 <1>       ; eax = ysrc * cheight * nbcols
6559                                 <1>       ; edx = 0
6560 000020C0 88CA                  <1>       mov    dl, cl ; edx = xstart
6561 000020C2 01D0                  <1>       add    eax, edx
6562 000020C4 89C6                  <1>       mov    esi, eax ; src
6563 000020C6 88EA                  <1>       mov    dl, ch ; ydest
6564 000020C8 58                    <1>       pop    eax ; cheight * nbcols
6565 000020C9 F7E2                  <1>       mul    edx
6566                                 <1>       ; eax = ydest * cheight * nbcols
6567 000020CB 88CA                  <1>       mov    dl, cl ; edx = xstart
6568 000020CD 01D0                  <1>       add    eax, edx
6569 000020CF 89C7                  <1>       mov    edi, eax ; dest
6570                                 <1>       ; esi = src
6571                                 <1>       ; edi = dest
6572                                 <1>       ; for(i=0;i<cheight;i++)
6573                                 <1>       ; {
6574                                 <1>       ;  memcpyb(0xa000,dest+i*nbcols,0xa000,src+i*nbcols,cols);
6575                                 <1>       ; }
6576 000020D1 51                    <1>       push   ecx
6577 000020D2 B900000A00            <1>       mov    ecx, 0A0000h
6578 000020D7 01CE                  <1>       add    esi, ecx
6579 000020D9 01CF                  <1>       add    edi, ecx
6580                                 <1>       ; 08/08/2016
6581 000020DB 8A35[C65E0000]        <1>       mov    dh, [CHAR_HEIGHT]
6582                                 <1>       ;; 07/08/2016
6583                                 <1>       ;mov   dh, 8 ; 07/08/2016
6584 000020E1 28D2                  <1>       sub    dl, dl ; i
6585                                 <1> vgamem_copy_pl4_0:
6586 000020E3 56                    <1>       push   esi
6587 000020E4 57                    <1>       push   edi
6588 000020E5 0FB605[C45E0000]      <1>       movzx  eax, byte [CRT_COLS]
6589 000020EC F6E2                  <1>       mul    dl
6590                                 <1>       ; eax = i * nbcols
6591 000020EE 01C7                  <1>       add    edi, eax ; dest+i*nbcols
6592 000020F0 01C6                  <1>       add    esi, eax
6593 000020F2 0FB6CF                <1>       movzx  ecx, bh ; cols
6594 000020F5 F3A4                  <1>       rep    movsb
6595 000020F7 5F                    <1>       pop    edi
6596 000020F8 5E                    <1>       pop    esi
6597 000020F9 FECE                  <1>       dec    dh
6598 000020FB 75E6                  <1>       jnz    short vgamem_copy_pl4_0
6599                                 <1> vgamem_copy_pl4_1:
6600 000020FD 59                    <1>       pop    ecx
6601                                 <1>
6602                                 <1>       ; outw(VGAREG_GRDC_ADDRESS, 0x0005);
6603 000020FE 66B80500              <1>       mov    ax, 0005h
6604 00002102 66BACE03              <1>       mov    dx, 3CEh ; VGAREG_GRDC_ADDRESS
6605 00002106 66EF                  <1>       out    dx, ax
6606                                 <1>
6607 00002108 58                    <1>       pop    eax
6608 00002109 5A                    <1>       pop    edx
6609                                 <1>
6610 0000210A C3                    <1>       retn
6611                                 <1>
6612                                 <1> vgamem_fill_pl4:
6613                                 <1>       ; 08/08/2016
6614                                 <1>       ; 07/08/2016
6615                                 <1>       ; 04/08/2016
6616                                 <1>       ; 07/07/2016 - TRDOS 386 (TRDOS v2.0)
6617                                 <1>       ;
6618                                 <1>       ; derived from 'Plex86/Bochs VGABios' source code
6619                                 <1>       ; vgabios-0.7a (2011)
6620                                 <1>       ; by the LGPL VGABios developers Team (2001-2008)
6621                                 <1>       ; 'vgabios.c', 'vgamem_fill_pl4'
6622                                 <1>       ;
6623                                 <1>       ; vgamem_fill_pl4(xstart,ystart,cols,nbcols,cheight,attr)
6624                                 <1>       ; cl = xstart, edi = ch = ystart, bh = cols,
6625                                 <1>       ; [CRT_COLS] = nbcols, [CHAR_HEIGHT] = cheight, attr = 0
6626                                 <1>
6627                                 <1>       ; dest=ystart*cheight*nbcols+xstart;
6628 0000210B 52                    <1>       push   edx
```

```
6629 0000210C 50                  <1>       push   eax
6630                              <1>
6631                              <1>       ; outw(VGAREG_GRDC_ADDRESS, 0x0205)
6632 0000210D 66B80502            <1>       mov    ax, 0205h
6633 00002111 66BACE03            <1>       mov    dx, 3CEh ; VGAREG_GRDC_ADDRESS
6634 00002115 66EF                <1>       out    dx, ax
6635                              <1>
6636                              <1>              ; 08/08/2016
6637 00002117 0FB605[C65E0000]    <1>       movzx   eax, byte [CHAR_HEIGHT]
6638 0000211E F6E5                <1>       mul    ch
6639                              <1>       ;; 07/08/2016
6640                              <1>       ;movzx eax, ch
6641                              <1>       ;shl   ax, 3 ; * 8 ; * [CHAR_HEIGHT]
6642 00002120 0FB615[C45E0000]    <1>       movzx  edx, byte [CRT_COLS] ; = [VGA_COLS]
6643 00002127 F7E2                <1>       mul    edx
6644                              <1>       ; edx  = 0
6645 00002129 88CA                <1>       mov    dl, cl
6646 0000212B 01D0                <1>       add    eax, edx
6647 0000212D 89C7                <1>       mov    edi, eax
6648                              <1>       ; edi = dest
6649                              <1>       ; for(i=0;i<cheight;i++)
6650                              <1>       ; {
6651                              <1>       ;  memsetb(0xa000,dest+i*nbcols,attr,cols);
6652                              <1>       ; }
6653 0000212F 81C700000A00        <1>       add    edi, 0A0000h
6654 00002135 51                  <1>       push   ecx
6655                              <1>       ; 08/08/2016
6656 00002136 8A35[C65E0000]      <1>       mov    dh, [CHAR_HEIGHT]
6657                              <1>       ;; 07/08/2016
6658                              <1>       ;mov   dh, 8 ; 07/08/2016
6659 0000213C 28D2                <1>       sub    dl, dl ; i
6660                              <1> vgamem_fill_pl4_0:
6661 0000213E 57                  <1>       push   edi
6662 0000213F 0FB605[C45E0000]    <1>       movzx  eax, byte [CRT_COLS]
6663 00002146 F6E2                <1>       mul    dl
6664                              <1>       ; eax = i * nbcols
6665 00002148 01C7                <1>       add    edi, eax ; dest+i*nbcols
6666 0000214A 88D8                <1>       mov    al, bl ; attr ; 04/08/2016
6667 0000214C 0FB6CF              <1>       movzx  ecx, bh ; cols
6668 0000214F F3AA                <1>       rep    stosb
6669 00002151 5F                  <1>       pop    edi
6670 00002152 75EA                <1>       jnz    short vgamem_fill_pl4_0
6671                              <1> vgamem_fill_pl4_1:
6672 00002154 59                  <1>       pop    ecx
6673                              <1>
6674                              <1>       ; outw(VGAREG_GRDC_ADDRESS, 0x0005);
6675 00002155 66B80500            <1>       mov    ax, 0005h
6676 00002159 66BACE03            <1>       mov    dx, 3CEh ; VGAREG_GRDC_ADDRESS
6677 0000215D 66EF                <1>       out    dx, ax
6678                              <1>
6679 0000215F 58                  <1>       pop    eax
6680 00002160 5A                  <1>       pop    edx
6681                              <1>
6682 00002161 C3                  <1>       retn
6683                              <1>
6684                              <1> vgamem_copy_l8:
6685                              <1>       ; 08/08/2016
6686                              <1>       ; 07/08/2016
6687                              <1>       ; 06/08/2016
6688                              <1>       ; 07/07/2016 - TRDOS 386 (TRDOS v2.0)
6689                              <1>       ;
6690                              <1>       ; TEMPORARY
6691                              <1>       ;
6692                              <1>       ; derived from 'Plex86/Bochs VGABios' source code
6693                              <1>       ; vgabios-0.7a (2011)
6694                              <1>       ; by the LGPL VGABios developers Team (2001-2008)
6695                              <1>       ; 'vgabios.c', 'vgamem_copy_pl4'
6696                              <1>       ;
6697                              <1>       ; vgamem_copy_pl4(xstart,ysrc,ydest,cols,nbcols,cheight)
6698                              <1>       ; cl = xstart, ah = ysrc (i+nblines), ch = ydest (i),
6699                              <1>       ; bh = cols, [CRT_COLS] = nbcols, [CHAR_HEIGHT] = cheight
6700                              <1>
6701                              <1>       ; src=ysrc*cheight*nbcols+xstart;
6702                              <1>       ; dest=ydest*cheight*nbcols+xstart;
6703                              <1>
6704 00002162 52                  <1>       push   edx
6705 00002163 50                  <1>       push   eax
6706                              <1>
6707                              <1>       ; outw(VGAREG_GRDC_ADDRESS, 0x0105)
6708                              <1>       ;mov   ax, 0105h
6709                              <1>       ;mov   dx, 3CEh ; VGAREG_GRDC_ADDRESS
6710                              <1>       ;out   dx, ax
6711                              <1>
6712                              <1>       ;mov   ah, [esp+1]
6713                              <1>
6714 00002164 0FB6D4              <1>       movzx  edx, ah ; ysrc
6715                              <1>       ; 08/08/2016
6716 00002167 0FB605[C65E0000]    <1>       movzx  eax, byte [CHAR_HEIGHT]
6717 0000216E 8A25[C45E0000]      <1>       mov    ah, [CRT_COLS] ; nbcols
6718 00002174 F6E4                <1>       mul    ah
6719                              <1>       ;; 07/08/2016
6720                              <1>       ;movzx eax, byte [CRT_COLS]
6721                              <1>       ;shl   ax, 3 ; * 8 ; * [CHAR_HEIGHT]
6722 00002176 50                  <1>       push   eax ; cheight * nbcols
6723 00002177 F7E2                <1>       mul    edx ; * ysrc
6724                              <1>       ; eax = ysrc * cheight * nbcols
6725                              <1>       ; edx = 0
6726 00002179 88CA                <1>       mov    dl, cl ; edx = xstart
6727 0000217B 01D0                <1>       add    eax, edx
6728 0000217D 89C6                <1>       mov    esi, eax ; src
6729 0000217F 66C1E603            <1>       shl    si, 3 ; * 8 ; 06/08/2016
6730 00002183 88EA                <1>       mov    dl, ch ; ydest
```

```
6731 00002185 58                <1>       pop    eax ; cheight * nbcols
6732 00002186 F7E2              <1>       mul    edx
6733                            <1>       ; eax = ydest * cheight * nbcols
6734 00002188 88CA              <1>       mov    dl, cl ; edx = xstart
6735 0000218A 01D0              <1>       add    eax, edx
6736 0000218C 89C7              <1>       mov    edi, eax ; dest
6737 0000218E 66C1E703          <1>       shl    di, 3 ; * 8 ; 06/08/2016
6738                            <1>       ; esi = src
6739                            <1>       ; edi = dest
6740                            <1>       ; for(i=0;i<cheight;i++)
6741                            <1>       ; {
6742                            <1>       ;  memcpyb(0xa000,dest+i*nbcols,0xa000,src+i*nbcols,cols);
6743                            <1>       ; }
6744 00002192 51                <1>       push   ecx
6745 00002193 B900000A00        <1>       mov    ecx, 0A0000h
6746 00002198 01CE              <1>       add    esi, ecx
6747 0000219A 01CF              <1>       add    edi, ecx
6748                            <1>       ; 08/08/2016
6749 0000219C 8A35[C65E0000]    <1>       mov    dh, [CHAR_HEIGHT]
6750                            <1>       ;; 07/08/2016
6751                            <1>       ;mov   dh, 8 ; 07/08/2016
6752 000021A2 28D2              <1>       sub    dl, dl ; i
6753                            <1> vgamem_copy_l8_0:
6754 000021A4 56                <1>       push   esi
6755 000021A5 57                <1>       push   edi
6756 000021A6 0FB605[C45E0000]  <1>       movzx  eax, byte [CRT_COLS]
6757 000021AD F6E2              <1>       mul    dl
6758                            <1>       ; eax = i * nbcols
6759 000021AF 66C1E003          <1>       shl    ax, 3 ; * 8 ; 06/08/2016
6760 000021B3 01C7              <1>       add    edi, eax ; dest+i*nbcols
6761 000021B5 01C6              <1>       add    esi, eax
6762 000021B7 0FB6CF            <1>       movzx  ecx, bh ; cols
6763 000021BA 66C1E103          <1>       shl    cx, 3 ; * 8 ; 06/08/2016
6764 000021BE F3A4              <1>       rep    movsb
6765 000021C0 5F                <1>       pop    edi
6766 000021C1 5E                <1>       pop    esi
6767 000021C2 FEC2              <1>       inc    dl ; 06/08/2016
6768 000021C4 FECE              <1>       dec    dh
6769 000021C6 75DC              <1>       jnz    short vgamem_copy_l8_0
6770                            <1> vgamem_copy_l8_1:
6771 000021C8 59                <1>       pop    ecx
6772                            <1>
6773                            <1>       ;; outw(VGAREG_GRDC_ADDRESS, 0x0005);
6774                            <1>       ;mov   ax, 0005h
6775                            <1>       ;mov   dx, 3CEh ; VGAREG_GRDC_ADDRESS
6776                            <1>       ;out   dx, ax
6777                            <1>
6778 000021C9 58                <1>       pop    eax
6779 000021CA 5A                <1>       pop    edx
6780                            <1>
6781 000021CB C3                <1>       retn
6782                            <1>
6783                            <1> vgamem_fill_l8:
6784                            <1>       ; 08/08/2016
6785                            <1>       ; 07/08/2016
6786                            <1>       ; 06/08/2016
6787                            <1>       ; 04/08/2016
6788                            <1>       ; 07/07/2016 - TRDOS 386 (TRDOS v2.0)
6789                            <1>       ;
6790                            <1>       ; TEMPORARY
6791                            <1>       ;
6792                            <1>       ; derived from 'Plex86/Bochs VGABios' source code
6793                            <1>       ; vgabios-0.7a (2011)
6794                            <1>       ; by the LGPL VGABios developers Team (2001-2008)
6795                            <1>       ; 'vgabios.c', 'vgamem_fill_pl4'
6796                            <1>       ;
6797                            <1>       ; vgamem_fill_pl4(xstart,ystart,cols,nbcols,cheight,attr)
6798                            <1>       ; cl = xstart, edi = ch = ystart, bh = cols,
6799                            <1>       ; [CRT_COLS] = nbcols, [CHAR_HEIGHT] = cheight, attr = 0
6800                            <1>
6801                            <1>       ; dest=ystart*cheight*nbcols+xstart;
6802 000021CC 52                <1>       push   edx
6803 000021CD 50                <1>       push   eax
6804                            <1>
6805                            <1>       ;; outw(VGAREG_GRDC_ADDRESS, 0x0205)
6806                            <1>       ;mov   ax, 0205h
6807                            <1>       ;mov   dx, 3CEh ; VGAREG_GRDC_ADDRESS
6808                            <1>       ;out   dx, ax
6809                            <1>
6810                            <1>         ; 08/08/2016
6811 000021CE 0FB605[C65E0000]  <1>       movzx   eax, byte [CHAR_HEIGHT]
6812 000021D5 F6E5              <1>       mul    ch
6813                            <1>       ;; 07/08/2016
6814                            <1>       ;movzx eax, ch
6815                            <1>       ;shl   ax, 3 ; * 8 ; * [CHAR_HEIGHT]
6816 000021D7 0FB615[C45E0000]  <1>       movzx  edx, byte [CRT_COLS] ; = [VGA_COLS]
6817 000021DE F7E2              <1>       mul    edx
6818                            <1>       ; edx  = 0
6819 000021E0 88CA              <1>       mov    dl, cl
6820 000021E2 01D0              <1>       add    eax, edx
6821 000021E4 89C7              <1>       mov    edi, eax
6822 000021E6 66C1E703          <1>       shl    di, 3 ; * 8 ; 06/08/2016
6823                            <1>       ; edi = dest
6824                            <1>       ; for(i=0;i<cheight;i++)
6825                            <1>       ; {
6826                            <1>       ;  memsetb(0xa000,dest+i*nbcols,attr,cols);
6827                            <1>       ; }
6828 000021EA 81C700000A00        <1>       add    edi, 0A0000h
6829 000021F0 51                <1>       push   ecx
6830                            <1>       ; 08/08/2016
6831 000021F1 8A35[C65E0000]    <1>       mov    dh, [CHAR_HEIGHT]
6832                            <1>       ;; 07/08/2016
```

```
                                <1>       ;mov   dh, 8 ; 07/08/2016
000021F7 28D2                   <1>       sub    dl, dl ; i
                                <1> vgamem_fill_l8_0:
000021F9 57                     <1>       push   edi
000021FA 0FB605[C45E0000]       <1>       movzx  eax, byte [CRT_COLS]
00002201 F6E2                   <1>       mul    dl
                                <1>       ; eax = i * nbcols
00002203 66C1E003               <1>       shl    ax, 3 ; * 8 ; 06/08/2016
00002207 01C7                   <1>       add    edi, eax ; dest+i*nbcols
00002209 88D8                   <1>       mov    al, bl ; attr ; 04/08/2016
0000220B 0FB6CF                 <1>       movzx  ecx, bh ; cols
0000220E 66C1E103               <1>       shl    cx, 3 ; * 8 ; 06/08/2016
00002212 F3AA                   <1>       rep    stosb
00002214 5F                     <1>       pop    edi
00002215 FEC2                   <1>       inc    dl ; 06/08/2016
00002217 FECE                   <1>       dec    dh
00002219 75DE                   <1>       jnz    short vgamem_fill_l8_0
                                <1> vgamem_fill_l8_1:
0000221B 59                     <1>       pop    ecx
                                <1>
                                <1>       ;; outw(VGAREG_GRDC_ADDRESS, 0x0005);
                                <1>       ;mov   ax, 0005h
                                <1>       ;mov   dx, 3CEh ; VGAREG_GRDC_ADDRESS
                                <1>       ;out   dx, ax
                                <1>
0000221C 58                     <1>       pop    eax
0000221D 5A                     <1>       pop    edx
                                <1>
0000221E C3                     <1>       retn
                                <1>
                                <1> vga_graphics_down:
                                <1>       ; 08/08/2016
                                <1>       ; 07/08/2016
                                <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>
0000221F FC                     <1>         cld     ; !!! Clear direction flag !!!
                                <1>
00002220 88C7                   <1>       mov    bh, al ; 31/07/2016
                                <1>
00002222 BE[DE5E0000]           <1>       mov    esi, vga_modes
00002227 89F7                   <1>       mov    edi, esi
00002229 83C710                 <1>       add    edi, vga_mode_count
                                <1> vga_g_down_0:
0000222C AC                     <1>       lodsb
0000222D 38E0                   <1>       cmp    al, ah ; [CRT_MODE]
0000222F 7405                   <1>       je     short vga_g_down_1
00002231 39FE                   <1>       cmp    esi, edi
00002233 72F7                   <1>       jb     short vga_g_down_0
                                <1>       ; xor  bh, bh ; 31/07/2016
00002235 C3                     <1>       retn   ; nothing to do
                                <1> vga_g_down_1:
00002236 88F8                   <1>       mov    al, bh ; 31/07/2016
00002238 83C64F                 <1>       add    esi, vga_memmodel - (vga_modes + 1)
                                <1>       ; [ESI] = VGA memory model number (LINEAR8, PLANAR4, PLANAR1)
                                <1>
                                <1>       ; if(rlr>=nbrows)rlr=nbrows-1;
                                <1>       ; if(clr>=nbcols)clr=nbcols-1;
                                <1>       ; if(nblines>nbrows)nblines=0;
                                <1>       ; cols=clr-cul+1;
                                <1>
0000223B 3A35[CA5E0000]         <1>       cmp    dh, [VGA_ROWS]
00002241 7208                   <1>       jb     short vga_g_down_2
00002243 8A35[CA5E0000]         <1>       mov    dh, [VGA_ROWS]
00002249 FECE                   <1>       dec    dh
                                <1> vga_g_down_2:
0000224B 3A15[C45E0000]         <1>       cmp    dl, [CRT_COLS]  ; = [VGA_COLS]
00002251 7208                   <1>       jb     short vga_g_down_3
00002253 8A15[C45E0000]         <1>       mov    dl, [CRT_COLS]
00002259 FECA                   <1>       dec    dl
                                <1> vga_g_down_3:
0000225B 3A05[CA5E0000]         <1>       cmp    al, [VGA_ROWS]
00002261 7602                   <1>       jna    short vga_g_down_4
00002263 28C0                   <1>       sub    al, al ; 0
                                <1> vga_g_down_4:
00002265 88F7                   <1>       mov    bh, dh ; clr
00002267 28CF                   <1>       sub    bh, cl ; cul
00002269 FEC7                   <1>       inc    bh ; cols = clr-cul+1
                                <1>
0000226B 20C0                   <1>       and    al, al ; nblines = 0
0000226D 755B                   <1>       jnz    short vga_g_down_6
0000226F 20ED                   <1>       and    ch, ch ; rul = 0
00002271 7557                   <1>       jnz    short vga_g_down_6
00002273 20C9                   <1>       and    cl, cl ; cul = 0
00002275 7553                   <1>       jnz    short vga_g_down_6
```

```
6935                                    <1>
6936 00002277 6650                      <1>       push   ax
6937 00002279 A0[CA5E0000]              <1>       mov    al, [VGA_ROWS]
6938 0000227E FEC8                      <1>       dec    al
6939 00002280 38C6                      <1>       cmp    dh, al ; rlr = nbrows-1
6940 00002282 7544                      <1>       jne    short vga_g_down_5
6941 00002284 A0[C45E0000]              <1>         mov     al, [CRT_COLS]  ; = VGA_COLS
6942 00002289 FEC8                      <1>       dec    al
6943 0000228B 38C2                      <1>       cmp    dl, al ; clr = nbcols-1
6944 0000228D 7539                      <1>       jne    short vga_g_down_5
6945 0000228F 6658                      <1>       pop    ax
6946                                    <1>
6947 00002291 66B80502                  <1>       mov    ax, 0205h
6948 00002295 66BACE03                  <1>       mov    dx, 3CEh ; VGAREG_GRDC_ADDRESS
6949 00002299 66EF                      <1>       out    dx, ax
6950 0000229B A0[CA5E0000]              <1>       mov    al, [VGA_ROWS]
6951 000022A0 8A25[C45E0000]            <1>       mov    ah, [CRT_COLS] ; = [VGA_COLS]
6952 000022A6 F6E4                      <1>       mul    ah
6953 000022A8 0FB7D0                    <1>       movzx  edx, ax
6954                                    <1>       ; 08/08/2016
6955 000022AB 0FB605[C65E0000]          <1>       movzx  eax, byte [CHAR_HEIGHT]
6956 000022B2 F7E2                      <1>       mul    edx
6957                                    <1>       ; eax = byte count
6958 000022B4 89C1                      <1>       mov    ecx, eax
6959                                    <1>       ;; 07/08/2016
6960                                    <1>       ;shl   dx, 3 ; * 8 ; * [CHAR_HEIGHT]
6961                                    <1>       ;mov   ecx, edx
6962 000022B6 88D8                      <1>       mov    al, bl ; fill value for blanked lines
6963 000022B8 BF00000A00                <1>       mov    edi, 0A0000h
6964 000022BD F3AA                      <1>       rep    stosb
6965                                    <1>
6966 000022BF B005                      <1>       mov    al, 5
6967 000022C1 66BACE03                  <1>       mov    dx, 3CEh ; VGAREG_GRDC_ADDRESS
6968 000022C5 66EF                      <1>       out    dx, ax ; 0005h
6969                                    <1>
6970 000022C7 C3                        <1>       retn
6971                                    <1>
6972                                    <1> vga_g_down_5:
6973 000022C8 6658                      <1>       pop    ax
6974                                    <1>
6975                                    <1> vga_g_down_6:
6976                                    <1>       ; [ESI] = VGA memory model number for current video mode
6977                                    <1>         ;
6978                                    <1>         ; LINEAR8 equ 5
6979                                    <1>         ; PLANAR4 equ 4
6980                                    <1>         ; PLANAR1 equ 3
6981                                    <1>
6982 000022CA 803E04                    <1>       cmp    byte [esi], PLANAR4
6983 000022CD 742C                      <1>       je     short vga_g_down_planar
6984 000022CF 803E03                    <1>       cmp    byte [esi], PLANAR1
6985 000022D2 7427                      <1>       je     short vga_g_down_planar
6986                                    <1> vga_g_down_linear8:
6987                                    <1>       ; 07/07/2016 (TEMPORARY)
6988                                    <1>       ;
6989                                    <1>       ; cl = upper left column ; cul
6990                                    <1>       ; ch = upper left row ; rul
6991                                    <1>       ; dl = lower rigth column ; clr
6992                                    <1>       ; dh = lower right row ; rlr
6993                                    <1>
6994                                    <1> vga_g_down_l0:
6995                                    <1>       ;{for(i=rlr;i>=rul;i--)
6996                                    <1>         ; if((i<rul+nblines)||(nblines==0))
6997 000022D4 08C0                      <1>       or     al, al
6998 000022D6 741C                      <1>       jz     short vga_g_down_l2
6999 000022D8 88C4                      <1>       mov    ah, al
7000 000022DA 00EC                      <1>       add    ah, ch
7001                                    <1>       ;jc    short vga_g_down_l2
7002 000022DC 86EE                      <1>       xchg   ch, dh
7003 000022DE 38E5                      <1>       cmp    ch, ah
7004 000022E0 7212                      <1>       jb     short vga_g_down_l2
7005 000022E2 88EC                      <1>       mov    ah, ch
7006 000022E4 28C4                      <1>       sub    ah, al ; ah = i - nblines
7007                                    <1>       ; else
7008                                    <1>       ; vgamem_copy_pl4(cul,i,i-nblines,cols,nbcols,cheight);
7009 000022E6 E877FEFFFF                <1>       call   vgamem_copy_l8
7010                                    <1> vga_g_down_l1:
7011 000022EB 86F5                      <1>       xchg   dh, ch
7012 000022ED FECE                      <1>       dec    dh
7013 000022EF 38EE                      <1>       cmp    dh, ch
7014 000022F1 73E1                      <1>       jnb    short vga_g_down_l0
7015 000022F3 C3                        <1>       retn
7016                                    <1>
7017                                    <1> vga_g_down_l2:
7018                                    <1>       ; vgamem_fill_pl4(cul,i,cols,nbcols,cheight,attr);
7019 000022F4 E8D3FEFFFF                <1>       call   vgamem_fill_l8
7020 000022F9 EBF0                      <1>       jmp    short vga_g_down_l1
7021                                    <1>
7022                                    <1> vga_g_down_planar:
7023                                    <1>       ; cl = upper left column ; cul
7024                                    <1>       ; ch = upper left row ; rul
7025                                    <1>       ; dl = lower rigth column ; clr
7026                                    <1>       ; dh = lower right row ; rlr
7027                                    <1> vga_g_down_pl0:
7028                                    <1>       ;{for(i=rlr;i>=rul;i--)
7029                                    <1>         ; if((i<rul+nblines)||(nblines==0))
7030 000022FB 08C0                      <1>       or     al, al
7031 000022FD 741C                      <1>       jz     short vga_g_down_pl2
7032 000022FF 88C4                      <1>       mov    ah, al
7033 00002301 00EC                      <1>       add    ah, ch
7034                                    <1>       ;jc    short vga_g_down_pl2
7035 00002303 86EE                      <1>       xchg   ch, dh
7036 00002305 38E5                      <1>       cmp    ch, ah
```

```
7037 00002307 7212              <1>       jb     short vga_g_down_pl2
7038 00002309 88EC              <1>       mov    ah, ch
7039 0000230B 28C4              <1>       sub    ah, al ; ah = i - nblines
7040                            <1>       ; else
7041                            <1>       ; vgamem_copy_pl4(cul,i,i-nblines,cols,nbcols,cheight);
7042 0000230D E88BFDFFFF        <1>       call   vgamem_copy_pl4
7043                            <1> vga_g_down_pl1:
7044 00002312 86F5              <1>       xchg   dh, ch
7045 00002314 FECE              <1>       dec    dh
7046 00002316 38EE              <1>       cmp    dh, ch
7047 00002318 73E1              <1>       jnb    short vga_g_down_pl0
7048 0000231A C3                <1>       retn
7049                            <1>
7050                            <1> vga_g_down_pl2:
7051                            <1>       ; vgamem_fill_pl4(cul,i,cols,nbcols,cheight,attr);
7052 0000231B E8EBFDFFFF        <1>       call   vgamem_fill_pl4
7053 00002320 EBF0              <1>       jmp    short vga_g_down_pl1
7054                            <1>
7055                            <1> ; 07/07/2016
7056                            <1> ; 27/06/2016 - TRDOS 386 (TRDOS v2.0)
7057                            <1> ; VIDEO1.ASM - 24/03/1985 (IBM PC-AT BIOS source code)
7058                            <1> ;--------------------------------------------------------
7059                            <1> ; SCROLL DOWN
7060                            <1> ;  THIS ROUTINE SCROLLS DOWN THE INFORMATION ON THE CRT
7061                            <1> ; ENTRY --
7062                            <1> ;  CH,CL = UPPER LEFT CORNER OF REGION TO SCROLL
7063                            <1> ;  DH,DL = LOWER RIGHT CORNER OF REGION TO SCROLL
7064                            <1> ;   BOTH OF THE ABOVE ARE IN CHARACTER POSITIONS
7065                            <1> ;  BH = FILL VALUE FOR BLANKED LINES
7066                            <1> ;  AL = # LINES TO SCROLL (AL=0 MEANS BLANK THE ENTIRE FIELD)
7067                            <1> ;  DS = DATA SEGMENT
7068                            <1> ;  ES = REGEN SEGMENT
7069                            <1> ; EXIT --
7070                            <1> ;  NOTHING, THE SCREEN IS SCROLLED
7071                            <1> ;--------------------------------------------------------
7072                            <1>
7073                            <1>       ; cl = upper left column
7074                            <1>       ; ch = upper left row
7075                            <1>       ; dl = lower rigth column
7076                            <1>       ; dh = lower right row
7077                            <1>       ;
7078                            <1>       ; al = line count (AL=0 means blank entire fields)
7079                            <1>       ; bl = fill value for blanked lines
7080                            <1>       ; bh = unused
7081                            <1>
7082                            <1> GRAPHICS_DOWN:
7083                            <1>       ; 07/07/2016
7084                            <1>       ;AH = Current video mode, [CRT_MODE]
7085                            <1>       ;STD                    ; SET DIRECTION
7086 00002322 80FC07            <1>       cmp    ah, 7
7087 00002325 0F87F4FEFFFF      <1>         ja     vga_graphics_down
7088                            <1>       ;je    _n0
7089                            <1>
7090 0000232B 88C7              <1>       MOV    bh, al           ; save line count in BH
7091 0000232D 6689D0            <1>       MOV    AX, DX           ; GET LOWER RIGHT POSITION INTO AX REG
7092                            <1>
7093                            <1> ;-----      USE CHARACTER SUBROUTINE FOR POSITIONING
7094                            <1> ;-----      ADDRESS RETURNED IS MULTIPLIED BY 2 FROM CORRECT VALUE
7095                            <1>
7096 00002330 E8F1010000        <1>       CALL   GRAPH_POSN
7097 00002335 0FB7F8            <1>       MOVzx  eDI, AX                  ; SAVE RESULT AS DESTINATION ADDRESS
7098                            <1>
7099                            <1> ;-----      DETERMINE SIZE OF WINDOW
7100                            <1>
7101 00002338 6629CA            <1>       SUB    DX, CX
7102 0000233B 6681C20101        <1>         ADD    DX, 101h               ; ADJUST VALUES
7103 00002340 C0E602            <1>       SAL    DH, 2            ; MULTIPLY ROWS BY 4 AT 8 VERT DOTS/CHAR
7104                            <1>                                ; AND EVEN/ODD ROWS
7105                            <1>
7106                            <1> ;-----      DETERMINE CRT MODE
7107                            <1>
7108 00002343 803D[C25E0000]06  <1>       CMP    byte [CRT_MODE], 6  ; TEST FOR MEDIUM RES
7109 0000234A 7307              <1>         JNC    short _R12              ; FIND_SOURCE_DOWN
7110                            <1>
7111                            <1> ;-----      MEDIUM RES DOWN
7112 0000234C D0E2              <1>       SAL    DL, 1            ; # COLUMNS * 2, SINCE 2 BYTES/CHAR
7113 0000234E 66D1E7            <1>       SAL    DI, 1            ; OFFSET *2 SINCE 2 BYTES/CHAR
7114 00002351 6647              <1>       INC    DI               ; POINT TO LAST BYTE
7115                            <1>
7116                            <1> ;-----      DETERMINE THE SOURCE ADDRESS IN THE BUFFER
7117                            <1>
7118                            <1> _R12:                                  ; FIND_SOURCE_DOWN
7119 00002353 81C700800B00      <1>       add    edi, 0B8000h
7120 00002359 6681C7F000        <1>       ADD    DI, 240                ; POINT TO LAST ROW OF PIXELS
7121 0000235E C0E702            <1>       sal    bh, 2            ; multiply number of lines by 4
7122 00002361 74(06)            <1>       JZ     short 6                ; IF ZERO, THEN BLANK ENTIRE FIELD
7123 00002363 B050              <1>       MOV    AL, 80           ; 80 BYTES/ROW
7124 00002365 F6E7              <1>       mul    bh               ; determine offset to source
7125 00002367 89FE              <1>       MOV    eSI, eDI         ; SET UP SOURCE
7126 00002369 6629C6            <1>       SUB    SI, AX           ; SUBTRACT THE OFFSET
7127 0000236C 88F4              <1>       MOV    AH, DH           ; NUMBER OF ROWS IN FIELD
7128 0000236E 28FC              <1>       sub    ah, bh           ; determine number to move
7129                            <1>
7130                            <1> ;-----      LOOP THROUGH, MOVING ONE ROW AT A TIME, BOTH EVEN AND ODD FIELDS
7131                            <1>
7132                            <1> _R13:                                  ; ROW_LOOP_DOWN
7133 00002370 E823000000        <1>         CALL   _R17                    ; MOVE ONE ROW
7134 00002375 6681EE5020        <1>       SUB    SI, 2000h+80     ; MOVE TO NEXT ROW
7135 0000237A 6681EF5020        <1>       SUB    DI, 2000h+80
7136 0000237F FECC              <1>       DEC    AH               ; NUMBER OF ROWS TO MOVE
7137 00002381 75ED              <1>         JNZ    short _R13              ; CONTINUE TILL ALL MOVED
7138                            <1>
```

```
7139                            <1> ;-----      FILL IN THE VACATED LINE(S)
7140                            <1> _R14:                                   ; CLEAR_ENTRY_DOWN
7141 00002383 88D8             <1>       mov   al, bl              ; attribute to fill with
7142                            <1> _R15_:                                  ; CLEAR_LOOP_DOWN
7143 00002385 E82A000000       <1>       CALL  _R18                ; CLEAR A ROW
7144 0000238A 6681EF5020       <1>       SUB   DI, 2000h+80        ; POINT TO NEXT LINE
7145 0000238F FECF             <1>       dec   bh                  ; number of lines to fill
7146 00002391 75F2             <1>         JNZ     short _R15_             ; CLEAR_LOOP_DOWN
7147                            <1>
7148 00002393 C3               <1>       retn                      ; EVERYYHING DONE
7149                            <1>
7150                            <1> _R16:                                   ; BLANK_FIELD_DOWN
7151 00002394 88F7             <1>       mov   bh, dh              ; set blank count to everything in field
7152 00002396 EBEB             <1>         JMP     short _R14              ; CLEAR THE FIELD
7153                            <1>
7154                            <1> ; 27/06/2016 - TRDOS 386 (TRDOS v2.0)
7155                            <1> ; VIDEO1.ASM - 24/03/1985 (IBM PC-AT BIOS source code)
7156                            <1>
7157                            <1> ;-----      ROUTINE TO MOVE ONE ROW OF INFORMATION
7158                            <1>
7159                            <1> _R17:
7160 00002398 0FB6CA           <1>       MOVzx ecx, DL                   ; NUMBER OF BYTES IN THE ROW
7161 0000239B 56               <1>       PUSH  eSI
7162 0000239C 57               <1>       PUSH  eDI                 ; SAVE POINTERS
7163 0000239D F3A4             <1>       REP   MOVSB               ; MOVE THE EVEN FIELD
7164 0000239F 5F               <1>       POP   eDI
7165 000023A0 5E               <1>       POP   eSI
7166 000023A1 6681C60020       <1>       ADD   SI, 2000h
7167 000023A6 6681C70020       <1>       ADD   DI, 2000h           ; POINT TO THE ODD FIELD
7168 000023AB 56               <1>       PUSH  eSI
7169 000023AC 57               <1>       PUSH  eDI                 ; SAVE THE POINTERS
7170 000023AD 88D1             <1>       MOV   CL, DL              ; COUNT BACK
7171 000023AF F3A4             <1>       REP   MOVSB               ; MOVE THE ODD FIELD
7172 000023B1 5F               <1>       POP   eDI
7173 000023B2 5E               <1>       POP   eSI                 ; POINTERS BACK
7174 000023B3 C3               <1>       RETn                      ; RETURN TO CALLER
7175                            <1>
7176                            <1> ;-----      CLEAR A SINGLE ROW
7177                            <1>
7178                            <1> _R18:
7179 000023B4 0FB6CA           <1>       MOVzx ecx, DL                   ; NUMBER OF BYTES IN FIELD
7180 000023B7 57               <1>       PUSH  eDI                 ; SAVE POINTER
7181 000023B8 F3AA             <1>       REP   STOSB               ; STORE THE NEW VALUE
7182 000023BA 5F               <1>       POP   eDI                 ; POINTER BACK
7183 000023BB 6681C70020       <1>       ADD   DI, 2000h           ; POINT TO ODD FIELD
7184 000023C0 57               <1>       PUSH  eDI
7185 000023C1 88D1             <1>       MOV   CL, DL
7186 000023C3 F3AA             <1>       REP   STOSB               ; FILL THE ODD FIELD
7187 000023C5 5F               <1>       POP   eDI
7188 000023C6 C3               <1>       RETn                      ; RETURN TO CALLER
7189                            <1>
7190                            <1> ; 04/07/2016
7191                            <1> ; 01/07/2016
7192                            <1> ; 30/06/2016 - TRDOS 386 (TRDOS v2.0)
7193                            <1> ; VIDEO1.ASM - 24/03/1985 (IBM PC-AT BIOS source code)
7194                            <1> ;-----------------------------------------------
7195                            <1> ; GRAPHICS WRITE
7196                            <1> ;  THIS ROUTINE WRITES THE ASCII CHARACTER TO THE CURRENT
7197                            <1> ;  POSITION ON THE SCREEN.
7198                            <1> ; ENTRY --
7199                            <1> ;  AL = CHARACTER TO WRITE
7200                            <1> ;  BL = COLOR ATTRIBUTE TO BE USED FOR FOREGROUND COLOR
7201                            <1> ;     IF BIT 7 IS SET, THE CHAR IS XOR'D INTO THE REGEN BUFFER
7202                            <1> ;     (0 IS USED FOR THE BACKGROUND COLOR)
7203                            <1> ;  CX = NUMBER OF CHARS TO WRITE
7204                            <1> ;  DS = DATA SEGMENT
7205                            <1> ;  ES = REGEN SEGMENT
7206                            <1> ; EXIT --
7207                            <1> ;  NOTHING IS RETURNED
7208                            <1> ;
7209                            <1> ; GRAPHICS READ
7210                            <1> ;  THIS ROUTINE READS THE ASCII CHARACTER AT THE CURRENT CURSOR
7211                            <1> ;  POSITION ON THE SCREEN BY MATCHING THE DOTS ON THE SCREEN TO THE
7212                            <1> ;  CHARACTER GENERATOR CODE POINTS
7213                            <1> ; ENTRY --
7214                            <1> ;  NONE (0 IS ASSUMED AS THE BACKGROUND COLOR)
7215                            <1> ; EXIT --
7216                            <1> ;  AL = CHARACTER READ AT THAT POSITION (0 RETURNED IF NONE FOUND)
7217                            <1> ;
7218                            <1> ; FOR BOTH ROUTINES, THE IMAGES USED TO FORM CHARS ARE CONTAINED IN ROM
7219                            <1> ;  FOR THE 1ST 128 CHARS.  TO ACCESS CHARS IN THE SECOND HALF, THE USER
7220                            <1> ;  MUST INITIALIZE THE VECTOR AT INTERRUPT 1FH (LOCATION 0007CH) TO
7221                            <1> ;  POINT TO THE USER SUPPLIED TABLE OF GRAPHIC IMAGES (8X8 BOXES).
7222                            <1> ;  FAILURE TO DO SO WILL CAUSE IN STRANGE RESULTS
7223                            <1> ;-------------------------------------------------
7224                            <1>
7225                            <1> GRAPHICS_WRITE:
7226 000023C7 25FF000000       <1>       and   eax, 0FFh           ; ZERO TO HIGH OF CODE POINT
7227 000023CC 50               <1>       PUSH  eAX                 ; SAVE CODE POINT VALUE
7228                            <1>
7229                            <1> ;-----      DETERMINE POSITION IN REGEN BUFFER TO PUT CODE POINTS
7230                            <1>
7231 000023CD E84D010000       <1>       CALL  S26                 ; FIND LOCATION IN REGEN BUFFER
7232 000023D2 89C7             <1>       MOV   eDI, eAX            ; REGEN POINTER IN DI
7233                            <1>
7234                            <1> ;-----      DETERMINE REGION TO GET CODE POINTS FROM
7235                            <1>
7236 000023D4 58               <1>       POP   eAX                 ; RECOVER CODE POINT
7237                            <1>
7238 000023D5 BE[44220100]     <1>       MOV   eSI, CRT_CHAR_GEN   ; OFFSET OF IMAGES
7239                            <1>
7240                            <1> ;-----      DETERMINE GRAPHICS MODE IN OPERATION
```

```
7241                                    <1>                             ; DETERMINE_MODE
7242 000023DA 66C1E003                  <1>     SAL     AX, 3           ; MULTIPLY CODE POINT VALUE BY 8
7243 000023DE 01C6                      <1>     ADD     eSI, eAX        ; SI HAS OFFSET OF DESIRED CODES
7244                                    <1>
7245 000023E0 803D[C25E0000]06          <1>     CMP     byte [CRT_MODE], 6
7246 000023E7 7231                      <1>     JC      short S6        ; TEST FOR MEDIUM RESOLUTION MODE
7247                                    <1>
7248                                    <1> ;-----      HIGH RESOLUTION MODE
7249                                    <1>
7250 000023E9 81C700800B00              <1>     add     edi, 0B8000h
7251                                    <1> S1:                         ; HIGH_CHAR
7252 000023EF 57                        <1>     PUSH    eDI             ; SAVE REGEN POINTER
7253 000023F0 56                        <1>     PUSH    eSI             ; SAVE CODE POINTER
7254 000023F1 B604                      <1>     MOV     DH, 4           ; NUMBER OF TIMES THROUGH LOOP
7255                                    <1> S2:
7256 000023F3 AC                        <1>     LODSB                   ; GET BYTE FROM CODE POINTS
7257 000023F4 F6C380                    <1>     TEST    BL, 80H                 ; SHOULD WE USE THE FUNCTION
7258 000023F7 7515                      <1>     JNZ     short S5        ; TO PUT CHAR IN
7259 000023F9 AA                        <1>     STOSB                   ; STORE IN REGEN BUFFER
7260 000023FA AC                        <1>     LODSB
7261                                    <1> S4:
7262 000023FB 8887FF1F0000              <1>     MOV     [eDI+2000H-1], AL ; STORE IN SECOND HALF
7263 00002401 83C74F                    <1>     ADD     eDI, 79                 ; MOVE TO NEXT ROW IN REGEN
7264 00002404 FECE                      <1>     DEC     DH              ; DONE WITH LOOP
7265 00002406 75EB                      <1>     JNZ     short S2
7266 00002408 5E                        <1>     POP     eSI
7267 00002409 5F                        <1>     POP     eDI             ; RECOVER REGEN POINTER
7268 0000240A 47                        <1>     INC     eDI             ; POINT TO NEXT CHAR POSITION
7269 0000240B E2E2                      <1>     LOOP    S1              ; MORE CHARS TO WRITE
7270 0000240D C3                        <1>     retn
7271                                    <1>
7272                                    <1> S5:
7273 0000240E 3207                      <1>     XOR     AL, [eDI]       ; EXCLUSIVE OR WITH CURRENT
7274 00002410 AA                        <1>     STOSB                   ; STORE THE CODE POINT
7275 00002411 AC                        <1>     LODSB                   ; AGAIN FOR ODD FIELD
7276 00002412 3287FF1F0000              <1>     XOR     AL, [eDI+2000H-1]
7277 00002418 EBE1                      <1>     JMP     short S4        ; BACK TO MAINSTREAM
7278                                    <1>
7279                                    <1> ;-----      MEDIUM RESOLUTION WRITE
7280                                    <1> S6:                         ; MED_RES_WRITE
7281 0000241A 88DA                      <1>     MOV     DL, BL          ; SAVE HIGH COLOR BIT
7282 0000241C 66D1E7                    <1>     SAL     DI, 1           ; OFFSET*2 SINCE 2 BYTES/CHAR
7283                                    <1>                             ; EXPAND BL TO FULL WORD OF COLOR
7284 0000241F 80E303                    <1>     AND     BL, 3           ; ISOLATE THE COLOR BITS ( LOW 2 BITS )
7285 00002422 B055                      <1>     MOV     AL, 055H        ; GET BIT CONVERSION MULTIPLIER
7286 00002424 F6E3                      <1>     MUL     BL              ; EXPAND 2 COLOR BITS TO 4 REPLICATIONS
7287 00002426 88C3                      <1>     MOV     BL, AL          ; PLACE BACK IN WORK REGISTER
7288 00002428 88C7                      <1>     MOV     BH, AL          ; EXPAND TO 8 REPLICATIONS OF COLOR BITS
7289 0000242A 81C700800B00              <1>     add     edi, 0B8000h
7290                                    <1> S7:                                 ; MED_CHAR
7291 00002430 57                        <1>     PUSH    eDI             ; SAVE REGEN POINTER
7292 00002431 56                        <1>     PUSH    eSI             ; SAVE THE CODE POINTER
7293 00002432 B604                      <1>     MOV     DH, 4           ; NUMBER OF LOOPS
7294                                    <1> S8:
7295 00002434 AC                        <1>     LODSB                   ; GET CODE POINT
7296 00002435 E8B3000000                <1>     CALL    S21             ; DOUBLE UP ALL THE BITS
7297 0000243A 6621D8                    <1>     AND     AX, BX          ; CONVERT TO FOREGROUND COLOR ( 0 BACK )
7298 0000243D 86E0                      <1>     XCHG    AH, AL          ; SWAP HIGH/LOW BYTES FOR WORD MOVE
7299 0000243F F6C280                    <1>     TEST    DL, 80H                 ; IS THIS XOR FUNCTION
7300 00002442 7403                      <1>     JZ      short S9        ; NO, STORE IT IN AS IS
7301 00002444 663307                    <1>     XOR     AX, [eDI]       ; DO FUNCTION WITH LOW/HIGH
7302                                    <1> S9:
7303 00002447 668907                    <1>     MOV     [eDI], AX       ; STORE FIRST BYTE HIGH, SECOND LOW
7304 0000244A AC                        <1>     LODSB                   ; GET CODE POINT
7305 0000244B E89D000000                <1>     CALL    S21
7306 00002450 6621D8                    <1>     AND     AX, BX          ; CONVERT TO COLOR
7307 00002453 86E0                      <1>     XCHG    AH, AL          ; SWAP HIGH/LOW BYTES FOR WORD MOVE
7308 00002455 F6C280                    <1>     TEST    DL, 80H                 ; AGAIN, IS THIS XOR FUNCTION
7309 00002458 7407                      <1>     JZ      short _S10      ; NO, JUST STORE THE VALUES
7310 0000245A 66338700200000            <1>     XOR     AX, [eDI+2000H]         ; FUNCTION WITH FIRST HALF LOW
7311                                    <1> _S10:
7312 00002461 66898700200000            <1>     MOV     [eDI+2000H], AX         ; STORE SECOND PORTION HIGH
7313 00002468 6683C750                  <1>     ADD     DI, 80          ; POINT TO NEXT LOCATION
7314 0000246C FECE                      <1>     DEC     DH
7315 0000246E 75C4                      <1>     JNZ     short S8        ; KEEP GOING
7316 00002470 5E                        <1>     POP     eSI             ; RECOVER CODE POINTER
7317 00002471 5F                        <1>     POP     eDI             ; RECOVER REGEN POINTER
7318 00002472 47                        <1>     INC     eDI             ; POINT TO NEXT CHAR POSITION
7319 00002473 47                        <1>     INC     eDI
7320 00002474 E2BA                      <1>     LOOP    S7              ; MORE TO WRITE
7321 00002476 C3                        <1>     retn
7322                                    <1>
7323                                    <1> ; 04/07/2016
7324                                    <1> ; 01/07/2016
7325                                    <1> ; 30/06/2016 - TRDOS 386 (TRDOS v2.0)
7326                                    <1> ; VIDEO1.ASM - 24/03/1985 (IBM PC-AT BIOS source code)
7327                                    <1> ;----------------------------------------
7328                                    <1> ; GRAPHICS READ
7329                                    <1> ;----------------------------------------
7330                                    <1> GRAPHICS_READ:
7331 00002477 E8A3000000                <1>     CALL    S26             ; CONVERTED TO OFFSET IN REGEN
7332 0000247C 89C6                      <1>     MOV     eSI, eAX        ; SAVE IN SI
7333 0000247E 81C600800B00              <1>     add     esi, 0B8000h    ; 01/07/2016
7334 00002484 83EC08                    <1>     SUB     eSP, 8          ; ALLOCATE SPACE FOR THE READ CODE POINT
7335 00002487 89E5                      <1>     MOV     eBP, eSP        ; POINTER TO SAVE AREA
7336                                    <1>
7337                                    <1> ;-----      DETERMINE GRAPHICS MODES
7338 00002489 B604                      <1>     mov     dh, 4           ; number of passes ; 01/07/2016
7339 0000248B 803D[C25E0000]06          <1>     CMP     byte [CRT_MODE], 6
7340 00002492 7219                      <1>     JC      short S12       ; MEDIUM RESOLUTION
7341                                    <1>
7342                                    <1> ;-----      HIGH RESOLUTION READ
```

```
7343                            <1> ;-----      GET VALUES FROM REGEN BUFFER AND CONVERT TO CODE POINT
7344                            <1>       ;MOV  DH,4                ; NUMBER OF PASSES
7345                            <1> S11:
7346 00002494 8A06              <1>       MOV   AL, [eSI]           ; GET FIRST BYTE
7347 00002496 884500            <1>       MOV   [eBP], AL           ; SAVE IN STORAGE AREA
7348 00002499 45                <1>       INC   eBP                 ; NEXT LOCATION
7349 0000249A 8A8600200000      <1>       MOV   AL, [eSI+2000H]           ; GET LOWER REGION BYTE
7350 000024A0 884500            <1>       MOV   [eBP], AL           ; ADJUST AND STORE
7351 000024A3 45                <1>       INC   eBP
7352 000024A4 83C650            <1>       ADD   eSI, 80                   ; POINTER INTO REGEN
7353 000024A7 FECE              <1>       DEC   DH                  ; LOOP CONTROL
7354 000024A9 75E9              <1>       JNZ   short S11           ; DO IT SOME MORE
7355 000024AB EB1D              <1>       JMP   SHORT S14           ; GO MATCH THE SAVED CODE POINTS
7356                            <1>
7357                            <1> ;-----      MEDIUM RESOLUTION READ
7358                            <1> S12:
7359 000024AD 66D1E6            <1>       SAL   SI, 1               ; OFFSET*2 SINCE 2 BYTES/CHAR
7360                            <1>       ;MOV  DH, 4               ; NUMBER OF PASSES
7361                            <1> S13:
7362 000024B0 E84D000000        <1>       CALL  S23                 ; GET BYTES FROM REGEN INTO SINGLE SAVE
7363 000024B5 81C6FE1F0000      <1>       ADD   eSI, 2000H-2        ; GO TO LOWER REGION
7364 000024BB E842000000        <1>       CALL  S23                 ; GET THIS PAIR INTO SAVE
7365 000024C0 81EEB21F0000      <1>       SUB   eSI, 2000H-80+2           ; ADJUST POINTER BACK INTO UPPER
7366 000024C6 FECE              <1>       DEC   DH
7367 000024C8 75E6              <1>       JNZ   short S13           ; KEEP GOING UNTIL ALL 8 DONE
7368                            <1>
7369                            <1> ;-----      SAVE AREA HAS CHARACTER IN IT, MATCH IT
7370                            <1> S14:                            ; FIND_CHAR
7371 000024CA BF[44220100]      <1>       MOV   eDI, CRT_CHAR_GEN   ; ESTABLISH ADDRESSING
7372 000024CF 83ED08            <1>       SUB   eBP, 8              ; ADJUST POINTER TO START OF SAVE AREA
7373 000024D2 89EE              <1>       MOV   eSI, eBP
7374                            <1> S15:
7375 000024D4 66B80001          <1>       mov   ax, 256                   ; NUMBER TO TEST AGAINST
7376                            <1> S16:
7377 000024D8 56                <1>       PUSH  eSI                 ; SAVE SAVE AREA POINTER
7378 000024D9 57                <1>       PUSH  eDI                 ; SAVE CODE POINTER
7379                            <1>       ;MOV  eCX, 4              ; NUMBER OF WORDS TO MATCH
7380                            <1>       ;REPE CMPSW               ; COMPARE THE 8 BYTES AS WORDS
7381 000024DA A7                <1>       cmpsd                     ; compare first 4 bytes
7382 000024DB 7501              <1>       jne   short S17           ;
7383 000024DD A7                <1>       cmpsd                     ; compare last 4 bytes
7384                            <1> S17:
7385 000024DE 5F                <1>       POP   eDI                 ; RECOVER THE POINTERS
7386 000024DF 5E                <1>       POP   eSI
7387                            <1>       ;JZ   short S18           ; IF ZERO FLAG SET, THEN MATCH OCCURRED
7388 000024E0 7407              <1>       je    short S18
7389                            <1>       ;                         ; NO MATCH, MOVE ON TO NEXT
7390 000024E2 83C708            <1>       ADD   eDI, 8              ; NEXT CODE POINT
7391 000024E5 6648              <1>       dec   ax                  ; LOOP CONTROL
7392 000024E7 75EF              <1>       JNZ   short S16           ; DO ALL OF THEM
7393                            <1>
7394                            <1> ;-----      CHARACTER IS FOUND ( AL=0 IF NOT FOUND )
7395                            <1> S18:
7396 000024E9 83C408            <1>       ADD   eSP, 8              ; READJUST THE STACK, THROW AWAY SAVE
7397 000024EC C3                <1>       retn                      ; ALL DONE
7398                            <1>
7399                            <1> ; 30/06/2016 - TRDOS 386 (TRDOS v2.0)
7400                            <1> ; VIDEO1.ASM - 24/03/1985 (IBM PC-AT BIOS source code)
7401                            <1> ;-----------------------------------------
7402                            <1> ; EXPAND BYTE
7403                            <1> ;  THIS ROUTINE TAKES THE BYTE IN AL AND DOUBLES ALL
7404                            <1> ;  OF THE BITS, TURNING THE 8 BITS INTO 16 BITS.
7405                            <1> ;  THE RESULT IS LEFT IN AX
7406                            <1> ;-----------------------------------------
7407                            <1> S21:
7408 000024ED 6651              <1>       PUSH  CX                  ; SAVE REGISTER
7409                            <1>       ;MOV  CX, 8               ; SHIFT COUNT REGISTER FOR ONE BYTE
7410 000024EF B108              <1>       mov   cl, 8
7411                            <1> S22:
7412 000024F1 D0C8              <1>       ROR   AL,1                ; SHIFT BITS, LOW BIT INTO CARRY FLAG
7413 000024F3 66D1DD            <1>       RCR   BP,1                ; MOVE CARRY FLAG (LOW BIT INTO RESULTS
7414 000024F6 66D1FD            <1>       SAR   BP,1                ; SIGN EXTEND HIGH BIT (DOUBLE IT)
7415                            <1>       ;LOOP S22                 ; REPEAT FOR ALL 8 BITS
7416 000024F9 FEC9              <1>       dec   cl
7417 000024FB 75F4              <1>       jnz   short S22
7418 000024FD 6695              <1>       XCHG  AX, BP              ; MOVE RESULTS TO PARAMETER REGISTER
7419 000024FF 6659              <1>       POP   CX                  ; RECOVER REGISTER
7420 00002501 C3                <1>       RETn                      ; ALL DONE
7421                            <1>
7422                            <1> ; 01/07/2016 - TRDOS 386 (TRDOS v2.0)
7423                            <1> ; VIDEO1.ASM - 24/03/1985 (IBM PC-AT BIOS source code)
7424                            <1> ;-------------------------------------------------
7425                            <1> ; MED_READ_BYTE
7426                            <1> ; THIS ROUTINE WILL TAKE 2 BYTES FROM THE REGEN BUFFER,
7427                            <1> ;  COMPARE AGAINST THE CURRENT FOREGROUND COLOR, AND PLACE
7428                            <1> ;  THE CORRESPONDING ON/OFF BIT PATTERN INTO THE CURRENT
7429                            <1> ;  POSITION IN THE SAVE AREA
7430                            <1> ; ENTRY --
7431                            <1> ;  SI,DS = POINTER TO REGEN AREA OF INTEREST
7432                            <1> ;  BX = EXPANDED FOREGROUND COLOR
7433                            <1> ;  BP = POINTER TO SAVE AREA
7434                            <1> ; EXIT --
7435                            <1> ;  SI AND BP ARE INCREMENTED
7436                            <1> ;-------------------------------------------------
7437                            <1> S23:
7438 00002502 66AD              <1>       LODSW                     ; GET FIRST BYTE AND SECOND BYTES
7439 00002504 86C4              <1>       XCHG  AL, AH              ; SWAP FOR COMPARE
7440 00002506 66B900C0          <1>       MOV   CX, 0C000H          ; 2 BIT MASK TO TEST THE ENTRIES
7441 0000250A B200              <1>       MOV   DL, 0               ; RESULT REGISTER
7442                            <1> S24:
7443 0000250C 6685C8            <1>       TEST  AX, CX              ; IS THIS SECTION BACKCROUND?
7444 0000250F 7401              <1>         JZ      short S25               ; IF ZERO, IT IS BACKGROUND (CARRY=0)
```

```
7445 00002511 F9                <1>       STC                     ; WASN'T, SO SET CARRY
7446                            <1> S25:
7447 00002512 D0D2              <1>       RCL    DL, 1            ; MOVE THAT BIT INTO THE RESULT
7448 00002514 66C1E902          <1>       SHR    CX, 2            ; MOVE THE MASK TO THE RIGHT BY 2 BITS
7449 00002518 73F2              <1>       JNC    short S24        ; DO IT AGAIN IF MASK DIDN'T FALL OUT
7450 0000251A 885500            <1>       MOV    [eBP], DL        ; STORE RESULT IN SAVE AREA
7451 0000251D 45                <1>       INC    eBP              ; ADJUST POINTER
7452 0000251E C3                <1>       RETn                    ; ALL DONE
7453                            <1>
7454                            <1> ; 30/06/2016 - TRDOS 386 (TRDOS v2.0)
7455                            <1> ; VIDEO1.ASM - 24/03/1985 (IBM PC-AT BIOS source code)
7456                            <1> ;----------------------------------------
7457                            <1> ; V4_POSITION
7458                            <1> ;  THIS ROUTINE TAKES THE CURSOR POSITION CONTAINED IN
7459                            <1> ;  THE MEMORY LOCATION, AND CONVERTS IT INTO AN OFFSET
7460                            <1> ;  INTO THE REGEN BUFFER, ASSUMING ONE BYTE/CHAR.
7461                            <1> ;  FOR MEDIUM RESOLUTION GRAPHICS, THE NUMBER MUST
7462                            <1> ;  BE DOUBLED.
7463                            <1> ; ENTRY -- NO REGISTERS,MEMORY LOCATION @CURSOR_POSN IS USED
7464                            <1> ; EXIT--
7465                            <1> ;  AX CONTAINS OFFSET INTO REGEN BUFFER
7466                            <1> ;----------------------------------------
7467                            <1> S26:
7468 0000251F 0FB705[E64D0100]  <1>       movzx  eax, word [CURSOR_POSN]   ; GET CURRENT CURSOR
7469                            <1> GRAPH_POSN:
7470 00002526 53                <1>       PUSH   eBX              ; SAVE REGISTER
7471 00002527 0FB6D8            <1>       movzx  ebx, al                  ; SAVE A COPY OF CURRENT CURSOR
7472 0000252A A0[C45E0000]      <1>       MOV    AL, [CRT_COLS]           ; GET BYTES PER COLUMN
7473 0000252F F6E4              <1>       MUL    AH               ; MULTIPLY BY ROWS
7474 00002531 66C1E002          <1>       SHL    AX, 2            ; MULTIPLY * 4 SINCE 4 ROWS/BYTE
7475 00002535 01D8              <1>       ADD    eAX, eBX         ; DETERMINE OFFSET
7476 00002537 5B                <1>       POP    eBX              ; RECOVER POINTER
7477 00002538 C3                <1>       RETn                    ; ALL DONE
7478                            <1>
7479                            <1> ; 09/07/2016
7480                            <1> ; 01/07/2016 - TRDOS 386 (TRDOS v2.0)
7481                            <1> ; VIDEO1.ASM - 24/03/1985 (IBM PC-AT BIOS source code)
7482                            <1> ;----------------------------------------------
7483                            <1> ; SET_COLOR
7484                            <1> ;     THIS ROUTINE WILL ESTABLISH THE BACKGROUND COLOR, THE OVERSCAN COLOR,
7485                            <1> ;     AND THE FOREGROUND COLOR SET FOR MEDIUM RESOLUTION GRAPHICS
7486                            <1> ; INPUT
7487                            <1> ;     (BH) HAS COLOR ID
7488                            <1> ;           IF BH=0, THE BACKGROUND COLOR VALUE IS SET
7489                            <1> ;                 FROM THE LOW BITS OF BL (0-31)
7490                            <1> ;           IF BH=1, THE PALETTE SELECTION IS MADE
7491                            <1> ;                 BASED ON THE LOW BIT OF BL:
7492                            <1> ;                       0 = GREEN, RED, YELLOW FOR COLORS 1,2,3
7493                            <1> ;                       1 = BLUE, CYAN, MAGENTA FOR COLORS 1,2,3
7494                            <1> ;     (BL) HAS THE COLOR VALUE TO BE USED
7495                            <1> ; OUTPUT
7496                            <1> ;     THE COLOR SELECTION IS UPDATED
7497                            <1> ;----------------------------------------------
7498                            <1> SET_COLOR:
7499 00002539 803D[C25E0000]07  <1>         cmp     byte [CRT_MODE], 7      ; 09/07/2016
7500 00002540 0F870EF0FFFF      <1>       ja     VIDEO_RETURN       ; nothing to do for VGA modes
7501                            <1>
7502                            <1>       ;MOV   DX, [ADDR_6845]           ; I/O PORT FOR PALETTE
7503                            <1>       ;mov   dx, 3D4h
7504                            <1>       ;ADD   DX,5               ; OVERSCAN PORT
7505 00002546 66BAD903          <1>       mov    dx, 3D9h
7506 0000254A A0[C55E0000]      <1>       MOV    AL, [CRT_PALETTE]  ; GET THE CURRENT PALETTE VALUE
7507 0000254F 08FF              <1>       OR     BH, BH             ; IS THIS COLOR 0?
7508 00002551 7512              <1>       JNZ    short M20          ; OUTPUT COLOR 1
7509                            <1>
7510                            <1> ;-----      HANDLE COLOR 0 BY SETTING THE BACKGROUND COLOR
7511                            <1>
7512 00002553 24E0              <1>       AND    AL, 0E0H           ; TURN OFF LOW 5 BITS OF CURRENT
7513 00002555 80E31F            <1>       AND    BL, 01FH           ; TURN OFF HIGH 3 BITS OF INPUT VALUE
7514 00002558 08D8              <1>       OR     AL, BL             ; PUT VALUE INTO REGISTER
7515                            <1> M19:                          ; OUTPUT THE PALETTE
7516 0000255A EE                <1>       OUT    DX, AL             ; OUTPUT COLOR SELECTION TO 3D9 PORT
7517 0000255B A2[C55E0000]      <1>       MOV    [CRT_PALETTE], AL  ; SAVE THE COLOR VALUE
7518 00002560 E9EFEFFFFF        <1>       JMP    VIDEO_RETURN
7519                            <1>
7520                            <1> ;-----      HANDLE COLOR 1 BY SELECTING THE PALETTE TO BE USED
7521                            <1>
7522                            <1> M20:
7523 00002565 24DF              <1>       AND    AL, 0DFH           ; TURN OFF PALETTE SELECT BIT
7524 00002567 D0EB              <1>       SHR    BL, 1              ; TEST THE LOW ORDER BIT OF BL
7525 00002569 73EF              <1>       JNC    short M19          ; ALREADY DONE
7526 0000256B 0C20              <1>       OR     AL, 20H                 ; TURN ON PALETTE SELECT BIT
7527 0000256D EBEB              <1>       JMP    short M19          ; GO DO IT
7528                            <1>
7529                            <1> ; 09/07/2016
7530                            <1> ; 01/07/2016 - TRDOS 386 (TRDOS v2.0)
7531                            <1> ; VIDEO1.ASM - 24/03/1985 (IBM PC-AT BIOS source code)
7532                            <1> ;----------------------------------------
7533                            <1> ; READ DOT -- WRITE DOT
7534                            <1> ; THESE ROUTINES WILL WRITE A DOT, OR READ THE
7535                            <1> ;  DOT AT THE INDICATED LOCATION
7536                            <1> ; ENTRY --
7537                            <1> ;   DX = ROW (0-199)     (THE ACTUAL VALUE DEPENDS ON THE MODE)
7538                            <1> ;   CX = COLUMN ( 0-639) ( THE VALUES ARE NOT RANGE CHECKED )
7539                            <1> ;   AL = DOT VALUE TO WRITE (1,2 OR 4 BITS DEPENDING ON MODE,
7540                            <1> ;     REQUIRED FOR WRITE DOT ONLY, RIGHT JUSTIFIED)
7541                            <1> ;     BIT 7 OF AL = 1 INDICATES XOR THE VALUE INTO THE LOCATION
7542                            <1> ;   DS = DATA SEGMENT
7543                            <1> ;   ES = REGEN SEGMENT
7544                            <1> ;
7545                            <1> ; EXIT
7546                            <1> ;     AL = DOT VALUE READ, RIGHT JUSTIFIED, READ ONLY
```

```
7547                                    <1> ;-----------------------------------------------
7548                                    <1>
7549                                    <1> READ_DOT:
7550                                    <1>       ; 09/07/2016
7551 0000256F 8A25[C25E0000]           <1>       mov    ah,  [CRT_MODE]
7552 00002575 80FC07                   <1>       cmp    ah, 7 ; 6!?
7553 00002578 760A                     <1>       jna    short read_dot_cga
7554                                    <1>
7555 0000257A E8CB030000               <1>       call   vga_read_pixel
7556                                    <1>       ; al = pixel value
7557 0000257F E9D5EFFFFF               <1>       jmp    _video_return
7558                                    <1>
7559                                    <1> read_dot_cga:
7560                                    <1>       ;je    VIDEO_RETURN ; 7
7561 00002584 80FC04                   <1>       cmp    ah, 4 ; graphics ?
7562 00002587 0F82C7EFFFFF             <1>       jb     VIDEO_RETURN ; no, text mode, nothing to do
7563                                    <1>
7564 0000258D E855000000               <1>       CALL   R3                  ; DETERMINE BYTE POSITION OF DOT
7565 00002592 8A06                     <1>       MOV    AL, [eSI]           ; GET THE BYTE
7566 00002594 20E0                     <1>       AND    AL, AH              ; MASK OFF THE OTHER BITS IN THE BYTE
7567 00002596 D2E0                     <1>       SHL    AL, CL              ; LEFT JUSTIFY THE VALUE
7568 00002598 88F1                     <1>       MOV    CL, DH              ; GET NUMBER OF BITS IN RESULT
7569 0000259A D2C0                     <1>       ROL    AL, CL              ; RIGHT JUSTIFY THE RESULT
7570                                    <1>       ;JMP   VIDEO_RETURN        ; RETURN FROM VIDEO I/O
7571 0000259C 0FB6C0                   <1>       movzx  eax, al
7572 0000259F E9B5EFFFFF               <1>       jmp    _video_return
7573                                    <1>
7574                                    <1> ; 09/07/2016
7575                                    <1> ; 01/07/2016 - TRDOS 386 (TRDOS v2.0)
7576                                    <1> ; VIDEO1.ASM - 24/03/1985 (IBM PC-AT BIOS source code)
7577                                    <1>
7578                                    <1> WRITE_DOT:
7579                                    <1>       ; 09/07/2016
7580 000025A4 8A25[C25E0000]           <1>       mov    ah, [CRT_MODE]
7581 000025AA 80FC07                   <1>       cmp    ah, 7 ; 6!?
7582 000025AD 760A                     <1>       jna    short write_dot_cga
7583                                    <1>
7584 000025AF E805030000               <1>       call   vga_write_pixel
7585 000025B4 E99BEFFFFF               <1>       jmp    VIDEO_RETURN
7586                                    <1>
7587                                    <1> write_dot_cga:
7588                                    <1>       ;je    VIDEO_RETURN ; 7
7589 000025B9 80FC04                   <1>       cmp    ah, 4 ; graphics ?
7590 000025BC 0F8292EFFFFF             <1>       jb     VIDEO_RETURN ; no, text mode, nothing to do
7591                                    <1>
7592                                    <1>       ;PUSH  AX                  ; SAVE DOT VALUE
7593 000025C2 6650                     <1>       PUSH   AX                  ; TWICE
7594 000025C4 E81E000000               <1>       CALL   R3                  ; DETERMINE BYTE POSITION OF THE DOT
7595 000025C9 D2E8                     <1>       SHR    AL, CL              ; SHIFT TO SET UP THE BITS FOR OUTPUT
7596 000025CB 20E0                     <1>       AND    AL, AH              ; STRIP OFF THE OTHER BITS
7597 000025CD 8A0E                     <1>       MOV    CL, [eSI]           ; GET THE CURRENT BYTE
7598 000025CF 665B                     <1>       POP    BX                  ; RECOVER XOR FLAG
7599 000025D1 F6C380                   <1>       TEST   BL, 80H                   ; IS IT ON
7600 000025D4 750D                     <1>       JNZ    short R2            ; YES, XOR THE DOT
7601 000025D6 F6D4                     <1>       NOT    AH                  ; SET MASK TO REMOVE THE INDICATED BITS
7602 000025D8 20E1                     <1>       AND    CL, AH
7603 000025DA 08C8                     <1>       OR     AL, CL              ; OR IN THE NEW VALUE OF THOSE BITS
7604                                    <1> R1:                            ; FINISH_DOT
7605 000025DC 8806                     <1>       MOV    [eSI], AL           ; RESTORE THE BYTE IN MEMORY
7606                                    <1>       ;POP   AX
7607 000025DE E971EFFFFF               <1>       JMP    VIDEO_RETURN        ; RETURN FROM VIDEO I/O
7608                                    <1> R2:                            ; XOR_DOT
7609 000025E3 30C8                     <1>       XOR    AL, CL              ; EXCLUSIVE OR THE DOTS
7610 000025E5 EBF5                     <1>       JMP    short R1            ; FINISH UP THE WRITING
7611                                    <1>
7612                                    <1> ; 01/07/2016 - TRDOS 386 (TRDOS v2.0)
7613                                    <1> ; VIDEO1.ASM - 24/03/1985 (IBM PC-AT BIOS source code)
7614                                    <1>
7615                                    <1> ;-----------------------------------------------
7616                                    <1> ; THIS SUBROUTINE DETERMINES THE REGEN BYTE LOCATION OF THE
7617                                    <1> ; INDICATED ROW COLUMN VALUE IN GRAPHICS MODE.
7618                                    <1> ; ENTRY --
7619                                    <1> ;  DX = ROW VALUE (0-199)
7620                                    <1> ;  CX = COLUMN VALUE (0-639)
7621                                    <1> ; EXIT --
7622                                    <1> ;  SI = OFFSET INTO REGEN BUFFER FOR BYTE OF INTEREST
7623                                    <1> ;  AH = MASK TO STRIP OFF THE BITS OF INTEREST
7624                                    <1> ;  CL = BITS TO SHIFT TO RIGHT JUSTIFY THE MASK IN AH
7625                                    <1> ;  DH = # BITS IN RESULT
7626                                    <1> ;  BX = MODIFIED
7627                                    <1> ;-----------------------------------------------
7628                                    <1> R3:
7629                                    <1>
7630                                    <1> ;-----      DETERMINE 1ST BYTE IN INDICATED ROW BY MULTIPLYING ROW VALUE BY 40
7631                                    <1> ;-----       ( LOW BIT OF ROW DETERMINES EVEN/ODD, 80 BYTES/ROW )
7632                                    <1>
7633 000025E7 0FB7F0                   <1>       movzx  esi, ax                   ; WILL SAVE AL AND AH DURING OPERATION
7634 000025EA B028                     <1>       MOV    AL, 40
7635 000025EC F6E2                     <1>       MUL    DL                  ; AX= ADDRESS OF START OF INDICATED ROW
7636 000025EE A808                     <1>       TEST   AL, 08H             ; TEST FOR EVEN/ODD ROW CALCULATED
7637 000025F0 7404                     <1>       JZ     short R4            ; JUMP IF EVEN ROW
7638 000025F2 6605D81F                 <1>       ADD    AX, 2000H-40        ; OFFSET TO LOCATION OF ODD ROWS ADJUST
7639                                    <1> R4:                            ; EVEN_ROW
7640 000025F6 6696                     <1>       XCHG   SI, AX              ; MOVE POINTER TO (SI) AND RECOVER (AX)
7641 000025F8 81C600800B00             <1>       add    esi, 0B8000h
7642 000025FE 6689CA                   <1>       MOV    DX, CX              ; COLUMN VALUE TO DX
7643                                    <1>
7644                                    <1> ;-----      DETERMINE GRAPHICS MODE CURRENTLY IN EFFECT
7645                                    <1>
7646                                    <1> ; SET UP THE REGISTERS ACCORDING TO THE MODE
7647                                    <1> ; CH = MASK FOR LOW OF COLUMN ADDRESS ( 7/3 FOR HIGH/MED RES )
7648                                    <1> ; CL = # OF ADDRESS BITS IN COLUMN VALUE ( 3/2 FOR H/M )
```

```
                        <1> ; BL = MASK TO SELECT BITS FROM POINTED BYTE ( 80H/C0H FOR H/M )
                        <1> ; BH = NUMBER OF VALID BITS IN POINTED BYTE ( 1/2 FOR H/M )
                        <1>
00002601 66BBC002       <1>       MOV    BX, 2C0H
00002605 66B90203       <1>       MOV    CX, 302H            ; SET PARMS FOR MED RES
00002609 803D[C25E0000]06 <1>     CMP    byte [CRT_MODE], 6
00002610 7208           <1>       JC     short R5            ; HANDLE IF MED RES
00002612 66BB8001       <1>       MOV    BX, 180H
00002616 66B90307       <1>       MOV    CX, 703H            ; SET PARMS FOR HIGH RES
                        <1>
                        <1> ;-----      DETERMINE BIT OFFSET IN BYTE FROM COLUMN MASK
                        <1> R5:
0000261A 20D5           <1>       AND    CH, DL              ; ADDRESS OF PEL WITHIN BYTE TO CH
                        <1>
                        <1> ;-----      DETERMINE BYTE OFFSET FOR THIS LOCATION IN COLUMN
                        <1>
0000261C 66D3EA         <1>       SHR    DX, CL              ; SHIFT BY CORRECT AMOUNT
0000261F 6601D6         <1>       ADD    SI, DX              ; INCREMENT THE POINTER
00002622 88FE           <1>       MOV    DH, BH              ; GET THE # OF BITS IN RESULT TO DH
                        <1>
                        <1> ;-----      MULTIPLY BH (VALID BITS IN BYTE) BY CH (BIT OFFSET)
                        <1>
00002624 28C9           <1>       SUB    CL, CL              ; ZERO INTO STORAGE LOCATION
                        <1> R6:
00002626 D0C8           <1>       ROR    AL, 1               ; LEFT JUSTIFY VALUE IN AL (FOR WRITE)
00002628 00E9           <1>       ADD    CL, CH              ; ADD IN THE BIT OFFSET VALUE
0000262A FECF           <1>       DEC    BH                  ; LOOP CONTROL
0000262C 75F8           <1>       JNZ    short R6            ; ON EXIT, CL HAS COUNT TO RESTORE BITS
0000262E 88DC           <1>       MOV    AH, BL              ;  GET MASK TO AH
00002630 D2EC           <1>       SHR    AH, CL              ;  MOVE THE MASK TO CORRECT LOCATION
00002632 C3             <1>       RETn                       ;  RETURN WITH EVERYTHING SET UP
                        <1>
                        <1> load_dac_palette:
                        <1>       ; 29/07/2016
                        <1>       ; 23/07/2016
                        <1>       ; 03/07/2016 (TRDOS 386 = TRDOS v2.0)
                        <1>       ; (set_mode_vga)
                        <1>       ; derived from 'Plex86/Bochs VGABios' source code
                        <1>       ; vgabios-0.7a (2011)
                        <1>       ; by the LGPL VGABios developers Team (2001-2008)
                        <1>       ; 'vgabios.c', 'load_dac_palette'
                        <1>       ;
                        <1>       ; Oracle VirtualBox 5.0.24 VGABios Source Code
                        <1>       ; ('vgabios.c', 'vgatables.h', 'vgafonts.h', 'vgarom.asm')
                        <1>       ;
                        <1>       ; INPUT -> AH = DAC selection number (3, 2 or 1)
                        <1>       ; OUTPUT -> ECX = 0, AX = 0
                        <1>       ; (Modifed registers: EAX, ECX, EDX, ESI)
                        <1>       ;
00002633 66BAC803       <1>       mov    dx, 3C8h  ; VGAREG_DAC_WRITE_ADDRESS
00002637 28C0           <1>       sub    al, al ; 0
00002639 EE             <1>       out    dx, al ; 0 ; color index, always 0 at the beginning
0000263A 6642           <1>       inc    dx   ; 3C9h ; VGAREG_DAC_DATA
0000263C B900010000     <1>       mov    ecx, 256   ; always 256*3 values
                        <1>       ;push  esi
00002641 88E0           <1>       mov    al, ah
00002643 B43F           <1>       mov    ah, 3Fh      ; 3Fh except DAC selection number 3
00002645 3C02           <1>       cmp    al, 2
00002647 7414           <1>       je     short l_dac_p_2
00002649 7719           <1>       ja     short l_dac_p_3
0000264B 20C0           <1>       and    al, al
0000264D 7507           <1>       jnz    short l_dac_p_1
                        <1> l_dac_p_0:
0000264F BE[041D0100]   <1>       mov    esi, palette0
00002654 EB15           <1>       jmp    short l_dac_p_4
                        <1> l_dac_p_1:
00002656 BE[C41D0100]   <1>       mov    esi, palette1
0000265B EB0E           <1>       jmp    short l_dac_p_4
                        <1> l_dac_p_2:
0000265D BE[841E0100]   <1>       mov    esi, palette2
00002662 EB07           <1>       jmp    short l_dac_p_4
                        <1> l_dac_p_3:
00002664 B4FF           <1>       mov    ah, 0FFh ; dac registers
00002666 BE[441F0100]   <1>       mov    esi, palette3
                        <1> l_dac_p_4:
0000266B AC             <1>       lodsb
0000266C EE             <1>       out    dx, al  ; Red
0000266D AC             <1>       lodsb
0000266E EE             <1>       out    dx, al ; Green
0000266F AC             <1>       lodsb
00002670 EE             <1>       out    dx, al ; Blue
00002671 20E4           <1>       and    ah, ah
00002673 7405           <1>       jz     short l_dac_p_5
00002675 FECC           <1>       dec    ah
00002677 E2F2           <1>       loop   l_dac_p_4
                        <1>       ;pop   esi
00002679 C3             <1>       retn
                        <1> l_dac_p_5:
                        <1>       ; 29/07/2016
0000267A FEC9           <1>       dec    cl
0000267C 7407           <1>       jz     short l_dac_p_7
                        <1>       ;
0000267E 28C0           <1>       sub    al, al ; 0
                        <1> l_dac_p_6:
00002680 EE             <1>       out    dx, al ; outb(VGAREG_DAC_DATA,0);
00002681 EE             <1>       out    dx, al
00002682 EE             <1>       out    dx, al
00002683 E2FB           <1>       loop   l_dac_p_6
                        <1> l_dac_p_7:
                        <1>       ;pop   esi
00002685 C3             <1>       retn
                        <1>
```

```
                                <1> gray_scale_summing:
                                <1>       ; 03/07/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>       ; (set_mode_vga)
                                <1>       ; derived from 'Plex86/Bochs VGABios' source code
                                <1>       ; vgabios-0.7a (2011)
                                <1>       ; by the LGPL VGABios developers Team (2001-2008)
                                <1>       ; 'vgabios.c', 'biosfn_perform_gray_scale_summing'
                                <1>       ;
                                <1>       ; Oracle VirtualBox 5.0.24 VGABios Source Code
                                <1>       ; ('vgabios.c', 'vgatables.h', 'vgafonts.h', 'vgarom.asm')
                                <1>       ;
                                <1>
                                <1>       ; INPUT -> EBX = Start address (color index <= 255)
                                <1>       ;         ECX = Count (<= 256)
                                <1>       ; OUTPUT -> (E)CX = 0
                                <1>       ; (Modifed registers: EAX, ECX, EDX, EBX)
                                <1>
00002686 66BADA03               <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
0000268A EC                     <1>       in     al, dx
0000268B 30C0                   <1>       xor    al, al ; 0
0000268D 66BAC003               <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
00002691 EE                     <1>       out    dx, al ; clear bit 5
                                <1>                     ; (while loading palette registers)
                                <1>       ; set read address and switch to read mode
                                <1> g_s_s_1:
00002692 66BAC703               <1>       mov    dx, 3C7h ; VGAREG_DAC_READ_ADDRESS
00002696 88D8                   <1>       mov    al, bl
00002698 EE                     <1>       out    dx, al
                                <1>       ; get 6-bit wide RGB data values
                                <1>       ; intensity = (0.3*Red)+(0.59*Green)+(0.11*Blue)
                                <1>       ; i = ( ( 77*r + 151*g + 28*b ) + 0x80 ) >> 8;
00002699 66BAC903               <1>       mov    dx, 3C9h ; VGAREG_DAC_DATA
0000269D EC                     <1>       in     al, dx ; red
0000269E B44D                   <1>       mov    ah, 77 ; 0.3* Red
000026A0 F6E4                   <1>       mul    ah
000026A2 6650                   <1>       push   ax
000026A4 EC                     <1>       in     al, dx ; green
000026A5 B497                   <1>       mov    ah, 151  ; 0.59 * Green
000026A7 F6E4                   <1>       mul    ah
000026A9 6650                   <1>       push   ax
000026AB EC                     <1>       in     al, dx ; blue
000026AC B41C                   <1>       mov    ah, 28 ; 0.11 * Blue
000026AE F6E4                   <1>       mul    ah
000026B0 665A                   <1>       pop    dx
000026B2 6601D0                 <1>       add    ax, dx
000026B5 665A                   <1>       pop    dx
000026B7 6601D0                 <1>       add    ax, dx
000026BA 66058000               <1>       add    ax, 80h
000026BE B03F                   <1>       mov    al, 3Fh
000026C0 38C4                   <1>       cmp    ah, al
000026C2 7602                   <1>       jna    short g_s_s_2
000026C4 88C4                   <1>       mov    ah, al
                                <1> g_s_s_2:
000026C6 66BAC803               <1>       mov    dx, 3C8h  ; VGAREG_DAC_WRITE_ADDRESS
000026CA 88D8                   <1>       mov    al, bl ; color index
000026CC EE                     <1>       out    dx, al
000026CD 88E0                   <1>       mov    al, ah ; intensity
000026CF 6642                   <1>       inc    dx ; 3C9h ; VGAREG_DAC_DATA
000026D1 EE                     <1>       out    dx, al ; R (R=G=B)
000026D2 88E0                   <1>       mov    al, ah ; intensity
000026D4 EE                     <1>       out    dx, al ; G (R=G=B)
000026D5 88E0                   <1>       mov    al, ah ; intensity
000026D7 EE                     <1>       out    dx, al ; B (R=G=B)
000026D8 6649                   <1>       dec    cx
000026DA 7404                   <1>       jz     short g_s_s_3
000026DC FEC3                   <1>       inc    bl    ; next color index value
000026DE EBB2                   <1>       jmp    short g_s_s_1
                                <1> g_s_s_3:
000026E0 66BADA03               <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
000026E4 EC                     <1>       in     al, dx
000026E5 B020                   <1>       mov    al, 20h
000026E7 66BAC003               <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
000026EB EE                     <1>       out    dx, al ; 20h -> set bit 5
                                <1>                     ; (after loading palette regs)
000026EC C3                     <1>       retn
                                <1>
                                <1> vga_write_char_attr:
                                <1> vga_write_char_only:
                                <1>       ; 08/07/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>       ;
                                <1>       ; derived from 'Plex86/Bochs VGABios' source code
                                <1>       ; vgabios-0.7a (2011)
                                <1>       ; by the LGPL VGABios developers Team (2001-2008)
                                <1>       ; 'vgabios.c', 'biosfn_write_char_attr'
                                <1>       ; 'biosfn_write_char_only'
                                <1>
                                <1>       ; INPUT ->
                                <1>       ; [CRT_MODE] = current video mode (>7)
                                <1>       ; CX = Count of characters to write
                                <1>       ; AL = Character to write
                                <1>       ; BL = Color of character
                                <1>       ; OUTPUT ->
                                <1>       ; Regen buffer updated
                                <1>
000026ED 8A25[C25E0000]         <1>       mov    ah, [CRT_MODE]
000026F3 668B15[E64D0100]       <1>       mov    dx, [CURSOR_POSN] ; cursor pos for page 0
                                <1>
000026FA BE[DE5E0000]           <1>       mov    esi, vga_modes
000026FF 89F7                   <1>       mov    edi, esi
00002701 83C710                 <1>       add    edi, vga_mode_count
                                <1> vga_wca_0:
00002704 AC                     <1>       lodsb
```

```
7853 00002705 38E0                <1>       cmp    al, ah ; [CRT_MODE]
7854 00002707 7405                <1>       je     short vga_wca_2
7855 00002709 39FE                <1>       cmp    esi, edi
7856 0000270B 72F7                <1>       jb     short vga_wca_0
7857                              <1> vga_wca_1:
7858 0000270D C3                  <1>       retn   ; nothing to do
7859                              <1> vga_wca_2:
7860 0000270E 83C64F              <1>       add    esi, vga_memmodel - (vga_modes + 1)
7861                              <1>       ; [ESI] = VGA memory model number (LINEAR8, PLANAR4, PLANAR1)
7862                              <1>
7863                              <1>       ; biosfn_write_char_attr (car,page,attr,count)
7864                              <1>       ; AL = car, page = 0, BL = attr, CX = count
7865 00002711 803E04              <1>       cmp    byte [esi], PLANAR4
7866 00002714 741D                <1>       je     short vga_wca_planar
7867 00002716 803E03              <1>       cmp    byte [esi], PLANAR1
7868 00002719 7418                <1>       je     short vga_wca_planar
7869                              <1> vga_wca_linear8:
7870                              <1>       ; while((count-->0) && (xcurs<nbcols))
7871                              <1>       ; CX = count
7872 0000271B 6621C9              <1>       and    cx, cx
7873 0000271E 74ED                <1>       jz     short vga_wca_1
7874 00002720 3A15[C45E0000]      <1>       cmp    dl, [CRT_COLS]
7875 00002726 73E5                <1>       jnb    short vga_wca_1
7876                              <1>       ; write_gfx_char_lin(car,attr,xcurs,ycurs,nbcols);
7877                              <1>       ; AL = car, BL = attr, DL = xcurs, DH = ycurs,
7878                              <1>       ; [CRT_COLS] = nbcols
7879 00002728 E81E000000          <1>       call   write_gfx_char_lin
7880 0000272D 6649                <1>       dec    cx ; count
7881 0000272F FEC2                <1>       inc    dl ; xcurs
7882 00002731 EBE8                <1>       jmp    short vga_wca_linear8
7883                              <1> vga_wca_planar:
7884                              <1>       ; while((count-->0) && (xcurs<nbcols))
7885                              <1>       ; CX = count
7886 00002733 6621C9              <1>       and    cx, cx
7887 00002736 74D5                <1>       jz     short vga_wca_1
7888 00002738 3A15[C45E0000]      <1>       cmp    dl, [CRT_COLS]
7889 0000273E 73CD                <1>       jnb    short vga_wca_1
7890                              <1>       ; write_gfx_char_pl4(car,attr,xcurs,ycurs,nbcols,cheight);
7891                              <1>       ; AL = car, BL = attr, DL = xcurs, DH = ycurs,
7892                              <1>       ; [CRT_COLS] = nbcols, [CHAR_HEIGHT] = cheight
7893 00002740 E89D000000          <1>       call   write_gfx_char_pl4
7894 00002745 6649                <1>       dec    cx ; count
7895 00002747 FEC2                <1>       inc    dl ; xcurs
7896 00002749 EBE8                <1>       jmp    short vga_wca_planar
7897                              <1>
7898                              <1> write_gfx_char_lin:
7899                              <1>       ; 08/08/2016
7900                              <1>       ; 31/07/2016
7901                              <1>       ; 08/07/2016 (TRDOS 386 = TRDOS v2.0)
7902                              <1>       ;
7903                              <1>       ; derived from 'Plex86/Bochs VGABios' source code
7904                              <1>       ; vgabios-0.7a (2011)
7905                              <1>       ; by the LGPL VGABios developers Team (2001-2008)
7906                              <1>       ; 'vgabios.c', 'write_gfx_char_lin'
7907                              <1>
7908                              <1>       ; write_gfx_char_lin(car,attr,xcurs,ycurs,nbcols)
7909                              <1>       ; INPUT ->
7910                              <1>       ; AL = car, BL = attr, DL = xcurs, DH = ycurs,
7911                              <1>       ; [CRT_COLS] = nbcols
7912                              <1>       ; OUTPUT ->
7913                              <1>       ; Regen buffer updated
7914                              <1>
7915 0000274B 51                  <1>       push   ecx
7916 0000274C 53                  <1>       push   ebx
7917 0000274D 52                  <1>       push   edx
7918 0000274E 50                  <1>       push   eax
7919                              <1>       ; addr=xcurs*8+ycurs*nbcols*64;
7920                              <1>       ; 08/08/2016
7921 0000274F 0FB6F0              <1>       movzx  esi, al ; car
7922 00002752 0FB6C6              <1>       movzx  eax, dh ; ycurs
7923 00002755 8A25[C45E0000]      <1>       mov    ah, [CRT_COLS] ; nbcols
7924 0000275B F6E4                <1>       mul    ah
7925                              <1>       ;shl   ax, 6 ; * 64
7926 0000275D 66C1E003            <1>       shl    ax, 3 ; * 8
7927                              <1>       ;sub   dh, dh
7928                              <1>       ;shl   dx, 3 ; xcurs * 8
7929                              <1>       ;movzx edi, dx
7930 00002761 0FB6FA              <1>       movzx  edi, dl
7931 00002764 66C1E703            <1>       shl    di, 3 ; xcurs * 8
7932 00002768 30F6                <1>       xor    dh, dh
7933 0000276A 8A15[C65E0000]      <1>       mov    dl, [CHAR_HEIGHT]
7934 00002770 66F7E2              <1>       mul    dx
7935                              <1>       ; eax = ycurs*nbcols*8*[CHAR_HEIGHT]
7936 00002773 01C7                <1>       add    edi, eax ; addr
7937 00002775 81C700000A00        <1>       add    edi, 0A0000h
7938                              <1>       ;shl   si, 3 ; car * 8
7939 0000277B 30E4                <1>       xor    ah, ah
7940 0000277D A0[C65E0000]        <1>       mov    al, [CHAR_HEIGHT]
7941 00002782 66F7E6              <1>       mul    si
7942 00002785 6689C6              <1>       mov    si, ax
7943                              <1>       ;; esi = src = car * 8
7944                              <1>       ; esi = src = car * [CHAR_HEIGHT]
7945                              <1>       ; i = 0
7946                              <1>       ;add   esi, vgafont8 ; fdata [src+i]
7947                              <1>       ; 08/08/2016
7948 00002788 A1[765B0100]        <1>       mov    eax, [VGA_INT43H]
7949 0000278D 3D[44380100]        <1>       cmp    eax, vgafont16
7950 00002792 740F                <1>         je      short wgfxl_0
7951 00002794 3D[442A0100]        <1>       cmp    eax, vgafont14
7952 00002799 7408                <1>       je     short wgfxl_0
7953 0000279B 81C6[44220100]      <1>       add    esi, vgafont8
7954 000027A1 EB02                <1>       jmp    short wgfxl_1
```

```
                                <1> wgfxl_0:
000027A3 01C6                   <1>         add    esi, eax
                                <1> wgfxl_1:
000027A5 28FF                   <1>         sub    bh, bh ; i = 0
                                <1> wgfxl_2:
                                <1>         ; for(i=0;i<8;i++)
000027A7 57                     <1>         push   edi ; addr
000027A8 0FB605[C45E0000]       <1>         movzx  eax, byte [CRT_COLS] ; nbcols
000027AF F6E7                   <1>         mul    bh ; nbcols*i
000027B1 66C1E003               <1>         shl    ax, 3 ; i*nbcols*8
                                <1>         ; dest=addr+i*nbcols*8;
000027B5 01C7                   <1>         add    edi, eax ; dest + j ; j = 0
000027B7 B180                   <1>         mov    cl, 80h ; mask = 0x80;
                                <1>         ; esi = fdata + src + i
                                <1>         ; for(j=0;j<8;j++)
000027B9 29D2                   <1>         sub    edx, edx ; j = 0
                                <1> wgfxl_3:
000027BB 8A06                   <1>         mov    al, [esi] ; al = fdata[src+i]
000027BD 20C8                   <1>         and    al, cl ; if (fdata[src+i] & mask)
000027BF 7402                   <1>         jz     short wgfxl_4  ; data = 0, zf = 1
000027C1 88D8                   <1>         mov    al, bl ; data = attr;
                                <1> wgfxl_4:
                                <1>         ; write_byte(0xa000,dest+j,data);
000027C3 AA                     <1>         stosb  ; dest + j (+ 0A0000h)
                                <1>         ;inc   dl ; j++
                                <1>         ;cmp   dl, 8
000027C4 80FA07                 <1>         cmp    dl, 7
000027C7 720E                   <1>         jb     short wgfxl_5
000027C9 5F                     <1>         pop    edi
                                <1>         ; 08/08/2016
                                <1>         ;cmp   bh, 7
                                <1>         ;jnb   short wgfxl_6
000027CA FEC7                   <1>         inc    bh ; i++
000027CC 3A3D[C65E0000]         <1>         cmp    bh, [CHAR_HEIGHT]
000027D2 7309                   <1>         jnb    short wgfxl_6
000027D4 46                     <1>         inc    esi
000027D5 EBD0                   <1>         jmp    short wgfxl_2
                                <1> wgfxl_5:
000027D7 D0E9                   <1>         shr    cl, 1 ; mask >>= 1;
000027D9 FEC2                   <1>         inc    dl ; j++
000027DB EBDE                   <1>           jmp     short wgfxl_3
                                <1> wgfxl_6:
000027DD 58                     <1>         pop    eax
000027DE 5A                     <1>         pop    edx
000027DF 5B                     <1>         pop    ebx
000027E0 59                     <1>         pop    ecx
000027E1 C3                     <1>         retn
                                <1>
                                <1> write_gfx_char_pl4:
                                <1>         ; 08/08/2016
                                <1>         ; 08/07/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>         ;
                                <1>         ; derived from 'Plex86/Bochs VGABios' source code
                                <1>         ; vgabios-0.7a (2011)
                                <1>         ; by the LGPL VGABios developers Team (2001-2008)
                                <1>         ; 'vgabios.c', 'write_gfx_char_pl4'
                                <1>
                                <1>         ; write_gfx_char_pl4(car,attr,xcurs,ycurs,nbcols,cheight)
                                <1>         ; INPUT ->
                                <1>         ; AL = car, BL = attr, DL = xcurs, DH = ycurs,
                                <1>         ; [CRT_COLS] = nbcols, [CHAR_HEIGHT] = cheight
                                <1>         ; OUTPUT ->
                                <1>         ; Regen buffer updated
                                <1>
000027E2 51                     <1>         push   ecx
000027E3 53                     <1>         push   ebx
000027E4 52                     <1>         push   edx
000027E5 50                     <1>         push   eax
                                <1> wgfxpl_f0:
                                <1>         ; switch(cheight)
000027E6 8A25[C65E0000]         <1>         mov    ah, [CHAR_HEIGHT]
000027EC 80FC10                 <1>         cmp    ah, 16 ; case 16:
000027EF 7507                   <1>         jne    short wgfxpl_f1
                                <1>         ; fdata = &vgafont16;
000027F1 BE[44380100]           <1>         mov    esi, vgafont16
000027F6 EB13                   <1>         jmp    short wgfxpl_f3
                                <1> wgfxpl_f1:
000027F8 80FC0E                 <1>         cmp    ah, 14 ; case 14:
000027FB 7507                   <1>         jne    short wgfxpl_f2
000027FD BE[442A0100]           <1>         mov    esi, vgafont14
00002802 EB07                   <1>         jmp    short wgfxpl_f3
                                <1> wgfxpl_f2:
                                <1>         ; default:
                                <1>         ;  fdata = &vgafont8;
00002804 BE[44220100]           <1>         mov    esi, vgafont8
00002809 B408                   <1>         mov    ah, 8
                                <1> wgfxpl_f3:
                                <1>         ; al = car
0000280B F6E4                   <1>         mul    ah ; ah = cheight
0000280D 25FFFF0000             <1>         and    eax, 0FFFFh ; car * cheight
                                <1>         ; src = car * cheight;
00002812 01C6                   <1>         add    esi, eax ; esi = fdata[src+i]
                                <1>         ; addr=xcurs*8+ycurs*nbcols*64;
00002814 88F0                   <1>         mov    al, dh ; ycurs
00002816 8A25[C45E0000]         <1>         mov    ah, [CRT_COLS] ; nbcols
0000281C F6E4                   <1>         mul    ah
                                <1>         ; 08/08/2016
                                <1>         ;shl   ax, 6 ; * 64
0000281E 66C1E003               <1>         shl    ax, 3 ; * 8
                                <1>         ;sub   dh, dh ; 0
                                <1>         ;shl   dx, 3 ; xcurs * 8
                                <1>         ;movzx edi, dx
```

```
8057 00002822 0FB6FA              <1>       movzx  edi, dl
8058 00002825 66C1E703            <1>       shl    di, 3 ; xcurs * 8
8059 00002829 30F6                <1>       xor    dh, dh
8060 0000282B 8A15[C65E0000]      <1>       mov    dl, [CHAR_HEIGHT]
8061 00002831 66F7E2              <1>       mul    dx
8062                              <1>       ; eax = ycurs*nbcols*8*[CHAR_HEIGHT]
8063 00002834 01C7                <1>       add    edi, eax ; addr
8064 00002836 81C700000A00        <1>       add    edi, 0A0000h
8065                              <1>       ;
8066                              <1>       ; outw(VGAREG_SEQU_ADDRESS, 0x0f02);
8067                              <1>       ; outw(VGAREG_GRDC_ADDRESS, 0x0205);
8068 0000283C 66BAC403            <1>       mov    dx, 3C4h ; VGAREG_SEQU_ADDRESS
8069 00002840 66B8020F            <1>       mov    ax, 0F02h
8070 00002844 66EF                <1>       out    dx, ax
8071 00002846 66BACE03            <1>       mov    dx, 3CEh ; VGAREG_GRDC_ADDRESS
8072 0000284A 66B80502            <1>       mov    ax, 0205h
8073 0000284E 66EF                <1>       out    dx, ax
8074                              <1>       ;
8075 00002850 66BACE03            <1>       mov    dx, 3CEh ; VGAREG_GRDC_ADDRESS
8076 00002854 F6C380              <1>       test   bl, 80h ; if(attr&0x80)
8077 00002857 7406                <1>       jz     short wgfxpl_f4 ; else
8078                              <1>       ; outw(VGAREG_GRDC_ADDRESS, 0x1803);
8079 00002859 66B80318            <1>       mov    ax, 1803h
8080 0000285D EB04                <1>       jmp    short wgfxpl_f5
8081                              <1> wgfxpl_f4:
8082                              <1>       ; outw(VGAREG_GRDC_ADDRESS, 0x0003);
8083 0000285F 66B80300            <1>       mov    ax, 0003h
8084                              <1> wgfxpl_f5:
8085 00002863 66EF                <1>       out    dx, ax
8086                              <1>       ;
8087 00002865 28FF                <1>       sub    bh, bh ; i = 0
8088                              <1> wgfxpl_0:
8089                              <1>       ; for(i=0;i<cheight;i++)
8090 00002867 57                  <1>       push   edi ; addr
8091 00002868 0FB605[C45E0000]    <1>       movzx  eax, byte [CRT_COLS] ; nbcols
8092 0000286F F6E7                <1>       mul    bh ; nbcols*i
8093                              <1>       ; dest=addr+i*nbcols
8094 00002871 01C7                <1>       add    edi, eax ; dest
8095 00002873 B580                <1>       mov    ch, 80h ; mask = 0x80;
8096                              <1>       ; for(j=0;j<8;j++)
8097 00002875 28C9                <1>       sub    cl, cl ; j = 0
8098                              <1> wgfxpl_1:
8099 00002877 D2ED                <1>       shr    ch, cl ; mask=0x80>>j;
8100                              <1>       ;
8101                              <1>       ; outw(VGAREG_GRDC_ADDRESS, (mask << 8) | 0x08);
8102                              <1>       ; read_byte(0xa000,dest);
8103                              <1>       ;mov   dx, 3CEh ; VGAREG_GRDC_ADDRESS
8104 00002879 88EC                <1>       mov    ah, ch
8105 0000287B B008                <1>       mov    al, 8
8106 0000287D 66EF                <1>       out    dx, ax
8107 0000287F 8A07                <1>       mov    al, [edi] ; ? (io delay?)
8108                              <1>       ;
8109 00002881 28C0                <1>       sub    al, al ; attr = 0
8110                              <1>       ; if (fdata[src+i] & mask)
8111 00002883 842E                <1>       test   byte [esi], ch
8112 00002885 7404                <1>       jz     short wgfxpl_2  ; zf = 1
8113                              <1>       ; write_byte(0xa000,dest,attr&0x0f);
8114 00002887 88D8                <1>       mov    al, bl ; attr;
8115 00002889 240F                <1>       and    al, 0Fh      ; attr&0x0f
8116                              <1> wgfxpl_2:
8117                              <1>       ; write_byte(0xa000,dest,0x00);
8118 0000288B 8807                <1>       mov    [edi], al ; dest (+ 0A0000h)
8119 0000288D FEC1                <1>       inc    cl ; j++
8120 0000288F 80F908              <1>       cmp    cl, 8
8121 00002892 72E3                <1>       jb     short wgfxpl_1
8122 00002894 5F                  <1>       pop    edi
8123                              <1>       ; 08/08/2016
8124                              <1>       ;cmp   bh, 7
8125                              <1>       ;jnb   short wgfxpl_3
8126 00002895 FEC7                <1>       inc    bh ; i++
8127 00002897 3A3D[C65E0000]      <1>       cmp    bh, [CHAR_HEIGHT]
8128 0000289D 7303                <1>       jnb    short wgfxpl_3
8129 0000289F 46                  <1>       inc    esi
8130 000028A0 EBC5                <1>       jmp    short wgfxpl_0
8131                              <1> wgfxpl_3:
8132                              <1>       ;mov   dx, 3CEh ; VGAREG_GRDC_ADDRESS
8133 000028A2 66B808FF            <1>       mov    ax, 0FF08h
8134 000028A6 66EF                <1>       out    dx, ax
8135 000028A8 66B80500            <1>       mov    ax, 0005h
8136 000028AC 66EF                <1>       out    dx, ax
8137 000028AE 66B80300            <1>       mov    ax, 0003h
8138 000028B2 66EF                <1>       out    dx, ax
8139                              <1>       ;
8140 000028B4 58                  <1>       pop    eax
8141 000028B5 5A                  <1>       pop    edx
8142 000028B6 5B                  <1>       pop    ebx
8143 000028B7 59                  <1>       pop    ecx
8144 000028B8 C3                  <1>       retn
8145                              <1>
8146                              <1> vga_write_pixel:
8147                              <1>       ; 09/07/2016 (TRDOS 386 = TRDOS v2.0)
8148                              <1>       ;
8149                              <1>       ; derived from 'Plex86/Bochs VGABios' source code
8150                              <1>       ; vgabios-0.7a (2011)
8151                              <1>       ; by the LGPL VGABios developers Team (2001-2008)
8152                              <1>       ; 'vgabios.c', 'biosfn_write_pixel'
8153                              <1>
8154                              <1>       ; INPUT ->
8155                              <1>       ;     DX = row (0-239)
8156                              <1>       ;     CX = column (0-799)
8157                              <1>       ;     AL = pixel value
8158                              <1>       ;     (AH = [CRT_MODE])
```

```
8159                                <1>       ; OUTPUT ->
8160                                <1>       ;      none
8161                                <1>
8162 000028B9 88C3                 <1>       mov    bl, al ; pixel value
8163                                <1>       ;mov   ah, [CRT_MODE]
8164 000028BB BE[DE5E0000]         <1>       mov    esi, vga_modes
8165 000028C0 89F7                 <1>       mov    edi, esi
8166 000028C2 83C710               <1>       add    edi, vga_mode_count
8167                                <1> vga_wp_0:
8168 000028C5 AC                   <1>       lodsb
8169 000028C6 38E0                 <1>       cmp    al, ah ; [CRT_MODE]
8170 000028C8 7405                 <1>       je     short vga_wp_1
8171 000028CA 39FE                 <1>       cmp    esi, edi
8172 000028CC 72F7                 <1>       jb     short vga_wp_0
8173 000028CE C3                   <1>       retn   ; nothing to do
8174                                <1> vga_wp_1:
8175 000028CF 83C64F               <1>       add    esi, vga_memmodel - (vga_modes + 1)
8176                                <1>       ; [ESI] = VGA memory model number (LINEAR8, PLANAR4, PLANAR1)
8177 000028D2 BF00000A00           <1>       mov    edi, 0A0000h
8178                                <1>       ;
8179 000028D7 803E04               <1>       cmp    byte [esi], PLANAR4
8180 000028DA 741D                 <1>       je     short vga_wp_planar
8181 000028DC 803E03               <1>       cmp    byte [esi], PLANAR1
8182 000028DF 7418                 <1>       je     short vga_wp_planar
8183                                <1> vga_wp_linear8:
8184                                <1>       ; addr=CX+DX*(read_word(BIOSMEM_SEG,BIOSMEM_NB_COLS)*8);
8185 000028E1 0FB605[C45E0000]     <1>       movzx  eax, byte [CRT_COLS] ; = [VGA_COLS] ; nbcols
8186 000028E8 66C1E003             <1>       shl    ax, 3 ; * 8
8187 000028EC 66F7E2               <1>       mul    dx
8188 000028EF 50                   <1>       push   eax
8189                                <1>       ;mov   edi, 0A0000h
8190 000028F0 6601CF               <1>       add    di, cx
8191 000028F3 58                   <1>       pop    eax
8192 000028F4 01C7                 <1>       add    edi, eax ; addr
8193                                <1>       ; write_byte(0xa000,addr,AL);
8194 000028F6 881F                 <1>       mov    [edi], bl
8195 000028F8 C3                   <1>       retn
8196                                <1> vga_wp_planar:
8197                                <1>       ; addr = CX/8+DX*read_word(BIOSMEM_SEG,BIOSMEM_NB_COLS);
8198 000028F9 0FB7C1               <1>       movzx  eax, cx
8199 000028FC 66C1E803             <1>       shr    ax, 3 ; CX/8
8200 00002900 50                   <1>       push   eax
8201 00002901 28E4                 <1>       sub    ah, ah ; 0
8202 00002903 A0[C45E0000]         <1>       mov    al, [CRT_COLS] ; = [VGA_COLS] ; nbcols
8203 00002908 66F7E2               <1>       mul    dx
8204                                <1>       ;mov   edi, 0A0000h
8205 0000290B 6601C7               <1>         add     di, ax
8206 0000290E 58                   <1>       pop    eax
8207 0000290F 01C7                 <1>       add    edi, eax ; addr
8208 00002911 80E107               <1>       and    cl, 7
8209 00002914 B580                 <1>       mov    ch, 80h ; mask
8210 00002916 D2ED                 <1>       shr    ch, cl       ; mask = 0x80 >> (CX & 0x07);
8211                                <1>
8212                                <1>       ; outw(VGAREG_GRDC_ADDRESS, (mask << 8) | 0x08);
8213 00002918 66BACE03             <1>       mov    dx, 3CEh ; VGAREG_GRDC_ADDRESS
8214 0000291C 88EC                 <1>       mov    ah, ch
8215 0000291E B008                 <1>       mov    al, 8
8216 00002920 66EF                 <1>       out    dx, ax
8217                                <1>       ; outw(VGAREG_GRDC_ADDRESS, 0x0205);
8218 00002922 66B80502             <1>       mov    ax, 0205h
8219 00002926 66EF                 <1>       out    dx, ax
8220                                <1>       ; data = read_byte(0xa000,addr);
8221 00002928 8A07                 <1>       mov    al, [edi] ; (delay?)
8222                                <1>       ; if (AL & 0x80)
8223                                <1>       ; {
8224                                <1>       ;  outw(VGAREG_GRDC_ADDRESS, 0x1803);
8225                                <1>       ; }
8226 0000292A F6C380               <1>       test   bl, 80h
8227 0000292D 7406                 <1>       jz     short vga_wp_2
8228 0000292F 66B80318             <1>       mov    ax, 1803h
8229 00002933 66EF                 <1>       out    dx, ax
8230                                <1> vga_wp_2:
8231                                <1>       ; write_byte(0xa000,addr,AL);
8232 00002935 881F                 <1>       mov    [edi], bl
8233                                <1>       ;
8234                                <1>       ;mov   dx, 3CEh ; VGAREG_GRDC_ADDRESS
8235 00002937 66B808FF             <1>       mov    ax, 0FF08h
8236 0000293B 66EF                 <1>       out    dx, ax
8237 0000293D 66B80500             <1>       mov    ax, 0005h
8238 00002941 66EF                 <1>       out    dx, ax
8239 00002943 66B80300             <1>       mov    ax, 0003h
8240 00002947 66EF                 <1>       out    dx, ax
8241                                <1>       ;
8242 00002949 C3                   <1>       retn
8243                                <1>
8244                                <1> vga_read_pixel:
8245                                <1>       ; 09/07/2016 (TRDOS 386 = TRDOS v2.0)
8246                                <1>       ;
8247                                <1>       ; derived from 'Plex86/Bochs VGABios' source code
8248                                <1>       ; vgabios-0.7a (2011)
8249                                <1>       ; by the LGPL VGABios developers Team (2001-2008)
8250                                <1>       ; 'vgabios.c', 'biosfn_read_pixel'
8251                                <1>
8252                                <1>       ; INPUT ->
8253                                <1>       ;      DX = row (0-239)
8254                                <1>       ;      CX = column (0-799)
8255                                <1>       ;      (AH = [CRT_MODE])
8256                                <1>       ; OUTPUT ->
8257                                <1>       ;      AL = pixel value
8258                                <1>
8259                                <1>       ;mov   ah, [CRT_MODE]
8260 0000294A BE[DE5E0000]         <1>       mov    esi, vga_modes
```

```
8261 0000294F 89F7              <1>       mov    edi, esi
8262 00002951 83C710            <1>       add    edi, vga_mode_count
8263                            <1> vga_rp_0:
8264 00002954 AC                <1>       lodsb
8265 00002955 38E0              <1>       cmp    al, ah ; [CRT_MODE]
8266 00002957 7405              <1>       je     short vga_rp_1
8267 00002959 39FE              <1>       cmp    esi, edi
8268 0000295B 72F7              <1>       jb     short vga_rp_0
8269 0000295D C3                <1>       retn   ; nothing to do
8270                            <1> vga_rp_1:
8271 0000295E 83C64F            <1>       add    esi, vga_memmodel - (vga_modes + 1)
8272                            <1>       ; [ESI] = VGA memory model number (LINEAR8, PLANAR4, PLANAR1)
8273 00002961 BF00000A00        <1>       mov    edi, 0A0000h
8274                            <1>       ;
8275 00002966 803E04            <1>       cmp    byte [esi], PLANAR4
8276 00002969 741D              <1>       je     short vga_rp_planar
8277 0000296B 803E03            <1>       cmp    byte [esi], PLANAR1
8278 0000296E 7418              <1>       je     short vga_rp_planar
8279                            <1> vga_rp_linear8:
8280                            <1>       ; addr=CX+DX*(read_word(BIOSMEM_SEG,BIOSMEM_NB_COLS)*8);
8281 00002970 0FB605[C45E0000]  <1>       movzx  eax, byte [CRT_COLS] ; = [VGA_COLS] ; nbcols
8282 00002977 66C1E003          <1>       shl    ax, 3 ; * 8
8283 0000297B 66F7E2            <1>       mul    dx
8284 0000297E 50                <1>       push   eax
8285                            <1>       ;mov   edi, 0A0000h
8286 0000297F 6601CF            <1>       add    di, cx
8287 00002982 58                <1>       pop    eax
8288 00002983 01C7              <1>       add    edi, eax ; addr
8289                            <1>       ; attr=read_byte(0xa000,addr);
8290 00002985 8A07              <1>       mov    al, [edi] ; pixel value
8291 00002987 C3                <1>       retn
8292                            <1> vga_rp_planar:
8293                            <1>       ; addr = CX/8+DX*read_word(BIOSMEM_SEG,BIOSMEM_NB_COLS);
8294 00002988 0FB7C1            <1>       movzx  eax, cx
8295 0000298B 66C1E803          <1>       shr    ax, 3 ; CX/8
8296 0000298F 50                <1>       push   eax
8297 00002990 28E4              <1>       sub    ah, ah ; 0
8298 00002992 A0[C45E0000]      <1>       mov    al, [CRT_COLS] ; = [VGA_COLS] ; nbcols
8299 00002997 66F7E2            <1>       mul    dx
8300                            <1>       ;mov   edi, 0A0000h
8301 0000299A 6601C7            <1>         add     di, ax
8302 0000299D 58                <1>       pop    eax
8303 0000299E 01C7              <1>       add    edi, eax ; addr
8304 000029A0 80E107            <1>       and    cl, 7
8305 000029A3 B580              <1>       mov    ch, 80h ; mask
8306 000029A5 D2ED              <1>       shr    ch, cl       ; mask = 0x80 >> (CX & 0x07);
8307                            <1>       ; attr = 0x00;
8308 000029A7 30DB              <1>       xor    bl, bl ; attr = bl = 0,
8309 000029A9 30C9              <1>       xor    cl, cl ; i = cl = 0
8310                            <1>       ; for(i=0;i<4;i++)
8311                            <1>             ; {
8312                            <1>             ;  outw(VGAREG_GRDC_ADDRESS, (i << 8) | 0x04);
8313                            <1>             ;  data = read_byte(0xa000,addr) & mask;
8314                            <1>             ;  if (data > 0) attr |= (0x01 << i);
8315                            <1>             ; }
8316                            <1> vga_rp_2:
8317 000029AB 88CC              <1>       mov    ah, cl ; i << 8
8318 000029AD B004              <1>       mov    al, 4  ; | 0x04
8319 000029AF 66BACE03          <1>       mov    dx, 3CEh ; VGAREG_GRDC_ADDRESS
8320 000029B3 66EF              <1>       out    dx, ax
8321                            <1>       ; data = read_byte(0xa000,addr) & mask;
8322 000029B5 8A07              <1>       mov    al, [edi]
8323 000029B7 20E8              <1>       and    al, ch ; & mask
8324                            <1>       ; if (data > 0) attr |= (0x01 << i);
8325 000029B9 08C0              <1>       or     al, al
8326 000029BB 7408              <1>       jz     short vga_rp_3 ; al = 0
8327 000029BD B701              <1>       mov    bh, 1
8328 000029BF D2E7              <1>       shl    bh, cl ; (0x01 << i)
8329 000029C1 08FB              <1>       or     bl, bh ; attr |= (0x01 << i)
8330 000029C3 88D8              <1>       mov    al, bl ; pixel value
8331                            <1> vga_rp_3:
8332 000029C5 C3                <1>       retn
8333                            <1>
8334                            <1> vga_beeper:
8335                            <1>       ; 04/08/2016  (TRDOS 386 = TRDOS v2.0)
8336 000029C6 FB                <1>       sti
8337                            <1>       ;mov   bh, [ACTIVE_PAGE]
8338 000029C7 E9CFF3FFFF        <1>         jmp     beeper_gfx
8339                            <1>
8340                            <1> vga_write_teletype:
8341                            <1>       ; 06/08/2016
8342                            <1>       ; 04/08/2016
8343                            <1>       ; 01/08/2016
8344                            <1>       ; 31/07/2016
8345                            <1>       ; 09/07/2016 (TRDOS 386 = TRDOS v2.0)
8346                            <1>       ;
8347                            <1>       ; derived from 'Plex86/Bochs VGABios' source code
8348                            <1>       ; vgabios-0.7a (2011)
8349                            <1>       ; by the LGPL VGABios developers Team (2001-2008)
8350                            <1>       ; 'vgabios.c', 'biosfn_write_teletype'
8351                            <1>       ; 'biosfn_write_char_only'
8352                            <1>
8353                            <1>       ; INPUT ->
8354                            <1>       ; [CRT_MODE] = current video mode (>7)
8355                            <1>       ; AL = Character to write
8356                            <1>       ; BL = Color of character
8357                            <1>       ; OUTPUT ->
8358                            <1>       ; Regen buffer updated
8359                            <1>
8360                            <1>       ; biosfn_write_teletype (car, page, attr, flag)
8361                            <1>       ; car = character (AL)
8362                            <1>       ; page = 0
```

```
8363                                <1>       ; attr = color (BL)
8364                                <1>       ; 'flag' not used
8365                                <1>
8366 000029CC 8A25[C25E0000]       <1>       mov    ah, [CRT_MODE]
8367 000029D2 88C7                 <1>       mov    bh, al ; character
8368 000029D4 668B15[E64D0100]     <1>       mov    dx, [CURSOR_POSN] ; cursor pos for page 0
8369                                <1>
8370 000029DB BE[E65E0000]         <1>       mov    esi, vga_g_modes
8371 000029E0 89F7                 <1>       mov    edi, esi
8372 000029E2 83C708               <1>       add    edi, vga_g_mode_count
8373                                <1> vga_wtty_0:
8374 000029E5 AC                   <1>       lodsb
8375 000029E6 38E0                 <1>       cmp    al, ah ; [CRT_MODE]
8376 000029E8 7405                 <1>       je     short vga_wtty_2
8377 000029EA 39FE                 <1>       cmp    esi, edi
8378 000029EC 72F7                 <1>       jb     short vga_wtty_0
8379                                <1> vga_wtty_1:
8380 000029EE C3                   <1>       retn   ; nothing to do
8381                                <1> vga_wtty_2:
8382 000029EF 80FF07               <1>       cmp    bh, 07h ; bell (beep)
8383 000029F2 74D2                 <1>       je     short vga_beeper  ; u11
8384 000029F4 80FF08               <1>       cmp    bh, 08h ; backspace
8385 000029F7 7508                 <1>       jne    short vga_wtty_3
8386                                <1>       ; if(xcurs>0)xcurs--;
8387 000029F9 08D2                 <1>       or     dl, dl ; xcurs (column)
8388 000029FB 74F1                 <1>       jz     short vga_wtty_1
8389 000029FD FECA                 <1>       dec    dl ; xcurs--;
8390 000029FF EB59                 <1>         jmp     short vga_wtty_12
8391                                <1> vga_wtty_3:
8392 00002A01 80FF0D               <1>       cmp    bh, 0Dh ; carriage return (\r)
8393 00002A04 7504                 <1>       jne    short vga_wtty_4
8394                                <1>       ; xcurs=0;
8395 00002A06 28D2                 <1>       sub    dl, dl ; 0
8396 00002A08 EB50                 <1>         jmp     short vga_wtty_12
8397                                <1> vga_wtty_4:
8398 00002A0A 80FF0A               <1>       cmp    bh, 0Ah ; new line (\n)
8399 00002A0D 7504                 <1>       jne    short vga_wtty_5
8400                                <1>       ; ycurs++;
8401 00002A0F FEC6                 <1>       inc    dh ; next row
8402 00002A11 EB62                 <1>         jmp     short vga_wtty_11
8403                                <1> vga_wtty_5:
8404 00002A13 80FF09               <1>       cmp    bh, 09h ; tab stop
8405 00002A16 7527                 <1>       jne    short vga_wtty_8
8406 00002A18 88D0                 <1>       mov    al, dl
8407 00002A1A 6698                 <1>       cbw
8408 00002A1C B108                 <1>       mov    cl, 8
8409 00002A1E F6F1                 <1>       div    cl
8410 00002A20 28E1                 <1>       sub    cl, ah
8411                                <1>       ;
8412 00002A22 B720                 <1>       mov    bh, 20h ; space
8413                                <1> vga_wtty_6: ; tab stop loop
8414 00002A24 6651                 <1>       push   cx
8415 00002A26 6653                 <1>       push   bx
8416 00002A28 E812000000           <1>       call   vga_wtty_8
8417 00002A2D 665B                 <1>       pop    bx  ; bh = character, bl = color
8418 00002A2F 6659                 <1>       pop    cx
8419 00002A31 FEC9                 <1>       dec    cl
8420 00002A33 7409                 <1>       jz     short vga_wtty_7
8421 00002A35 668B15[E64D0100]     <1>       mov    dx, [CURSOR_POSN] ; new cursor position (pg 0)
8422 00002A3C EBE6                 <1>       jmp    short vga_wtty_6
8423                                <1> vga_wtty_7:
8424 00002A3E C3                   <1>       retn
8425                                <1>       ;
8426                                <1> vga_wtty_8:
8427 00002A3F 83C64F               <1>       add    esi, vga_g_memmodel - (vga_g_modes + 1)
8428                                <1>       ; [ESI] = VGA memory model number (LINEAR8, PLANAR4, PLANAR1)
8429 00002A42 BF00000A00           <1>       mov    edi, 0A0000h
8430                                <1>       ;
8431 00002A47 88F8                 <1>       mov    al, bh ; character
8432                                <1>       ;
8433 00002A49 803E04               <1>       cmp    byte [esi], PLANAR4
8434 00002A4C 7414                 <1>       je     short vga_wtty_planar
8435 00002A4E 803E03               <1>       cmp    byte [esi], PLANAR1
8436 00002A51 740F                 <1>       je     short vga_wtty_planar
8437                                <1> vga_wtty_linear8:
8438                                <1>       ; write_gfx_char_lin(car,attr,xcurs,ycurs,nbcols);
8439                                <1>       ; AL = car, BL = attr (color), DL = xcurs, DH = ycurs,
8440                                <1>       ; [CRT_COLS] = nbcols
8441 00002A53 E8F3FCFFFF           <1>       call   write_gfx_char_lin
8442 00002A58 EB0D                 <1>       jmp    short vga_wtty_9
8443                                <1>
8444                                <1> vga_wtty_12:
8445                                <1>       ; 09/07/2016
8446                                <1>       ; set cursor position
8447                                <1>       ; NOTE: Hardware cursor position will not be set
8448                                <1>       ;   in any VGA modes (>7)
8449                                <1>       ;   But, cursor position will be saved into
8450                                <1>       ;   [CURSOR_POSN].
8451                                <1>       ;   TRDOS 386 (TRDOS v2.0) uses only one page
8452                                <1>       ;   (page 0) for all graphics modes.
8453                                <1>
8454 00002A5A 668915[E64D0100]     <1>       mov    [CURSOR_POSN], dx ; save cursor pos for pg 0
8455                                <1>       ; 04/08/2016
8456                                <1>       ;mov   bh, [ACTIVE_PAGE] ; = 0
8457                                <1>       ;call  _set_cpos
8458 00002A61 C3                   <1>       retn
8459                                <1>
8460                                <1> vga_wtty_planar:
8461                                <1>       ; write_gfx_char_pl4(car,attr,xcurs,ycurs,nbcols,cheight);
8462                                <1>       ; AL = car, BL = attr (color), DL = xcurs, DH = ycurs,
8463                                <1>       ; [CRT_COLS]= nbcols, [CHAR_HEIGHT] = cheight
8464 00002A62 E87BFDFFFF           <1>       call   write_gfx_char_pl4
```

```
8465                                <1> vga_wtty_9:
8466 00002A67 FEC2                 <1>         inc    dl ; xcurs++;
8467                                <1> vga_wtty_10:
8468                                <1>         ; Do we need to wrap ?
8469                                <1>         ; if(xcurs==nbcols)
8470 00002A69 3A15[C45E0000]       <1>         cmp    dl, [CRT_COLS] ; [VGA_COLS]
8471 00002A6F 7204                 <1>         jb     short vga_wtty_11 ; no
8472 00002A71 28D2                 <1>         sub    dl, dl ; xcurs=0;
8473 00002A73 FEC6                 <1>         inc    dh ;  ycurs++;
8474                                <1> vga_wtty_11:
8475                                <1>         ; Do we need to scroll ?
8476                                <1>         ; if(ycurs==nbrows)
8477 00002A75 3A35[CA5E0000]       <1>         cmp    dh, [VGA_ROWS]
8478 00002A7B 72DD                 <1>         jb     short vga_wtty_12 ; no
8479                                <1>         ;
8480                                <1>         ; biosfn_scroll (nblines,attr,rul,cul,rlr,clr,page,dir)
8481                                <1>         ; al = nblines = 1, bl = attr (color) = 0
8482                                <1>         ; ch = rul, cl = cul, dh = rlr, dl = clr, page = 0
8483                                <1>         ; dir = SCROLL_UP
8484                                <1>
8485 00002A7D B001                 <1>         mov    al, 1
8486 00002A7F 28DB                 <1>         sub    bl, bl ; 0 ; blank/black line (attr=0) will be used
8487 00002A81 6629C9               <1>         sub    cx, cx ; 0,0
8488                                <1>
8489                                <1>         ; 06/08/2016
8490 00002A84 8A35[CA5E0000]       <1>         mov    dh, [VGA_ROWS]
8491 00002A8A FECE                 <1>         dec    dh ; nbrows -1
8492                                <1>
8493 00002A8C 6652                 <1>         push   dx     ; 04/08/2016
8494 00002A8E 8A15[C45E0000]       <1>         mov    dl, [CRT_COLS]
8495 00002A94 FECA                 <1>         dec    dl ; nbcols -1
8496                                <1>
8497 00002A96 8A25[C25E0000]       <1>         mov    ah, [CRT_MODE]
8498                                <1>
8499                                <1>         ; biosfn_scroll(0x01,0x00,0,0,nbrows-1,nbcols-1,page,SCROLL_UP);
8500 00002A9C E808F5FFFF           <1>         call   vga_graphics_up
8501                                <1>         ; 04/08/2016
8502 00002AA1 665A                 <1>         pop    dx
8503                                <1>         ;dec   dh ; ycurs-=1
8504 00002AA3 EBB5                 <1>         jmp    short vga_wtty_12
8505                                <1>
8506                                <1> font_setup:
8507                                <1>         ; 09/07/2016
8508                                <1>         ; character generator (font loading) functions
8509                                <1>         ;
8510                                <1>         ; derived from 'Plex86/Bochs VGABios' source code
8511                                <1>         ; vgabios-0.7a (2011)
8512                                <1>         ; by the LGPL VGABios developers Team (2001-2008)
8513                                <1>         ; 'vgabios.c', 'int10_func'
8514                                <1>
8515                                <1>         ; AX = 1100H ; Load User-Defined Font (EGA/VGA)
8516                                <1>         ;
8517                                <1>           ; BH    height of each character (bytes per character definition)
8518                                <1>           ; (BL   font block to load (EGA: 0-3; VGA: 0-7))
8519                                <1>         ; CX    number of characters to redefine (<=256)
8520                                <1>           ; DX    ASCII code of the first character defined at ES:BP
8521                                <1>           ; EBP      address of font-definition information
8522                                <1>         ;      (in user's memory space)
8523                                <1>
8524                                <1>         ; case 0x11:
8525                                <1>         ; switch(GET_AL())
8526                                <1>              ; {
8527                                <1>         ; case 0x00:
8528                                <1>           ; case 0x10:
8529                                <1>           ; biosfn_load_text_user_pat(GET_AL(),ES,BP,CX,DX,GET_BL(),GET_BH());
8530                                <1>           ; break;
8531                                <1>
8532                                <1>         ; AX = 1110H ; Load and Activate User-Defined Font (EGA/VGA)
8533 00002AA5 08C0                 <1>         or     al, al ; 0
8534 00002AA7 7404                 <1>         jz     short font_setup_0
8535 00002AA9 3C10                 <1>         cmp    al, 10h
8536 00002AAB 7511                 <1>         jne    short font_setup_1
8537                                <1> font_setup_0:
8538 00002AAD E8B7000000           <1>         call   transfer_user_fonts
8539 00002AB2 721C                 <1>         jc     short font_setup_error
8540 00002AB4 E8C2000000           <1>         call   load_text_user_pat
8541 00002AB9 E996EAFFFF           <1>           jmp     VIDEO_RETURN
8542                                <1> font_setup_1:
8543                                <1>         ; AX = 1101H ; Load ROM 8x14 Character Set (EGA/VGA)
8544                                <1>         ; case 0x01:
8545                                <1>           ; case 0x11:
8546                                <1>           ; biosfn_load_text_8_14_pat(GET_AL(),GET_BL());
8547                                <1>           ; break;
8548 00002ABE 3C01                 <1>         cmp    al, 1
8549 00002AC0 7404                 <1>         je     short font_setup_2
8550 00002AC2 3C11                 <1>         cmp    al, 11h
8551 00002AC4 7511                 <1>         jne    short font_setup_3
8552                                <1> font_setup_2:
8553                                <1>         ; AX = 1111H ; Load and Activate ROM 8x14 Character Set (EGA/VGA)
8554                                <1>         ; (BL = font block to load (EGA: 0-3; VGA: 0-7))
8555 00002AC6 E8EE010000           <1>         call   load_text_8_14_pat
8556 00002ACB E984EAFFFF           <1>           jmp     VIDEO_RETURN
8557                                <1> font_setup_error:
8558 00002AD0 29C0                 <1>         sub    eax, eax ; 0 -> fonts could not be loaded
8559 00002AD2 E982EAFFFF           <1>         jmp    _video_return
8560                                <1> font_setup_3:
8561                                <1>         ; AX = 1102H ; Load ROM 8x8 Character Set (EGA/VGA)
8562                                <1>         ; case 0x02:
8563                                <1>           ; case 0x12:
8564                                <1>           ; biosfn_load_text_8_8_pat(GET_AL(),GET_BL());
8565                                <1>           ; break;
8566 00002AD7 3C02                 <1>         cmp    al, 2
```

```
8567 00002AD9 7404                <1>       je    short font_setup_4
8568 00002ADB 3C12                <1>       cmp   al, 12h
8569 00002ADD 750A                <1>       jne   short font_setup_5
8570                              <1> font_setup_4:
8571                              <1>       ; AX = 1112H ; Load and Activate ROM 8x8 Character Set (EGA/VGA)
8572                              <1>       ; (BL = font block to load (EGA: 0-3; VGA: 0-7))
8573 00002ADF E805020000          <1>       call  load_text_8_8_pat
8574 00002AE4 E96BEAFFFF          <1>         jmp     VIDEO_RETURN
8575                              <1> font_setup_5:
8576                              <1>       ; AX = 1104H ; Load ROM 8x16 Character Set (EGA/VGA)
8577                              <1>       ; case 0x04:
8578                              <1>         ; case 0x14:
8579                              <1>         ; biosfn_load_text_8_16_pat(GET_AL(),GET_BL());
8580                              <1>         ; break;
8581 00002AE9 3C04                <1>       cmp   al, 4
8582 00002AEB 7404                <1>       je    short font_setup_6
8583 00002AED 3C14                <1>       cmp   al, 14h
8584 00002AEF 750A                <1>       jne   short font_setup_7
8585                              <1> font_setup_6:
8586                              <1>       ; AX = 1114H ; Load and Activate ROM 8x16 Character Set (EGA/VGA)
8587                              <1>       ; (BL = font block to load (EGA: 0-3; VGA: 0-7))
8588 00002AF1 E823020000          <1>       call  load_text_8_16_pat
8589 00002AF6 E959EAFFFF          <1>         jmp     VIDEO_RETURN
8590                              <1> font_setup_7:
8591                              <1>       ; Note: AX=1120h (Setup INT 1Fh, EXT_PTR) is not needed
8592                              <1>       ; for TRDOS 386 (TRDIOS v2.0) video functionality;
8593                              <1>       ; because, originally EXT_PTR (font address) was used for
8594                              <1>       ; chars 80h to 0FFh (after the first 128 ASCII char fonts), for
8595                              <1>       ; CGA graphics mode; currenty, 'vgafont8' address has 256 chars!
8596                              <1>       ;
8597                              <1>       ; case 0x20:
8598                              <1>         ; biosfn_load_gfx_8_8_chars(ES,BP);
8599                              <1>         ; break;
8600                              <1>       ; case 0x21:
8601                              <1>         ; biosfn_load_gfx_user_chars(ES,BP,CX,GET_BL(),GET_DL());
8602                              <1>         ; break;
8603                              <1>       ; AX = 1121H ; Setup User-Defined Font for Graphics Mode (VGA)
8604                              <1>       ; BL   screen rows code: 00H = user-specified (in DL)
8605                              <1>         ;                      01H = 14 rows
8606                              <1>         ;                      02H = 25 rows
8607                              <1>         ;                      03H = 43 rows
8608                              <1>         ; CX   bytes per character definition
8609                              <1>         ; DL   (when BL=0) custom number of character rows on screen
8610                              <1>         ; EBP  address of font-definition information (user's mem space)
8611                              <1>
8612 00002AFB 3C21               <1>       cmp   al, 21h
8613 00002AFD 751A                <1>       jne   short font_setup_9
8614                              <1>
8615                              <1>       ; TRDOS 386 modification !
8616                              <1>       ; dh = 0 -> 256 characters
8617                              <1>       ; dh = 80h -> 128 characters
8618                              <1>       ; (If DH <> 0 and DH <> 80h -> invalid)
8619 00002AFF 20F6                <1>       and   dh, dh
8620 00002B01 7405                <1>       jz    short font_setup_8 ; 256 characters
8621 00002B03 80FE80              <1>       cmp   dh, 80h ; 128 characters
8622 00002B06 75C8                <1>       jne   short font_setup_error ; invalid !
8623                              <1> font_setup_8:
8624 00002B08 E85C000000          <1>       call  transfer_user_fonts
8625 00002B0D 72C1                <1>       jc    short font_setup_error
8626                              <1>       ; ebp = user's font data address in system's memory space
8627 00002B0F E836020000          <1>       call  load_gfx_user_chars
8628 00002B14 E93BEAFFFF          <1>         jmp     VIDEO_RETURN
8629                              <1> font_setup_9:
8630                              <1>       ; case 0x22:
8631                              <1>         ; biosfn_load_gfx_8_14_chars(GET_BL());
8632                              <1>         ; break;
8633 00002B19 3C22                <1>       cmp   al, 22h
8634 00002B1B 750A                <1>       jne   short font_setup_10
8635 00002B1D E866020000          <1>       call  load_gfx_8_14_chars
8636 00002B22 E92DEAFFFF          <1>         jmp     VIDEO_RETURN
8637                              <1> font_setup_10:
8638                              <1>       ; case 0x23:
8639                              <1>         ; biosfn_load_gfx_8_8_dd_chars(GET_BL());
8640                              <1>         ; break;
8641 00002B27 3C23                <1>       cmp   al, 23h
8642 00002B29 750A                <1>       jne   short font_setup_11
8643 00002B2B E899020000          <1>       call  load_gfx_8_8_chars
8644 00002B30 E91FEAFFFF          <1>         jmp     VIDEO_RETURN
8645                              <1> font_setup_11:
8646                              <1>       ; case 0x24:
8647                              <1>         ; biosfn_load_gfx_8_16_chars(GET_BL());
8648                              <1>         ; break;
8649 00002B35 3C24                <1>       cmp   al, 24h
8650 00002B37 750A                <1>       jne   short font_setup_12
8651 00002B39 E8CC020000          <1>       call  load_gfx_8_16_chars
8652 00002B3E E911EAFFFF          <1>         jmp     VIDEO_RETURN
8653                              <1> font_setup_12:
8654                              <1>       ; case 0x30:
8655                              <1>         ; biosfn_get_font_info(GET_BH(),&ES,&BP,&CX,&DX);
8656                              <1>         ; break;
8657 00002B43 3C30                <1>       cmp   al, 30h
8658 00002B45 750A                <1>       jne   short font_setup_13
8659 00002B47 E8FF020000          <1>       call  get_font_info
8660                              <1>       ; eax = return value (info: 4 bytes for 4 parms)
8661                              <1>       ; eax = 0 -> invalid function (input)
8662 00002B4C E908EAFFFF          <1>         jmp     _video_return
8663                              <1> font_setup_13:
8664 00002B51 3C03                <1>       cmp   al, 03h ; AX = 1103h
8665 00002B53 750D                <1>       jne   short font_setup_14
8666                              <1>       ; biosfn_set_text_block_specifier:
8667                              <1>       ; BL = font block selector code
8668                              <1>       ; NOTE: TRDOS 386 only uses and sets font block 0
```

```
                        <1>       ; (It is as BL = 0 for TRDOS 386)
00002B55 66BAC403       <1>       mov    dx, 3C4h ; VGAREG_SEQU_ADDRESS
                        <1>       ;mov   ah, bl
00002B59 28E4           <1>       sub    ah, ah ; 0
                        <1>       ;mov   al, 03h
00002B5B 66EF           <1>       out    dx, ax
00002B5D E9F2E9FFFF     <1>       jmp     VIDEO_RETURN
                        <1>
                        <1> font_setup_14:
00002B62 29C0           <1>       sub    eax, eax ; 0 = invalid function
00002B64 E9F0E9FFFF     <1>         jmp     _video_return
                        <1>
                        <1> transfer_user_fonts:
                        <1>       ; 09/07/2016
                        <1>       ;and   ecx, 0FFFFh
                        <1>       ; ECX = byte count
                        <1>       ;push  ecx
00002B69 89EE           <1>       mov    esi, ebp ; user buffer
00002B6B BF00000700     <1>       mov    edi, Cluster_Buffer  ; system buffer
00002B70 E857BD0000     <1>       call   transfer_from_user_buffer
                        <1>       ;pop   ecx
                        <1>       ; ecx = transfer (byte) count = character count
00002B75 BD00000700     <1>       mov    ebp, Cluster_Buffer
                        <1>       ; jc   VIDEO_RETURN -> failed
00002B7A C3             <1>       retn
                        <1>
                        <1> load_text_user_pat:
                        <1>       ; 26/07/2016
                        <1>       ; 09/07/2016
                        <1>       ; load user defined (EGA/VGA) text fonts
                        <1>       ;
                        <1>       ; derived from 'Plex86/Bochs VGABios' source code
                        <1>       ; vgabios-0.7a (2011)
                        <1>       ; by the LGPL VGABios developers Team (2001-2008)
                        <1>       ; 'vgabios.c', 'biosfn_load_text_user_pat'
                        <1>
                        <1>       ; biosfn_load_text_user_pat (AL,ES,BP,CX,DX,BL,BH)
                        <1>
                        <1>       ; get_font_access();
                        <1>       ; blockaddr = ((BL & 0x03) << 14) + ((BL & 0x04) << 11);
                        <1>       ; for(i=0;i<CX;i++)
                        <1>       ; {
                        <1>       ;  src = BP + i * BH;
                        <1>       ;  dest = blockaddr + (DX + i) * 32;
                        <1>       ;  memcpyb(0xA000, dest, ES, src, BH);
                        <1>       ; }
                        <1>       ; release_font_access();
                        <1>       ; if(AL>=0x10)
                        <1>       ; {
                        <1>       ; set_scan_lines(BH);
                        <1>       ; }
                        <1>
00002B7B 50             <1>       push   eax
00002B7C E83C000000     <1>       call   get_font_access
00002B81 28DB           <1>       sub    bl, bl ; i = 0
                        <1> ltup_1:
00002B83 88D8           <1>       mov    al, bl
00002B85 F6E7           <1>       mul    bh
00002B87 0FB7F0         <1>       movzx  esi, ax
00002B8A 01EE           <1>       add    esi, ebp
00002B8C 88D8           <1>       mov    al, bl
00002B8E 28E4           <1>       sub    ah, ah
00002B90 6601D0         <1>       add    ax, dx ; (DX + i)
00002B93 66C1E005       <1>       shl    ax, 5  ; * 32
00002B97 0FB7F8         <1>       movzx  edi, ax
00002B9A 81C700000A00   <1>       add    edi, 0A0000h
00002BA0 51             <1>       push   ecx
00002BA1 0FB6CF         <1>       movzx  ecx, bh
00002BA4 F3A4           <1>       rep    movsb
00002BA6 59             <1>       pop    ecx
00002BA7 FEC3           <1>       inc    bl
00002BA9 38CB           <1>       cmp    bl, cl
00002BAB 75D6           <1>       jne    short ltup_1
                        <1>       ;
00002BAD E840000000     <1>       call   release_font_access
                        <1>       ;
00002BB2 58             <1>       pop    eax
                        <1>       ; if(AL>=0x10)
00002BB3 3C10           <1>       cmp    al, 10h
00002BB5 7205           <1>       jb     short ltup_2
                        <1>       ; set_scan_lines(BH);
00002BB7 E875000000     <1>       call   set_scan_lines
                        <1> ltup_2:
00002BBC C3             <1>       retn
                        <1>
                        <1> get_font_access:
                        <1>       ; 09/07/2016
                        <1>       ;
                        <1>       ; derived from 'Plex86/Bochs VGABios' source code
                        <1>       ; vgabios-0.7a (2011)
                        <1>       ; by the LGPL VGABios developers Team (2001-2008)
                        <1>       ; 'vgabios.c', 'get_font_access'
                        <1>
                        <1>       ; get_font_access()
00002BBD 52             <1>       push   edx
00002BBE 66BAC403       <1>       mov    dx, 3C4h ; VGAREG_SEQU_ADDRESS
00002BC2 66B80001       <1>       mov    ax, 0100h
00002BC6 66EF           <1>       out    dx, ax
00002BC8 66B80204       <1>       mov    ax, 0402h
00002BCC 66EF           <1>       out    dx, ax
00002BCE 66B80407       <1>       mov    ax, 0704h
00002BD2 66EF           <1>       out    dx, ax
```

```
8771 00002BD4 66B80003            <1>       mov    ax, 0300h
8772 00002BD8 66EF                <1>       out    dx, ax
8773 00002BDA 66BACE03            <1>       mov    dx, 3CEh ; VGAREG_GRDC_ADDRESS
8774 00002BDE 66B80402            <1>       mov    ax, 0204h
8775 00002BE2 66EF                <1>       out    dx, ax
8776 00002BE4 66B80500            <1>       mov    ax, 0005h
8777 00002BE8 66EF                <1>       out    dx, ax
8778 00002BEA 66B80604            <1>       mov    ax, 0406h
8779 00002BEE 66EF                <1>       out    dx, ax
8780 00002BF0 5A                  <1>       pop    edx
8781 00002BF1 C3                  <1>       retn
8782                              <1>
8783                              <1> release_font_access:
8784                              <1>       ; 29/07/2016
8785                              <1>       ; 09/07/2016
8786                              <1>       ;
8787                              <1>       ; derived from 'Plex86/Bochs VGABios' source code
8788                              <1>       ; vgabios-0.7a (2011)
8789                              <1>       ; by the LGPL VGABios developers Team (2001-2008)
8790                              <1>       ; 'vgabios.c', 'release_font_access'
8791                              <1>
8792 00002BF2 66BAC403            <1>       mov    dx, 3C4h ; VGAREG_SEQU_ADDRESS
8793 00002BF6 66B80001            <1>       mov    ax, 0100h
8794 00002BFA 66EF                <1>       out    dx, ax
8795 00002BFC 66B80203            <1>       mov    ax, 0302h
8796 00002C00 66EF                <1>       out    dx, ax
8797 00002C02 66B80403            <1>       mov    ax, 0304h
8798 00002C06 66EF                <1>       out    dx, ax
8799 00002C08 66B80003            <1>       mov    ax, 0300h
8800 00002C0C 66EF                <1>       out    dx, ax
8801 00002C0E 66BACC03            <1>       mov    dx, 3CCh ; VGAREG_READ_MISC_OUTPUT
8802 00002C12 EC                  <1>       in     al, dx
8803 00002C13 2401                <1>       and    al, 01h
8804 00002C15 C0E002              <1>       shl    al, 2
8805 00002C18 0C0A                <1>       or     al, 0Ah
8806 00002C1A 88C4                <1>       mov    ah, al
8807 00002C1C B006                <1>       mov    al, 06h
8808 00002C1E 66BACE03            <1>       mov    dx, 3CEh ; VGAREG_GRDC_ADDRESS
8809 00002C22 66EF                <1>       out    dx, ax
8810 00002C24 66B80400            <1>       mov    ax, 0004h
8811 00002C28 66EF                <1>       out    dx, ax
8812 00002C2A 66B80510            <1>       mov    ax, 1005h
8813 00002C2E 66EF                <1>       out    dx, ax
8814 00002C30 C3                  <1>       retn
8815                              <1>
8816                              <1> set_scan_lines:
8817                              <1>       ; 09/07/2016
8818                              <1>       ;
8819                              <1>       ; derived from 'Plex86/Bochs VGABios' source code
8820                              <1>       ; vgabios-0.7a (2011)
8821                              <1>       ; by the LGPL VGABios developers Team (2001-2008)
8822                              <1>       ; 'vgabios.c', 'set_scan_lines'
8823                              <1>
8824                              <1>       ; set_scan_lines(lines)
8825                              <1>       ; BH = lines
8826                              <1>
8827                              <1>       ; outb(crtc_addr, 0x09);
8828 00002C31 66BAD403            <1>       mov    dx, 3D4h ; CRTC_ADDRESS = 3D4h (always)
8829 00002C35 B009                <1>       mov    al, 09h
8830 00002C37 EE                  <1>       out    dx, al
8831                              <1>       ; crtc_r9 = inb(crtc_addr+1);
8832 00002C38 6642                <1>       inc    dx ; 3D5h
8833 00002C3A EC                  <1>       in     al, dx
8834                              <1>       ; crtc_r9 = (crtc_r9 & 0xe0) | (lines - 1);
8835 00002C3B 24E0                <1>       and    al, 0E0h
8836 00002C3D FECF                <1>       dec    bh ; lines - 1
8837 00002C3F 08F8                <1>       or     al, bh
8838                              <1>       ; outb(crtc_addr+1, crtc_r9);
8839 00002C41 EE                  <1>       out    dx, al
8840                              <1>       ;inc   bh
8841                              <1>       ; if(lines==8)
8842                              <1>       ;cmp   bh, 8
8843 00002C42 80FF07              <1>       cmp    bh, 7
8844 00002C45 7506                <1>       jne    short ssl_1
8845                              <1>       ; biosfn_set_cursor_shape(0x06,0x07);
8846 00002C47 66B90706            <1>       mov    cx, 0607h
8847 00002C4B EB06                <1>       jmp    short ssl_2
8848                              <1> ssl_1:
8849                              <1>       ; biosfn_set_cursor_shape(lines-4,lines-3);
8850 00002C4D 88F9                <1>       mov    cl, bh ; lines - 1
8851 00002C4F 88CD                <1>       mov    ch, cl ; lines - 1 (16 -> 15)
8852 00002C51 FECD                <1>       dec    ch  ; lines - 2 (16 -> 14)
8853                              <1> ssl_2:
8854                              <1>       ; CH = start line, CL = stop line
8855 00002C53 B40A                <1>       mov    ah, 10 ; 6845 register for cursor set
8856 00002C55 66890D[DB5E0000]    <1>       mov    [CURSOR_MODE], cx ; save in data area
8857 00002C5C E812F1FFFF          <1>       call   m16    ; output cx register
8858                              <1>       ; write_word(BIOSMEM_SEG,BIOSMEM_CHAR_HEIGHT, lines);
8859 00002C61 FEC7                <1>       inc    bh ; lines
8860 00002C63 883D[C65E0000]      <1>       mov    [CHAR_HEIGHT], bh
8861                              <1>       ;  outb(crtc_addr, 0x12);
8862 00002C69 66BAD403            <1>       mov    dx, 3D4h ; CRTC_ADDRESS
8863 00002C6D B012                <1>       mov    al, 12h
8864 00002C6F EE                  <1>       out    dx, al
8865                              <1>       ; vde = inb(crtc_addr+1);
8866 00002C70 6642                <1>       inc    dx
8867 00002C72 EC                  <1>       in     al, dx
8868 00002C73 88C4                <1>       mov    ah, al
8869                              <1>       ; outb(crtc_addr, 0x07);
8870 00002C75 664A                <1>       dec    dx
8871 00002C77 B007                <1>       mov    al, 07h
8872 00002C79 EE                  <1>       out    dx, al
```

```
8873                                    <1>       ; ovl = inb(crtc_addr+1);
8874 00002C7A 6642                     <1>       inc    dx
8875 00002C7C EC                       <1>       in     al, dx
8876                                    <1>       ; vde += (((ovl & 0x02) << 7) + ((ovl & 0x40) << 3) + 1);
8877 00002C7D 88E2                     <1>       mov    dl, ah ; vde
8878 00002C7F 88C6                     <1>       mov    dh, al ; ovl
8879 00002C81 6683E002                 <1>       and    ax, 02h
8880 00002C85 66C1E007                 <1>       shl    ax, 7
8881 00002C89 6689C1                   <1>       mov    cx, ax ; (ovl & 0x02) << 7)
8882 00002C8C 88F0                     <1>       mov    al, dh ; ovl
8883 00002C8E 6683E040                 <1>       and    ax, 40h
8884 00002C92 66C1E003                 <1>       shl    ax, 3  ; (ovl & 0x40) << 3)
8885 00002C96 6640                     <1>       inc    ax ; + 1
8886 00002C98 6601C8                   <1>       add    ax, cx
8887 00002C9B 30F6                     <1>       xor    dh, dh
8888 00002C9D 6601D0                   <1>       add    ax, dx ; + vde
8889                                    <1>       ; rows = vde / lines;
8890 00002CA0 F6F7                     <1>       div    bh
8891                                    <1>       ;dec   al ; rows -1
8892                                    <1>       ; write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, rows-1);
8893 00002CA2 A2[CA5E0000]             <1>       mov    [VGA_ROWS], al ; rows (not 'rows-1' !)
8894                                    <1>       ; write_word(BIOSMEM_SEG,BIOSMEM_PAGE_SIZE, rows * cols * 2);
8895 00002CA7 8A25[C45E0000]           <1>       mov    ah, [CRT_COLS]
8896 00002CAD F6E4                     <1>       mul    ah
8897 00002CAF 66D1E0                   <1>       shl    ax, 1
8898 00002CB2 66A3[645B0100]           <1>       mov    [CRT_LEN], ax
8899 00002CB8 C3                       <1>       retn
8900                                    <1>
8901                                    <1> load_text_8_14_pat:
8902                                    <1>       ; 26/07/2016
8903                                    <1>       ; 25/07/2016
8904                                    <1>       ; 23/07/2016
8905                                    <1>       ; 09/07/2016
8906                                    <1>       ; load user defined (EGA/VGA) text fonts
8907                                    <1>       ;
8908                                    <1>       ; derived from 'Plex86/Bochs VGABios' source code
8909                                    <1>       ; vgabios-0.7a (2011)
8910                                    <1>       ; by the LGPL VGABios developers Team (2001-2008)
8911                                    <1>       ; 'vgabios.c', 'biosfn_load_text_8_14_pat'
8912                                    <1>
8913                                    <1>       ; biosfn_load_text_8_14_pat (AL,BL)
8914                                    <1>
8915                                    <1>       ; get_font_access();
8916                                    <1>       ; blockaddr = ((BL & 0x03) << 14) + ((BL & 0x04) << 11);
8917                                    <1>       ; for(i=0;i<0x100;i++)
8918                                    <1>       ; {
8919                                    <1>       ;  src = i * 14;
8920                                    <1>       ;  dest = blockaddr + i * 32;
8921                                    <1>       ;  memcpyb(0xA000, dest, 0xC000, vgafont14+src, 14);
8922                                    <1>       ; }
8923                                    <1>       ; release_font_access();
8924                                    <1>       ; if(AL>=0x10)
8925                                    <1>       ; {
8926                                    <1>       ; set_scan_lines(14);
8927                                    <1>       ; }
8928                                    <1>
8929 00002CB9 50                       <1>       push   eax
8930 00002CBA E8FEFEFFFF               <1>       call   get_font_access
8931                                    <1>
8932                                    <1>       ; blockaddr = ((BL & 0x03) << 14) + ((BL & 0x04) << 11);
8933                                    <1>       ;mov   dl, bl
8934                                    <1>       ;and   dl, 3
8935                                    <1>       ;shl   dx, 14
8936                                    <1>       ;xchg  dx, bx
8937                                    <1>       ;and   dl, 4
8938                                    <1>       ;shl   dx, 11
8939                                    <1>       ;add   dx, bx
8940                                    <1>
8941                                    <1>       ;xor   dx, dx  ; blockaddr = 0
8942                                    <1>       ; Always block 0 for TRDOS 386 ! (blockaddr=0(
8943                                    <1>
8944 00002CBF 28DB                     <1>       sub    bl, bl ; i = 0
8945 00002CC1 B70E                     <1>       mov    bh, 14
8946 00002CC3 BE[442A0100]             <1>       mov    esi, vgafont14
8947 00002CC8 BF00000A00               <1>       mov    edi, 0A0000h
8948                                    <1> lt8_14_1:
8949                                    <1>       ;mov   al, bl
8950                                    <1>       ;mul   bh
8951                                    <1>       ;movzx esi, ax
8952                                    <1>       ;add   esi, vgafont14
8953                                    <1>       ;mov   al, bl
8954                                    <1>       ;sub   ah, ah
8955                                    <1>       ;shl   ax, 5 ; * 32
8956                                    <1>       ;;add  ax, dx ; blockaddr + i * 32;
8957                                    <1>       ;movzx edi, ax ; dest
8958                                    <1>       ;add   edi, 0A0000h
8959 00002CCD 0FB6CF                   <1>       movzx  ecx, bh
8960 00002CD0 F3A4                     <1>       rep    movsb
8961 00002CD2 83C712                   <1>       add    edi, 18 ; 32 - 14
8962 00002CD5 FEC3                     <1>       inc    bl
8963 00002CD7 75F4                     <1>       jnz    short lt8_14_1
8964                                    <1>       ;
8965 00002CD9 E814FFFFFF               <1>       call   release_font_access
8966                                    <1>       ;
8967 00002CDE 58                       <1>       pop    eax
8968                                    <1>       ; if(AL>=0x10)
8969 00002CDF 3C10                     <1>       cmp    al, 10h
8970 00002CE1 7205                     <1>       jb     short lt8_14_4
8971                                    <1>       ; BH = 14
8972                                    <1>       ; set_scan_lines(14);
8973 00002CE3 E849FFFFFF               <1>       call   set_scan_lines
8974                                    <1> lt8_14_4:
```

```
8975 00002CE8 C3                <1>       retn
8976                            <1>
8977                            <1> load_text_8_8_pat:
8978                            <1>       ; 26/07/2016
8979                            <1>       ; 25/07/2016
8980                            <1>       ; 23/07/2016
8981                            <1>       ; 09/07/2016
8982                            <1>       ; load user defined (EGA/VGA) text fonts
8983                            <1>       ;
8984                            <1>       ; derived from 'Plex86/Bochs VGABios' source code
8985                            <1>       ; vgabios-0.7a (2011)
8986                            <1>       ; by the LGPL VGABios developers Team (2001-2008)
8987                            <1>       ; 'vgabios.c', 'biosfn_load_text_8_8_pat'
8988                            <1>
8989                            <1>       ; biosfn_load_text_8_8_pat (AL,BL)
8990                            <1>
8991                            <1>       ; get_font_access();
8992                            <1>       ; blockaddr = ((BL & 0x03) << 14) + ((BL & 0x04) << 11);
8993                            <1>       ; for(i=0;i<0x100;i++)
8994                            <1>       ; {
8995                            <1>       ;  src = i * 8;
8996                            <1>       ;  dest = blockaddr + i * 32;
8997                            <1>       ;  memcpyb(0xA000, dest, 0xC000, vgafont8+src, 8);
8998                            <1>       ; }
8999                            <1>       ; release_font_access();
9000                            <1>       ; if(AL>=0x10)
9001                            <1>       ; {
9002                            <1>       ; set_scan_lines(8);
9003                            <1>       ; }
9004                            <1>
9005 00002CE9 50              <1>       push   eax
9006 00002CEA E8CEFEFFFF      <1>       call   get_font_access
9007                            <1>
9008                            <1>       ; blockaddr = ((BL & 0x03) << 14) + ((BL & 0x04) << 11);
9009                            <1>       ;mov   dl, bl
9010                            <1>       ;and   dl, 3
9011                            <1>       ;shl   dx, 14
9012                            <1>       ;xchg  dx, bx
9013                            <1>       ;and   dl, 4
9014                            <1>       ;shl   dx, 11
9015                            <1>       ;add   dx, bx
9016                            <1>
9017                            <1>       ;xor   dx, dx  ; blockaddr = 0
9018                            <1>       ; Always block 0 for TRDOS 386 ! (blockaddr=0(
9019                            <1>
9020 00002CEF 28DB              <1>       sub    bl, bl ; i = 0
9021 00002CF1 B708              <1>       mov    bh, 8
9022 00002CF3 BE[44220100]      <1>       mov    esi, vgafont8
9023 00002CF8 BF00000A00        <1>       mov    edi, 0A0000h
9024                            <1> lt8_8_1:
9025                            <1>       ;mov   al, bl
9026                            <1>       ;mul   bh
9027                            <1>       ;movzx esi, ax
9028                            <1>       ;add   esi, vgafont8
9029                            <1>       ;mov   al, bl
9030                            <1>       ;sub   ah, ah
9031                            <1>       ;shl   ax, 5 ; * 32
9032                            <1>       ;;add  ax, dx ; blockaddr + i * 32;
9033                            <1>       ;movzx edi, ax ; dest
9034                            <1>       ;add   edi, 0A0000h
9035 00002CFD 0FB6CF            <1>       movzx  ecx, bh
9036 00002D00 F3A4              <1>       rep    movsb
9037 00002D02 83C718            <1>       add    edi, 24 ; 32 - 8
9038 00002D05 FEC3              <1>       inc    bl
9039 00002D07 75F4              <1>       jnz    short lt8_8_1
9040                            <1>       ;
9041 00002D09 E8E4FEFFFF        <1>       call   release_font_access
9042                            <1>       ;
9043 00002D0E 58                <1>       pop    eax
9044                            <1>       ; if(AL>=0x10)
9045 00002D0F 3C10              <1>       cmp    al, 10h
9046 00002D11 7205              <1>       jb     short lt8_8_2
9047                            <1>       ; BH = 8
9048                            <1>       ; set_scan_lines(8);
9049 00002D13 E819FFFFFF        <1>       call   set_scan_lines
9050                            <1> lt8_8_2:
9051 00002D18 C3                <1>       retn
9052                            <1>
9053                            <1> load_text_8_16_pat:
9054                            <1>       ; 26/07/2016
9055                            <1>       ; 25/07/2016
9056                            <1>       ; 23/07/2016
9057                            <1>       ; 09/07/2016
9058                            <1>       ; load user defined (EGA/VGA) text fonts
9059                            <1>       ;
9060                            <1>       ; derived from 'Plex86/Bochs VGABios' source code
9061                            <1>       ; vgabios-0.7a (2011)
9062                            <1>       ; by the LGPL VGABios developers Team (2001-2008)
9063                            <1>       ; 'vgabios.c', 'biosfn_load_text_8_16_pat'
9064                            <1>
9065                            <1>       ; biosfn_load_text_8_16_pat (AL,BL)
9066                            <1>
9067                            <1>       ; get_font_access();
9068                            <1>       ; blockaddr = ((BL & 0x03) << 14) + ((BL & 0x04) << 11);
9069                            <1>       ; for(i=0;i<0x100;i++)
9070                            <1>       ; {
9071                            <1>       ;  src = i * 16;
9072                            <1>       ;  dest = blockaddr + i * 32;
9073                            <1>       ;  memcpyb(0xA000, dest, 0xC000, vgafont16+src, 16);
9074                            <1>       ; }
9075                            <1>       ; release_font_access();
9076                            <1>       ; if(AL>=0x10)
```

```
                                <1>       ; {
                                <1>       ; set_scan_lines(16);
                                <1>       ; }
                                <1>
00002D19 50                     <1>       push   eax
00002D1A E89EFEFFFF             <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>
00002D1F 28DB                   <1>       sub    bl, bl ; i = 0
00002D21 B710                   <1>       mov    bh, 16
00002D23 BE[44380100]           <1>       mov    esi, vgafont16
00002D28 BF00000A00             <1>       mov    edi, 0A0000h
00002D2D 0FB6C7                 <1>       movzx  eax, bh
                                <1> lt8_16_1:
                                <1>       ;mov   al, bl
                                <1>       ;mul   bh
                                <1>       ;movzx esi, ax
                                <1>       ;add   esi, vgafont16
                                <1>       ;mov   al, bl ; i
                                <1>       ;sub   ah, ah
                                <1>       ;shl   ax, 5 ; * 32
                                <1>       ;;add  ax, dx ; blockaddr + i * 32;
                                <1>       ;movzx edi, ax ; dest
                                <1>       ;add   edi, 0A0000h
                                <1>       ;movzx ecx, bh
00002D30 89C1                   <1>       mov    ecx, eax ; 16
00002D32 F3A4                   <1>       rep    movsb
00002D34 01C7                   <1>       add    edi, eax ; add edi, 16
00002D36 FEC3                   <1>       inc    bl
00002D38 75F6                   <1>       jnz    short lt8_16_1
                                <1>       ;
00002D3A E8B3FEFFFF             <1>       call   release_font_access
                                <1>       ;
00002D3F 58                     <1>       pop    eax
                                <1>       ; if(AL>=0x10)
00002D40 3C10                   <1>       cmp    al, 10h
00002D42 7205                   <1>       jb     short lt8_16_2
                                <1>       ; BH = 16
                                <1>       ; set_scan_lines(16);
00002D44 E8E8FEFFFF             <1>       call   set_scan_lines
                                <1> lt8_16_2:
00002D49 C3                     <1>       retn
                                <1>
                                <1> load_gfx_user_chars:
                                <1>       ; 08/08/2016
                                <1>       ; 10/07/2016
                                <1>       ; Setup User-Defined Font for Graphics Mode (VGA)
                                <1>       ;
                                <1>       ; derived from 'Plex86/Bochs VGABios' source code
                                <1>       ; vgabios-0.7a (2011)
                                <1>       ; by the LGPL VGABios developers Team (2001-2008)
                                <1>       ; 'vgabios.c', 'biosfn_load_gfx_user_chars'
                                <1>
                                <1>       ; biosfn_load_gfx_user_chars (ES,BP,CX,BL,DL)
                                <1>       ; /* set 0x43 INT pointer */
                                <1>       ; write_word(0x0, 0x43*4, BP);
                                <1>       ; write_word(0x0, 0x43*4+2, ES);
00002D4A 31C0                   <1>       xor    eax, eax
00002D4C 48                     <1>       dec    eax ; 0FFFFFFFFh (user defined fonts)
00002D4D A3[765B0100]           <1>       mov    [VGA_INT43H], eax
                                <1>
                                <1>       ; BL   screen rows code: 00H = user-specified (in DL)
                                <1>         ;                      01H = 14 rows
                                <1>         ;                      02H = 25 rows
                                <1>         ;                      03H = 43 rows
                                <1>         ; CX   bytes per character definition
                                <1>         ; DL   (when BL=0) custom number of character rows on screen
                                <1>       ; dh = 0 -> 256 characters
                                <1>       ; dh = 80h -> 128 characters
                                <1>       ; (If DH <> 0 and DH <> 80h -> invalid)
                                <1>         ; EBP  address of font-definition information (user's mem space)
                                <1>
                                <1>       ; switch (BL) {
                                <1>       ; case 0:
                                <1>       ;    write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, DL-1);
                                <1>       ;    break;
00002D52 20DB                   <1>       and    bl, bl
00002D54 7508                   <1>       jnz    short l_gfx_uc_1
00002D56 8815[CA5E0000]         <1>       mov    [VGA_ROWS], dl  ; not DL-1 !
00002D5C EB23                   <1>       jmp    short l_gfx_uc_4
                                <1> l_gfx_uc_1:
                                <1>       ; case 1:
                                <1>       ;    write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, 13);
                                <1>       ;    break;
00002D5E FECB                   <1>       dec    bl
00002D60 7509                   <1>       jnz    short l_gfx_uc_2
                                <1>       ; bl = 1
00002D62 C605[CA5E0000]0E       <1>       mov    byte [VGA_ROWS], 14  ; not 13 !
00002D69 EB16                   <1>       jmp    short l_gfx_uc_4
                                <1> l_gfx_uc_2:
00002D6B FECB                   <1>       dec    bl
```

```
9179 00002D6D 740B                <1>       jz    short l_gfx_uc_3 ; bl = 2
9180 00002D6F FECB                <1>       dec   bl
9181 00002D71 750E                <1>       jnz   short l_gfx_uc_4 ; bl > 3
9182                              <1>       ; bl = 3
9183                              <1>       ; case 3:
9184                              <1>       ;    write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, 42);
9185                              <1>       ;    break;
9186 00002D73 C605[CA5E0000]2B    <1>       mov   byte [VGA_ROWS], 43  ; not 42 !
9187                              <1> l_gfx_uc_3:
9188                              <1>       ; case 2:
9189                              <1>       ; default:
9190                              <1>       ;    write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, 24);
9191                              <1>       ;    break;
9192                              <1>       ; bl = 2 or bl > 3
9193 00002D7A C605[CA5E0000]19    <1>       mov   byte [VGA_ROWS], 25  ; not 24 !
9194                              <1>       ; }
9195                              <1> l_gfx_uc_4:
9196                              <1>       ; write_byte(BIOSMEM_SEG, BIOSMEM_CHAR_HEIGHT, CX);
9197 00002D81 880D[C65E0000]      <1>       mov   [CHAR_HEIGHT], cl
9198                              <1>       ; }
9199 00002D87 C3                  <1>       retn
9200                              <1>
9201                              <1> load_gfx_8_14_chars:
9202                              <1>       ; 08/08/2016
9203                              <1>       ; 10/07/2016
9204                              <1>       ; Setup ROM 8x14 Font for Graphics Mode (VGA)
9205                              <1>       ;
9206                              <1>       ; derived from 'Plex86/Bochs VGABios' source code
9207                              <1>       ; vgabios-0.7a (2011)
9208                              <1>       ; by the LGPL VGABios developers Team (2001-2008)
9209                              <1>       ; 'vgabios.c', 'biosfn_load_gfx_8_14_chars'
9210                              <1>
9211                              <1>       ; biosfn_load_gfx_8_14_chars (BL)
9212                              <1>       ; /* set 0x43 INT pointer */
9213                              <1>       ; write_word(0x0, 0x43*4, &vgafont14);
9214                              <1>       ; write_word(0x0, 0x43*4+2, 0xC000);
9215 00002D88 C705[765B0100]-     <1>       mov   dword [VGA_INT43H], vgafont14
9216 00002D8E [442A0100]          <1>
9217                              <1>
9218                              <1>       ; BL    screen rows code: 00H = user-specified (in DL)
9219                              <1>         ;                       01H = 14 rows
9220                              <1>         ;                       02H = 25 rows
9221                              <1>         ;                       03H = 43 rows
9222                              <1>         ; DL    (when BL=0) custom number of char rows on screen
9223                              <1>
9224                              <1>       ; switch (BL) {
9225                              <1>       ; case 0:
9226                              <1>       ;    write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, DL-1);
9227                              <1>       ;    break;
9228 00002D92 20DB                <1>       and   bl, bl
9229 00002D94 7508                <1>       jnz   short l_gfx_8_14c_1
9230 00002D96 8815[CA5E0000]      <1>       mov   [VGA_ROWS], dl  ; not DL-1 !
9231 00002D9C EB23                <1>       jmp   short l_gfx_8_14c_4
9232                              <1> l_gfx_8_14c_1:
9233                              <1>       ; case 1:
9234                              <1>       ;    write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, 13);
9235                              <1>       ;    break;
9236 00002D9E FECB                <1>       dec   bl
9237 00002DA0 7509                <1>       jnz   short l_gfx_8_14c_2
9238                              <1>       ; bl = 1
9239 00002DA2 C605[CA5E0000]0E    <1>       mov   byte [VGA_ROWS], 14  ; not 13 !
9240 00002DA9 EB16                <1>       jmp   short l_gfx_8_14c_4
9241                              <1> l_gfx_8_14c_2:
9242 00002DAB FECB                <1>       dec   bl
9243 00002DAD 740B                <1>       jz    short l_gfx_8_14c_3 ; bl = 2
9244 00002DAF FECB                <1>       dec   bl
9245 00002DB1 750E                <1>       jnz   short l_gfx_8_14c_4 ; bl > 3
9246                              <1>       ; bl = 3
9247                              <1>       ; case 3:
9248                              <1>       ;    write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, 42);
9249                              <1>       ;    break;
9250 00002DB3 C605[CA5E0000]2B    <1>       mov   byte [VGA_ROWS], 43  ; not 42 !
9251                              <1> l_gfx_8_14c_3:
9252                              <1>       ; case 2:
9253                              <1>       ; default:
9254                              <1>       ;    write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, 24);
9255                              <1>       ;    break;
9256                              <1>       ; bl = 2 or bl > 3
9257 00002DBA C605[CA5E0000]19    <1>       mov   byte [VGA_ROWS], 25  ; not 24 !
9258                              <1>       ; }
9259                              <1> l_gfx_8_14c_4:
9260                              <1>       ; write_byte(BIOSMEM_SEG, BIOSMEM_CHAR_HEIGHT, 14);
9261 00002DC1 C605[C65E0000]0E    <1>         mov     byte [CHAR_HEIGHT], 14
9262                              <1>       ; }
9263 00002DC8 C3                  <1>       retn
9264                              <1>
9265                              <1> load_gfx_8_8_chars:
9266                              <1>       ; 08/08/2016
9267                              <1>       ; 10/07/2016
9268                              <1>       ; Setup ROM 8x14 Font for Graphics Mode (VGA)
9269                              <1>       ;
9270                              <1>       ; derived from 'Plex86/Bochs VGABios' source code
9271                              <1>       ; vgabios-0.7a (2011)
9272                              <1>       ; by the LGPL VGABios developers Team (2001-2008)
9273                              <1>       ; 'vgabios.c', 'biosfn_load_gfx_8_8_dd_chars'
9274                              <1>
9275                              <1>       ; biosfn_load_gfx_8_8_dd_chars (BL)
9276                              <1>       ; /* set 0x43 INT pointer */
9277                              <1>       ; write_word(0x0, 0x43*4, &vgafont8);
9278                              <1>       ; write_word(0x0, 0x43*4+2, 0xC000);
9279 00002DC9 C705[765B0100]-     <1>       mov   dword [VGA_INT43H], vgafont8
9280 00002DCF [44220100]          <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;
00002DD3 20DB                   <1>       and    bl, bl
00002DD5 7508                   <1>       jnz    short l_gfx_8_8c_1
00002DD7 8815[CA5E0000]         <1>       mov    [VGA_ROWS], dl  ; not DL-1 !
00002DDD 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;
00002DDF FECB                   <1>       dec    bl
00002DE1 7509                   <1>       jnz    short l_gfx_8_8c_2
                                <1>       ; bl = 1
00002DE3 C605[CA5E0000]0E       <1>       mov    byte [VGA_ROWS], 14  ; not 13 !
00002DEA EB16                   <1>       jmp    short l_gfx_8_8c_4
                                <1> l_gfx_8_8c_2:
00002DEC FECB                   <1>       dec    bl
00002DEE 740B                   <1>       jz     short l_gfx_8_8c_3 ; bl = 2
00002DF0 FECB                   <1>       dec    bl
00002DF2 750E                   <1>       jnz    short l_gfx_8_8c_4 ; bl > 3
                                <1>       ; bl = 3
                                <1>       ; case 3:
                                <1>       ;    write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, 42);
                                <1>       ;    break;
00002DF4 C605[CA5E0000]2B       <1>       mov    byte [VGA_ROWS], 43  ; not 42 !
                                <1> l_gfx_8_8c_3:
                                <1>       ; case 2:
                                <1>       ; default:
                                <1>       ;    write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, 24);
                                <1>       ;    break;
                                <1>       ; bl = 2 or bl > 3
00002DFB C605[CA5E0000]19       <1>       mov    byte [VGA_ROWS], 25  ; not 24 !
                                <1>       ; }
                                <1> l_gfx_8_8c_4:
                                <1>       ; write_byte(BIOSMEM_SEG, BIOSMEM_CHAR_HEIGHT, 8);
                                <1>         mov     byte [CHAR_HEIGHT], 8
                                <1>       ; }
00002E09 C3                     <1>       retn
                                <1>
                                <1> load_gfx_8_16_chars:
                                <1>       ; 08/08/2016
                                <1>       ; 10/07/2016
                                <1>       ; Setup ROM 8x14 Font for Graphics Mode (VGA)
                                <1>       ;
                                <1>       ; derived from 'Plex86/Bochs VGABios' source code
                                <1>       ; vgabios-0.7a (2011)
                                <1>       ; by the LGPL VGABios developers Team (2001-2008)
                                <1>       ; 'vgabios.c', 'biosfn_load_gfx_8_16_chars'
                                <1>
                                <1>       ; biosfn_load_gfx_8_16_chars (BL)
                                <1>       ; /* set 0x43 INT pointer */
                                <1>       ; write_word(0x0, 0x43*4, &vgafont16);
                                <1>       ; write_word(0x0, 0x43*4+2, 0xC000);
00002E0A C705[765B0100]-        <1>       mov    dword [VGA_INT43H], vgafont16
00002E10 [44380100]             <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;
00002E14 20DB                   <1>       and    bl, bl
00002E16 7508                   <1>       jnz    short l_gfx_8_16c_1
00002E18 8815[CA5E0000]         <1>       mov    [VGA_ROWS], dl  ; not DL-1 !
00002E1E EB23                   <1>       jmp    short l_gfx_8_16c_4
                                <1> l_gfx_8_16c_1:
                                <1>       ; case 1:
                                <1>       ;    write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, 13);
                                <1>       ;    break;
00002E20 FECB                   <1>       dec    bl
00002E22 7509                   <1>       jnz    short l_gfx_8_16c_2
                                <1>       ; bl = 1
00002E24 C605[CA5E0000]0E       <1>       mov    byte [VGA_ROWS], 14  ; not 13 !
00002E2B EB16                   <1>       jmp    short l_gfx_8_16c_4
                                <1> l_gfx_8_16c_2:
00002E2D FECB                   <1>       dec    bl
00002E2F 740B                   <1>       jz     short l_gfx_8_16c_3 ; bl = 2
00002E31 FECB                   <1>       dec    bl
00002E33 750E                   <1>       jnz    short l_gfx_8_16c_4 ; bl > 3
                                <1>       ; bl = 3
                                <1>       ; case 3:
                                <1>       ;    write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, 42);
                                <1>       ;    break;
00002E35 C605[CA5E0000]2B       <1>       mov    byte [VGA_ROWS], 43  ; not 42 !
                                <1> l_gfx_8_16c_3:
                                <1>       ; case 2:
                                <1>       ; default:
                                <1>       ;    write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, 24);
```

```
9383                                    <1>       ;    break;
9384                                    <1>       ; bl = 2 or bl > 3
9385 00002E3C C605[CA5E0000]19          <1>       mov    byte [VGA_ROWS], 25  ; not 24 !
9386                                    <1>       ; }
9387                                    <1> l_gfx_8_16c_4:
9388                                    <1>       ; write_byte(BIOSMEM_SEG, BIOSMEM_CHAR_HEIGHT, 16);
9389 00002E43 C605[C65E0000]10          <1>         mov     byte [CHAR_HEIGHT], 16
9390                                    <1>       ; }
9391 00002E4A C3                        <1>       retn
9392                                    <1>
9393                                    <1> get_font_info:
9394                                    <1>       ; 19/09/2016
9395                                    <1>       ; 08/08/2016
9396                                    <1>       ; 10/07/2016
9397                                    <1>       ; Get Current Character Generator Info (VGA)
9398                                    <1>       ;
9399                                    <1>       ; derived from 'Plex86/Bochs VGABios' source code
9400                                    <1>       ; vgabios-0.7a (2011)
9401                                    <1>       ; by the LGPL VGABios developers Team (2001-2008)
9402                                    <1>       ; 'vgabios.c', 'biosfn_get_font_info'
9403                                    <1>
9404                                    <1>       ; Modified for TRDOS 386 !
9405                                    <1>       ;
9406                                    <1>       ; INPUT ->
9407                                    <1>       ;    AX = 1130h
9408                                    <1>       ;    BL = 0 -> Get info for current VGA font
9409                                    <1>       ;              (BH = unused)
9410                                    <1>       ;    19/09/2016
9411                                    <1>       ;    BL > 0 -> Get requested character font data
9412                                    <1>       ;       BL = 1 -> vgafont8
9413                                    <1>       ;         BL = 2 -> vgafont14
9414                                    <1>       ;       BL = 3 -> vgafont16
9415                                    <1>       ;       BL > 3 -> Invalid function (for now!)
9416                                    <1>       ;       BH = ASCII code of the first character
9417                                    <1>       ;       ECX = Number of characters from the 1st char
9418                                    <1>       ;       ECX >= 256 -> All (256-BH) characters
9419                                    <1>       ;       ECX = 0 -> All characters (BH = unused)
9420                                    <1>       ;         EDX = User's Buffer Address
9421                                    <1>       ; OUTPUT ->
9422                                    <1>       ;    AL = height (scanlines), bytes per character
9423                                    <1>       ;    AH = screen rows
9424                                    <1>       ;    Byte 16-23 of EAX = number of columns
9425                                    <1>       ;    Byte 24-31 of EAX =
9426                                    <1>       ;        0 -> default font (not configured yet)
9427                                    <1>       ;        0FFh -> user defined font
9428                                    <1>       ;        14 = vgafont14
9429                                    <1>       ;         8 = vgafont8
9430                                    <1>       ;        16 = vgafont16
9431                                    <1>       ;   If BL input > 0 ->
9432                                    <1>       ;       EAX = Actual transfer count
9433                                    <1>       ;
9434 00002E4B 20DB                     <1>       and    bl, bl
9435 00002E4D 7408                     <1>       jz     short gfi_0
9436                                    <1>       ; invalid function (input)
9437 00002E4F 80FB03                   <1>       cmp    bl, 3
9438 00002E52 7642                     <1>       jna    short gfi_4
9439 00002E54 31C0                     <1>       xor    eax, eax ; 0
9440 00002E56 C3                       <1>       retn
9441                                    <1> gfi_0:
9442 00002E57 A0[C65E0000]             <1>       mov    al, [CHAR_HEIGHT]
9443 00002E5C 8A25[CA5E0000]           <1>       mov    ah, [VGA_ROWS]
9444 00002E62 C1E010                   <1>       shl    eax, 16
9445 00002E65 A0[C45E0000]             <1>       mov    al, [CRT_COLS]
9446 00002E6A 8B0D[765B0100]           <1>       mov    ecx, [VGA_INT43H]
9447 00002E70 21C9                     <1>       and    ecx, ecx
9448 00002E72 741E                     <1>       jz     short gfi_2 ; 0 = default font
9449 00002E74 41                       <1>       inc    ecx ; 0FFFFFFFFh -> 0 (user defined font)
9450 00002E75 7504                     <1>       jnz    short gfi_1
9451 00002E77 FECC                     <1>       dec    ah ; 0FFh
9452 00002E79 EB17                     <1>       jmp    short gfi_2
9453                                    <1> gfi_1:
9454 00002E7B 49                       <1>       dec    ecx ; 08/08/2016
9455 00002E7C B40E                     <1>       mov    ah, 14
9456 00002E7E 81F9[442A0100]           <1>       cmp    ecx, vgafont14
9457 00002E84 740C                     <1>       je     short gfi_2
9458 00002E86 B408                     <1>       mov    ah, 8
9459 00002E88 81F9[44220100]           <1>       cmp    ecx, vgafont8
9460 00002E8E 7402                     <1>       je     short gfi_2
9461                                    <1>       ; vgafont16
9462 00002E90 D0E4                     <1>       shl    ah, 1 ; ah = 16
9463                                    <1> gfi_2:
9464 00002E92 C1C010                   <1>       rol    eax, 16
9465                                    <1> gfi_3:
9466 00002E95 C3                       <1>       retn
9467                                    <1> gfi_4:
9468 00002E96 89D7                     <1>       mov    edi, edx ; **
9469 00002E98 80FB02                   <1>       cmp    bl, 2
9470 00002E9B 720B                     <1>       jb     short gfi_5
9471 00002E9D 772F                     <1>       ja     short gfi_7
9472                                    <1>       ;BL = 2 -> vgafont14
9473 00002E9F BE[442A0100]             <1>       mov    esi, vgafont14 ; *
9474 00002EA4 B30E                     <1>       mov    bl, 14
9475 00002EA6 EB07                     <1>       jmp    short gfi_6
9476                                    <1> gfi_5:
9477                                    <1>       ;BL = 1 -> vgafont8
9478 00002EA8 BE[44220100]             <1>       mov    esi, vgafont8 ; *
9479 00002EAD B308                     <1>       mov    bl, 8
9480                                    <1> gfi_6:
9481 00002EAF 09C9                     <1>       or     ecx, ecx
9482 00002EB1 7424                     <1>       jz     short gfi_8 ; all chars from the 00h
9483 00002EB3 88F8                     <1>       mov    al, bh ; character index
9484 00002EB5 F6E3                     <1>       mul    bl ; char index * char height/size
```

```
9485 00002EB7 0FB7D0              <1>       movzx  edx, ax
9486 00002EBA 01D6                <1>       add    esi, edx ; *
9487 00002EBC 66BAFF00            <1>       mov    dx, 255
9488 00002EC0 28FA                <1>       sub    dl, bh
9489 00002EC2 6642                <1>       inc    dx
9490 00002EC4 39D1                <1>       cmp    ecx, edx
9491 00002EC6 770F                <1>       ja     short gfi_8
9492 00002EC8 7412                <1>       je     short gfi_9
9493 00002ECA 89D1                <1>       mov    ecx, edx
9494 00002ECC EB0E                <1>       jmp    short gfi_9
9495                              <1> gfi_7:
9496                              <1>       ;BL = 3 -> vgafont16
9497 00002ECE BE[44380100]        <1>       mov    esi, vgafont16 ; *
9498 00002ED3 B310                <1>       mov    bl, 16
9499 00002ED5 EBD8                <1>       jmp    short gfi_6
9500                              <1> gfi_8:
9501 00002ED7 B900010000          <1>       mov    ecx, 256
9502                              <1> gfi_9:
9503 00002EDC 6689C8              <1>       mov    ax, cx ; character count
9504 00002EDF 30FF                <1>       xor    bh, bh
9505 00002EE1 66F7E3              <1>       mul    bx ; char count * char height/size
9506 00002EE4 6689C1              <1>       mov    cx, ax
9507                              <1>
9508                              <1>       ; ESI = source address in system space
9509                              <1>       ; EDI = user's buffer address
9510                              <1>       ; ECX = transfer (byte) count
9511 00002EE7 E896B90000          <1>       call   transfer_to_user_buffer
9512 00002EEC 89C8                <1>       mov    eax, ecx ; actual transfer count
9513 00002EEE C3                  <1>       retn
9514                              <1>
9515                              <1> vga_pal_funcs:
9516                              <1>       ; 10/08/2016
9517                              <1>       ; VGA Palette functions
9518                              <1>       ;
9519                              <1>       ; derived from 'Plex86/Bochs VGABios' source code
9520                              <1>       ; vgabios-0.7a (2011)
9521                              <1>       ; by the LGPL VGABios developers Team (2001-2008)
9522                              <1>       ; 'vgabios.c', 'vgarom.asm'
9523                              <1>
9524 00002EEF 3C00                <1>       cmp    al, 0
9525 00002EF1 0F848F000000        <1>         je      set_single_palette_reg
9526                              <1> vga_palf_1001:
9527 00002EF7 3C01                <1>       cmp    al, 1
9528 00002EF9 0F84B4000000        <1>         je      set_overscan_border_color
9529                              <1> vga_palf_1002:
9530 00002EFF 3C02                <1>       cmp    al, 2
9531 00002F01 0F84B0000000        <1>         je      set_all_palette_reg
9532                              <1> vga_palf_1003:
9533 00002F07 3C03                <1>       cmp    al, 3
9534 00002F09 0F84E8000000        <1>         je      toggle_intensity
9535                              <1> vga_palf_1007:
9536 00002F0F 3C07                <1>       cmp    al, 7
9537 00002F11 0F840D010000        <1>         je      get_single_palette_reg
9538 00002F17 7266                <1>       jb     short vga_palf_unknown
9539                              <1> vga_palf_1008:
9540 00002F19 3C08                <1>       cmp    al, 8
9541 00002F1B 0F8437010000        <1>         je      read_overscan_border_color
9542                              <1> vga_palf_1009:
9543 00002F21 3C09                <1>       cmp    al, 9
9544 00002F23 0F8433010000        <1>       je     get_all_palette_reg
9545                              <1> vga_palf_1010:
9546 00002F29 3C10                <1>       cmp    al, 10h
9547 00002F2B 0F8487010000        <1>       je     set_single_dac_reg
9548 00002F31 724C                <1>       jb     short vga_palf_unknown
9549                              <1> vga_palf_1012:
9550 00002F33 3C12                <1>       cmp    al, 12h
9551 00002F35 0F8498010000        <1>         je      set_all_dac_reg
9552 00002F3B 7242                <1>       jb     short vga_palf_unknown
9553                              <1> vga_palf_1013:
9554 00002F3D 3C13                <1>       cmp    al, 13h
9555 00002F3F 0F84CC010000        <1>         je      select_video_dac_color_page
9556                              <1> vga_palf_1015:
9557 00002F45 3C15                <1>       cmp    al, 15h
9558 00002F47 0F8412020000        <1>       je     read_single_dac_reg
9559 00002F4D 7230                <1>       jb     short vga_palf_unknown
9560                              <1> vga_palf_1017:
9561 00002F4F 3C17                <1>       cmp    al, 17h
9562 00002F51 0F8428020000        <1>         je      read_all_dac_reg
9563 00002F57 7226                <1>       jb     short vga_palf_unknown
9564                              <1> vga_palf_1018:
9565 00002F59 3C18                <1>       cmp    al, 18h
9566 00002F5B 0F845E020000        <1>         je      set_pel_mask
9567                              <1> vga_palf_1019:
9568 00002F61 3C19                <1>       cmp    al, 19h
9569 00002F63 0F8462020000        <1>         je      read_pel_mask
9570                              <1> vga_palf_101A:
9571 00002F69 3C1A                <1>       cmp    al, 1Ah
9572 00002F6B 0F8468020000        <1>         je      read_video_dac_state
9573                              <1> vga_palf_101B:
9574 00002F71 3C1B                <1>       cmp    al, 1Bh
9575                              <1>       ;jne   short vga_palf_unknown
9576 00002F73 770A                <1>       ja     short vga_palf_unknown
9577                              <1>
9578 00002F75 E80CF7FFFF          <1>       call   gray_scale_summing
9579 00002F7A E9D5E5FFFF          <1>         jmp     VIDEO_RETURN
9580                              <1>
9581                              <1> vga_palf_unknown:
9582 00002F7F 29C0                <1>       sub    eax, eax ; 0 = invalid function
9583 00002F81 E9D3E5FFFF          <1>         jmp     _video_return
9584                              <1>
9585                              <1> set_single_palette_reg:
9586                              <1>       ; 10/08/2016
```

```
                                <1>       ; Set One Palette Register
                                <1>       ; BL = register number to set
                                <1>       ;     (a 4-bit attribute nibble: 00h-0Fh)
                                <1>       ; BH = 6-bit RGB color to display
                                <1>       ;      for that attribute
                                <1>
00002F86 80FB14                 <1>       cmp    bl, 14h
                                <1>       ;ja    short no_actl_reg1
00002F89 0F87C5E5FFFF           <1>       ja     VIDEO_RETURN
00002F8F 6650                   <1>       push   ax
00002F91 6652                   <1>       push   dx
00002F93 66BADA03               <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
00002F97 EC                     <1>       in     al, dx
00002F98 66BAC003               <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
00002F9C 88D8                   <1>       mov    al, bl
00002F9E EE                     <1>       out    dx, al
00002F9F 88F8                   <1>       mov    al, bh
00002FA1 EE                     <1>       out    dx, al
00002FA2 B020                   <1>       mov    al, 20h
00002FA4 EE                     <1>       out    dx, al
                                <1>       ; ifdef VBOX
00002FA5 66BADA03               <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
00002FA9 EC                     <1>       in     al, dx
                                <1>       ; endif ; VBOX
00002FAA 665A                   <1>       pop    dx
00002FAC 6658                   <1>       pop    ax
                                <1> ;no_actl_reg1:
00002FAE E9A1E5FFFF             <1>       jmp    VIDEO_RETURN
                                <1>
                                <1> set_overscan_border_color:
                                <1>       ; 10/08/2016
                                <1>       ; Set Overscan/Border Color Register
                                <1>       ; BH = 6-bit RGB color to display
                                <1>       ;      for that attribute
                                <1>
00002FB3 B311                   <1>       mov    bl, 11h
00002FB5 EBCF                   <1>       jmp    short set_single_palette_reg
                                <1>
                                <1> set_all_palette_reg:
                                <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>
00002FB7 89D6                   <1>       mov    esi, edx ; user buffer
00002FB9 B911000000             <1>       mov    ecx, 17
00002FBE 89E7                   <1>       mov    edi, esp
00002FC0 83EC14                 <1>       sub    esp, 20
00002FC3 E804B90000             <1>       call   transfer_from_user_buffer
                                <1>       ;jc    VIDEO_RETURN
                                <1>
00002FC8 66BADA03               <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
00002FCC EC                     <1>       in     al, dx
00002FCD B100                   <1>       mov    cl, 0
00002FCF 66BAC003               <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
                                <1> set_palette_loop:
00002FD3 88C8                   <1>       mov    al, cl
00002FD5 EE                     <1>       out    dx, al
00002FD6 8A07                   <1>       mov    al, [edi]
00002FD8 EE                     <1>       out    dx, al
00002FD9 47                     <1>       inc    edi
00002FDA FEC1                   <1>       inc    cl
00002FDC 80F910                 <1>       cmp    cl, 10h
00002FDF 75F2                   <1>       jne    short set_palette_loop
00002FE1 B011                   <1>       mov    al, 11h
00002FE3 EE                     <1>       out    dx, al
00002FE4 8A07                   <1>       mov    al, [edi]
00002FE6 EE                     <1>       out    dx, al
00002FE7 B020                   <1>       mov    al, 20h
00002FE9 EE                     <1>       out    dx, al
                                <1>       ; ifdef VBOX
00002FEA 66BADA03               <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
00002FEE EC                     <1>       in     al, dx
                                <1>       ; endif ; VBOX
00002FEF 83C414                 <1>       add    esp, 20
00002FF2 E95DE5FFFF             <1>       jmp    VIDEO_RETURN
                                <1>
                                <1> toggle_intensity:
                                <1>       ; 10/08/2016
                                <1>       ; Select Foreground Blink or Bold Background
                                <1>       ; BL = 00h = enable bold backgrounds
                                <1>       ;          (16 background colors)
                                <1>         ;      01h = enable blinking foreground
                                <1>       ;          (8 background colors)
                                <1>
00002FF7 66BADA03               <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
00002FFB EC                     <1>       in     al, dx
00002FFC 66BAC003               <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
00003000 B010                   <1>       mov    al, 10h
00003002 EE                     <1>       out    dx, al
00003003 66BAC103               <1>       mov    dx, 3C1h ; VGAREG_ACTL_READ_DATA
00003007 EC                     <1>       in     al, dx
00003008 24F7                   <1>       and    al, 0F7h
0000300A 80E301                 <1>       and    bl, 01h
0000300D C0E303                 <1>       shl    bl, 3
00003010 08D8                   <1>       or     al, bl
00003012 66BAC003               <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
00003016 EE                     <1>       out    dx, al
00003017 B020                   <1>       mov    al, 20h
00003019 EE                     <1>       out    dx, al
                                <1>       ; ifdef VBOX
```

```
9689 0000301A 66BADA03            <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
9690 0000301E EC                  <1>       in     al, dx
9691                              <1>       ; endif ; VBOX
9692 0000301F E930E5FFFF          <1>       jmp    VIDEO_RETURN
9693                              <1>
9694                              <1> get_single_palette_reg:
9695                              <1>       ; 10/08/2016
9696                              <1>       ; Read One Palette Register
9697                              <1>         ; INPUT:
9698                              <1>       ; BL = Palette register to read (00h-0Fh)
9699                              <1>       ; OUTPUT:
9700                              <1>       ; BH = Current rgbRGB value of specified register
9701                              <1>       ;      for that attribute
9702                              <1>
9703 00003024 80FB14              <1>       cmp    bl, 14h
9704                              <1>       ;ja    short no_actl_reg2
9705 00003027 0F8727E5FFFF        <1>       ja     VIDEO_RETURN
9706                              <1>
9707 0000302D 66BADA03            <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
9708 00003031 EC                  <1>       in     al, dx
9709 00003032 66BAC003            <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
9710 00003036 88D8                <1>       mov    al, bl
9711 00003038 EE                  <1>       out    dx, al
9712 00003039 66BAC103            <1>       mov    dx, 3C1h ; VGAREG_ACTL_READ_DATA
9713 0000303D EC                  <1>       in     al, dx
9714 0000303E 8844240D            <1>       mov    [esp+13], al ; bh
9715 00003042 66BADA03            <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
9716 00003046 EC                  <1>       in     al, dx
9717 00003047 66BAC003            <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
9718 0000304B B020                <1>       mov    al, 20h
9719 0000304D EE                  <1>       out    dx, al
9720                              <1>       ; ifdef VBOX
9721 0000304E 66BADA03            <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
9722 00003052 EC                  <1>       in     al, dx
9723                              <1>       ; endif ; VBOX
9724 00003053 E9FCE4FFFF          <1>       jmp    VIDEO_RETURN
9725                              <1>
9726                              <1> read_overscan_border_color:
9727                              <1>       ; 10/08/2016
9728                              <1>       ; Read Overscan Register
9729                              <1>       ; OUTPUT:
9730                              <1>       ; BH = current rgbRGB value
9731                              <1>       ;      of the overscan/border register
9732                              <1>
9733 00003058 B311                <1>       mov    bl, 11h
9734 0000305A EBC8                <1>       jmp    short get_single_palette_reg
9735                              <1>
9736                              <1> get_all_palette_reg:
9737                              <1>       ; 10/08/2016
9738                              <1>       ; Read All Palette Registers
9739                              <1>       ; EDX = Address of 17-byte buffer
9740                              <1>       ;      to receive data
9741                              <1>
9742 0000305C 89D7                <1>       mov    edi, edx
9743 0000305E 89E3                <1>       mov    ebx, esp
9744 00003060 89DE                <1>       mov    esi, ebx
9745 00003062 83EC14              <1>       sub    esp, 20
9746                              <1>
9747 00003065 B100                <1>       mov    cl, 0
9748                              <1> get_palette_loop:
9749 00003067 66BADA03            <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
9750 0000306B EC                  <1>       in     al, dx
9751 0000306C 66BAC003            <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
9752 00003070 88C8                <1>       mov    al, cl
9753 00003072 EE                  <1>       out    dx, al
9754 00003073 66BAC103            <1>       mov    dx, 3C1h ; VGAREG_ACTL_READ_DATA
9755 00003077 EC                  <1>       in     al, dx
9756 00003078 8803                <1>       mov    [ebx], al
9757 0000307A 43                  <1>       inc    ebx
9758 0000307B FEC1                <1>       inc    cl
9759 0000307D 80F910              <1>       cmp    cl, 10h
9760 00003080 75E5                <1>       jne    short get_palette_loop
9761 00003082 66BADA03            <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
9762 00003086 EC                  <1>       in     al, dx
9763 00003087 66BAC003            <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
9764 0000308B B011                <1>       mov    al, 11h
9765 0000308D EE                  <1>       out    dx, al
9766 0000308E 66BAC103            <1>       mov    dx, 3C1h ; VGAREG_ACTL_READ_DATA
9767 00003092 EC                  <1>       in     al, dx
9768 00003093 8803                <1>       mov    [ebx], al
9769 00003095 66BADA03            <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
9770 00003099 EC                  <1>       in     al, dx
9771 0000309A 66BAC003            <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
9772 0000309E B020                <1>       mov    al, 20h
9773 000030A0 EE                  <1>       out    dx, al
9774                              <1>       ; ifdef VBOX
9775 000030A1 66BADA03            <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
9776 000030A5 EC                  <1>       in     al, dx
9777                              <1>       ; endif ; VBOX
9778                              <1>
9779 000030A6 B911000000          <1>       mov    ecx, 17 ; transfer (byte) count
9780                              <1>       ; ESI = source address in system space
9781                              <1>       ; EDI = user's buffer address
9782 000030AB E8D2B70000          <1>       call   transfer_to_user_buffer
9783                              <1>
9784 000030B0 83C414              <1>       add    esp, 20
9785 000030B3 E99CE4FFFF          <1>       jmp    VIDEO_RETURN
9786                              <1>
9787                              <1> set_single_dac_reg:
9788                              <1>       ; 10/08/2016
9789                              <1>       ; Set One DAC Color Register
9790                              <1>       ; BX = color register to set (0-255)
```

```
                                <1>         ; CH = green value (00h-3Fh)
                                <1>         ; CL = blue value  (00h-3Fh)
                                <1>         ; DH = red value   (00h-3Fh)
                                <1>
000030B8 6652                   <1>       push   dx
000030BA 66BAC803               <1>       mov    dx, 3C8h ; VGAREG_DAC_WRITE_ADDRESS
000030BE 88D8                   <1>       mov    al, bl
000030C0 EE                     <1>       out    dx, al
                                <1>       ;mov   dx, 3C9h ; VGAREG_DAC_DATA
000030C1 6642                   <1>       inc    dx
000030C3 6658                   <1>       pop    ax
000030C5 88E0                   <1>       mov    al, ah
000030C7 EE                     <1>       out    dx, al
000030C8 88E8                   <1>       mov    al, ch
000030CA EE                     <1>       out    dx, al
000030CB 88C8                   <1>       mov    al, cl
000030CD EE                     <1>       out    dx, al
000030CE E981E4FFFF             <1>       jmp    VIDEO_RETURN
                                <1>
                                <1> set_all_dac_reg:
                                <1>       ; 12/08/2016
                                <1>       ; 11/08/2016
                                <1>       ; 10/08/2016
                                <1>       ; Set a Block of DAC Color Register
                                <1>       ; BX = first DAC register to set (0-00FFh)
                                <1>       ; ECX = number of registers to set (0-00FFh)
                                <1>       ; EDX = addr of a table of R,G,B values
                                <1>       ;     (it will be CX*3 bytes long)
                                <1>
000030D3 89D6                   <1>       mov    esi, edx ; user buffer
000030D5 89CA                   <1>       mov    edx, ecx
000030D7 66D1E1                 <1>       shl    cx, 1 ; *2
000030DA 01D1                   <1>       add    ecx, edx ; ecx = 3*ecx
000030DC 89E5                   <1>       mov    ebp, esp
000030DE 89EF                   <1>       mov    edi, ebp
000030E0 29CF                   <1>       sub    edi, ecx
000030E2 6683E7FC               <1>       and    di, 0FFFCh ; (dword alignment)
000030E6 89FC                   <1>       mov    esp, edi
000030E8 E8DFB70000             <1>       call   transfer_from_user_buffer
                                <1>       ;jc    VIDEO_RETURN
                                <1>
000030ED 89D1                   <1>       mov    ecx, edx
000030EF 66BAC803               <1>       mov    dx, 3C8h ; VGAREG_DAC_WRITE_ADDRESS
000030F3 88D8                   <1>       mov    al, bl
000030F5 EE                     <1>       out    dx, al
000030F6 66BAC903               <1>       mov    dx, 3C9h ; VGAREG_DAC_DATA
                                <1> set_dac_loop:
000030FA 8A07                   <1>       mov    al, [edi]
000030FC EE                     <1>       out    dx, al
000030FD 47                     <1>       inc    edi
000030FE 8A07                   <1>       mov    al, [edi]
00003100 EE                     <1>       out    dx, al
00003101 47                     <1>       inc    edi
00003102 8A07                   <1>       mov    al, [edi]
00003104 EE                     <1>       out    dx, al
00003105 47                     <1>       inc    edi
00003106 6649                   <1>       dec    cx
00003108 75F0                   <1>       jnz    short set_dac_loop
0000310A 89EC                   <1>       mov    esp, ebp
0000310C E943E4FFFF             <1>       jmp    VIDEO_RETURN
                                <1>
                                <1> select_video_dac_color_page:
                                <1>       ; 10/08/2016
                                <1>       ; DAC Color Paging Functions
                                <1>       ; BL = 00H = select color paging mode
                                <1>         ;       BH = paging mode
                                <1>         ;            00h = 4 blocks of 64 registers
                                <1>         ;            01h = 16 blocks of 16 registers
                                <1>       ; BL = 01H = activate color page
                                <1>         ;       BH = DAC color page number
                                <1>         ;            00h-03h (4-page/64-reg mode)
                                <1>         ;            00h-0Fh (16-page/16-reg mode)
                                <1>
00003111 66BADA03               <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
00003115 EC                     <1>       in     al, dx
00003116 66BAC003               <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
0000311A B010                   <1>       mov    al, 10h
0000311C EE                     <1>       out    dx, al
0000311D 66BAC103               <1>       mov    dx, 3C1h ; VGAREG_ACTL_READ_DATA
00003121 EC                     <1>       in     al, dx
00003122 80E301                 <1>       and    bl, 01h
00003125 750E                   <1>       jnz    short set_dac_page
00003127 247F                   <1>       and    al, 07Fh
00003129 C0E707                 <1>       shl    bh, 7
0000312C 08F8                   <1>       or     al, bh
0000312E 66BAC003               <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
00003132 EE                     <1>       out    dx, al
00003133 EB1D                   <1>       jmp    short set_actl_normal
                                <1> set_dac_page:
00003135 6650                   <1>       push   ax
00003137 66BADA03               <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
0000313B EC                     <1>       in     al, dx
0000313C 66BAC003               <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
00003140 B014                   <1>       mov    al, 14h
00003142 EE                     <1>       out    dx, al
00003143 6658                   <1>       pop    ax
00003145 2480                   <1>       and    al, 80h
00003147 7503                   <1>       jnz    short set_dac_16_page
00003149 C0E702                 <1>       shl    bh, 2
                                <1> set_dac_16_page:
0000314C 80E70F                 <1>       and    bh, 0Fh
0000314F 88F8                   <1>       mov    al, bh
```

```
9893 00003151 EE                <1>       out    dx, al
9894                            <1> set_actl_normal:
9895 00003152 B020              <1>       mov    al, 20h
9896 00003154 EE                <1>       out    dx, al
9897                            <1>       ; ifdef VBOX
9898 00003155 66BADA03          <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
9899 00003159 EC                <1>       in     al, dx
9900                            <1>       ; endif ; VBOX
9901 0000315A E9F5E3FFFF        <1>       jmp    VIDEO_RETURN
9902                            <1>
9903                            <1> read_single_dac_reg:
9904                            <1>       ; 10/08/2016
9905                            <1>       ; Read One DAC Color Register
9906                            <1>       ; INPUT:
9907                            <1>       ; BX = color register to read (0-255)
9908                            <1>       ; OUTPUT:
9909                            <1>       ; CH = green value (00h-3Fh)
9910                            <1>         ; CL = blue value  (00h-3Fh)
9911                            <1>         ; DH = red value   (00h-3Fh)
9912                            <1>
9913 0000315F 66BAC703          <1>       mov    dx, 3C7h ; VGAREG_DAC_READ_ADDRESS
9914 00003163 88D8              <1>       mov    al, bl
9915 00003165 EE                <1>       out    dx, al
9916 00003166 66BAC903          <1>       mov    dx, 3C9h ; VGAREG_DAC_DATA
9917 0000316A EC                <1>       in     al, dx
9918 0000316B 88442415          <1>       mov    [esp+21], al ; dh
9919 0000316F EC                <1>       in     al, dx
9920 00003170 88C5              <1>       mov    ch, al
9921 00003172 EC                <1>       in     al, dx
9922 00003173 88C1              <1>       mov    cl, al
9923 00003175 66894C2410        <1>       mov    [esp+16], cx ; cx
9924 0000317A E9D5E3FFFF        <1>       jmp    VIDEO_RETURN
9925                            <1>
9926                            <1> read_all_dac_reg:
9927                            <1>       ; 12/08/2016
9928                            <1>       ; 11/08/2016
9929                            <1>       ; 10/08/2016
9930                            <1>       ; Read a Block of DAC Color Registers
9931                            <1>         ; BX = first DAC register to read (0-00FFh)
9932                            <1>         ; ECX = number of registers to read (0-00FFh)
9933                            <1>         ; EDX = addr of a buffer to hold R,G,B values
9934                            <1>       ;     (CX*3 bytes long)
9935                            <1>
9936 0000317F 89D7              <1>       mov    edi, edx ; user buffer
9937 00003181 89CA              <1>       mov    edx, ecx
9938 00003183 66D1E2            <1>       shl    dx, 1 ; *2
9939 00003186 01CA              <1>       add    edx, ecx ; edx = 3*ecx
9940 00003188 89E5              <1>       mov    ebp, esp
9941 0000318A 89EE              <1>       mov    esi, ebp
9942 0000318C 29D6              <1>       sub    esi, edx
9943 0000318E 6683E6FC          <1>       and    si, 0FFFCh ; (dword alignment)
9944 00003192 89F4              <1>       mov    esp, esi
9945 00003194 52                <1>       push   edx ; 3*ecx
9946 00003195 66BAC703          <1>       mov    dx, 3C7h ; VGAREG_DAC_READ_ADDRESS
9947 00003199 88D8              <1>       mov    al, bl
9948 0000319B EE                <1>       out    dx, al
9949 0000319C 66BAC903          <1>       mov    dx, 3C9h ; VGAREG_DAC_DATA
9950 000031A0 89F3              <1>       mov    ebx, esi
9951                            <1> read_dac_loop:
9952 000031A2 EC                <1>       in     al, dx
9953 000031A3 8803              <1>       mov    [ebx], al
9954 000031A5 43                <1>       inc    ebx
9955 000031A6 EC                <1>       in     al, dx
9956 000031A7 8803              <1>       mov    [ebx], al
9957 000031A9 43                <1>       inc    ebx
9958 000031AA EC                <1>       in     al, dx
9959 000031AB 8803              <1>       mov    [ebx], al
9960 000031AD 43                <1>       inc    ebx
9961 000031AE 6649              <1>       dec    cx
9962 000031B0 75F0              <1>       jnz    short read_dac_loop
9963 000031B2 59                <1>       pop    ecx ; 3*ecx
9964                            <1>       ; ECX = transfer (byte) count
9965                            <1>       ; ESI = source address in system space
9966                            <1>       ; EDI = user's buffer address
9967 000031B3 E8CAB60000        <1>       call   transfer_to_user_buffer
9968 000031B8 89EC              <1>       mov    esp, ebp
9969 000031BA E995E3FFFF        <1>       jmp    VIDEO_RETURN
9970                            <1>
9971                            <1> set_pel_mask:
9972                            <1>       ; 10/08/2016
9973                            <1>       ; BL = mask value
9974 000031BF 66BAC603          <1>       mov    dx, 3C6h ; VGAREG_PEL_MASK
9975 000031C3 88D8              <1>       mov    al, bl
9976 000031C5 EE                <1>       out    dx, al
9977 000031C6 E989E3FFFF        <1>       jmp    VIDEO_RETURN
9978                            <1>
9979                            <1> read_pel_mask:
9980                            <1>       ; 10/08/2016
9981                            <1>       ; Output: BL = mask value
9982 000031CB 66BAC603          <1>       mov    dx, 3C6h ; VGAREG_PEL_MASK
9983 000031CF EC                <1>       in     al, dx
9984 000031D0 8844240C          <1>       mov    [esp+12], al ; bl
9985 000031D4 E97BE3FFFF        <1>       jmp    VIDEO_RETURN
9986                            <1>
9987                            <1> read_video_dac_state:
9988                            <1>       ; 10/08/2016
9989                            <1>       ; Query DAC Color Paging State
9990                            <1>       ; Output:
9991                            <1>       ; BH = current active DAC color page
9992                            <1>         ; BL = current active DAC paging mode
9993                            <1>
9994 000031D9 66BADA03          <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
```

```
 9995 000031DD EC                <1>       in    al, dx
 9996 000031DE 66BAC003          <1>       mov   dx, 3C0h ; VGAREG_ACTL_ADDRESS
 9997 000031E2 B010              <1>       mov   al, 10h
 9998 000031E4 EE                <1>       out   dx, al
 9999 000031E5 66BAC103          <1>       mov   dx, 3C1h ; VGAREG_ACTL_READ_DATA
10000 000031E9 EC                <1>       in    al, dx
10001 000031EA 88C3              <1>       mov   bl, al
10002 000031EC C0EB07            <1>       shr   bl, 7
10003 000031EF 66BADA03          <1>       mov   dx, 3DAh ; VGAREG_ACTL_RESET
10004 000031F3 EC                <1>       in    al, dx
10005 000031F4 66BAC003          <1>       mov   dx, 3C0h ; VGAREG_ACTL_ADDRESS
10006 000031F8 B014              <1>       mov   al, 14h
10007 000031FA EE                <1>       out   dx, al
10008 000031FB 66BAC103          <1>       mov   dx, 3C1h ; VGAREG_ACTL_READ_DATA
10009 000031FF EC                <1>       in    al, dx
10010 00003200 88C7              <1>       mov   bh, al
10011 00003202 80E70F            <1>       and   bh, 0Fh
10012 00003205 F6C301            <1>       test  bl, 01
10013 00003208 7503              <1>       jnz   short get_dac_16_page
10014 0000320A C0EF02            <1>       shr   bh, 2
10015                            <1> get_dac_16_page:
10016 0000320D 66BADA03          <1>       mov   dx, 3DAh ; VGAREG_ACTL_RESET
10017 00003211 EC                <1>       in    al, dx
10018 00003212 66BAC003          <1>       mov   dx, 3C0h ; VGAREG_ACTL_ADDRESS
10019 00003216 B020              <1>       mov   al, 20h
10020 00003218 EE                <1>       out   dx, al
10021                            <1>       ; ifdef VBOX
10022 00003219 66BADA03          <1>       mov   dx, 3DAh ; VGAREG_ACTL_RESET
10023 0000321D EC                <1>       in    al, dx
10024                            <1>       ; endif ; VBOX
10025 0000321E 66895C240C        <1>       mov   [esp+12], bx ; bx
10026 00003223 E92CE3FFFF        <1>       jmp   VIDEO_RETURN
10027                            <1>
10028                            <1> ; % include 'vidata.s' ; VIDEO DATA
10029                            <1>
10030                            <1> ; /// End Of VIDEO FUNCTIONS ///
10031
10032                                setup_rtc_int:
10033                                ; source: http://wiki.osdev.org/RTC
10034 00003228 FA                        cli          ; disable interrupts
10035                                     ; default int frequency is 1024 Hz (Lower 4 bits of register A is 0110b or 6)
10036                                     ; in order to change this ...
10037                                     ; frequency  = 32768 >> (rate-1) --> 32768 >> 5 = 1024
10038                                     ; (rate must be above 2 and not over 15)
10039                                     ; new rate = 15 --> 32768 >> (15-1) = 2 Hz
10040 00003229 B08A                     mov   al, 8Ah
10041 0000322B E670                     out   70h, al ; set index to register A, disable NMI
10042 0000322D 90                       nop
10043 0000322E E471                     in    al, 71h ; get initial value of register A
10044 00003230 88C4                     mov   ah, al
10045 00003232 80E4F0                   and   ah, 0F0h
10046 00003235 B08A                     mov   al, 8Ah
10047 00003237 E670                     out   70h, al ; reset index to register A
10048 00003239 88E0                     mov   al, ah
10049 0000323B 0C0F                     or    al, 0Fh      ; new rate (0Fh -> 15)
10050 0000323D E671                     out   71h, al ; write only our rate to A. Note, rate is the bottom 4 bits.
10051                                   ; enable RTC interrupt
10052 0000323F B08B                     mov   al, 8Bh ;
10053 00003241 E670                     out   70h, al ; select register B and disable NMI
10054 00003243 90                       nop
10055 00003244 E471                     in    al, 71h ; read the current value of register B
10056 00003246 88C4                     mov   ah, al  ;
10057 00003248 B08B                     mov   al, 8Bh ;
10058 0000324A E670                     out   70h, al ; set the index again (a read will reset the index to register B)
10059 0000324C 88E0                     mov   al, ah  ;
10060 0000324E 0C40                     or    al, 40h ;
10061 00003250 E671                     out   71h, al ; write the previous value ORed with 0x40. This turns on bit 6 of register B
10062 00003252 FB                       sti
10063 00003253 C3                       retn
10064
10065                                ; Write memory information
10066                                ; 29/01/2016
10067                                ; 06/11/2014
10068                                ; 14/08/2015
10069                                memory_info:
10070 00003254 A1[CC4D0100]                mov   eax, [memory_size] ; in pages
10071 00003259 50                          push  eax
10072 0000325A C1E00C                      shl   eax, 12               ; in bytes
10073 0000325D BB0A000000                  mov   ebx, 10
10074 00003262 89D9                        mov   ecx, ebx        ; 10
10075 00003264 BE[5D0E0100]                mov   esi, mem_total_b_str
10076 00003269 E8BD000000                  call  bintdstr
10077 0000326E 58                          pop   eax
10078 0000326F B107                        mov   cl, 7
10079 00003271 BE[810E0100]                mov   esi, mem_total_p_str
10080 00003276 E8B0000000                  call  bintdstr
10081                                      ; 14/08/2015
10082 0000327B E8C8000000                  call  calc_free_mem
10083                                      ; edx = calculated free pages
10084                                      ; ecx = 0
10085 00003280 A1[D04D0100]                mov   eax, [free_pages]
10086 00003285 39D0                        cmp   eax, edx ; calculated free mem value
10087                                            ; and initial free mem value are same or not?
10088 00003287 751D                        jne   short pmim ; print mem info with '?' if not
10089 00003289 52                          push  edx ; free memory in pages
10090                                      ;mov  eax, edx
10091 0000328A C1E00C                      shl   eax, 12 ; convert page count
10092                                                  ; to byte count
10093 0000328D B10A                        mov   cl, 10
10094 0000328F BE[A10E0100]                mov   esi, free_mem_b_str
10095 00003294 E892000000                  call  bintdstr
10096 00003299 58                          pop   eax
```

```
10097 0000329A B107                     mov   cl, 7
10098 0000329C BE[C50E0100]             mov   esi, free_mem_p_str
10099 000032A1 E885000000               call  bintdstr
10100                                  pmim:
10101 000032A6 BE[4B0E0100]             mov   esi, msg_memory_info
10102                                   ;
10103 000032AB B407                     mov   ah, 07h ; Black background,
10104                                                 ; light gray forecolor
10105                                  print_kmsg: ; 29/01/2016
10106 000032AD 8825[F74D0100]           mov   [ccolor], ah
10107                                  pkmsg_loop:
10108 000032B3 AC                       lodsb
10109 000032B4 08C0                     or    al, al
10110 000032B6 7410                     jz    short pkmsg_ok
10111 000032B8 56                       push  esi
10112                                   ; 13/05/2016
10113 000032B9 0FB61D[F74D0100]         movzx ebx, byte [ccolor]
10114                                                 ; Video page 0 (bh=0)
10115 000032C0 E8EDE9FFFF               call  _write_tty
10116 000032C5 5E                       pop   esi
10117 000032C6 EBEB                     jmp   short pkmsg_loop
10118                                  pkmsg_ok:
10119 000032C8 C3                       retn
10120
10121                                  ; Convert binary number to hexadecimal string
10122                                  ; 10/05/2015
10123                                  ; dsectpm.s (28/02/2015)
10124                                  ; Retro UNIX 386 v1 - Kernel v0.2.0.6
10125                                  ; 01/12/2014
10126                                  ; 25/11/2014
10127                                  ;
10128                                  bytetohex:
10129                                   ; INPUT ->
10130                                   ;      AL = byte (binary number)
10131                                   ; OUTPUT ->
10132                                   ;      AX = hexadecimal string
10133                                   ;
10134 000032C9 53                       push  ebx
10135 000032CA 31DB                     xor   ebx, ebx
10136 000032CC 88C3                     mov   bl, al
10137 000032CE C0EB04                   shr   bl, 4
10138 000032D1 8A9B[1B330000]           mov   bl, [ebx+hexchrs]
10139 000032D7 86D8                     xchg  bl, al
10140 000032D9 80E30F                   and   bl, 0Fh
10141 000032DC 8AA3[1B330000]           mov   ah, [ebx+hexchrs]
10142 000032E2 5B                       pop   ebx
10143 000032E3 C3                       retn
10144
10145                                  wordtohex:
10146                                   ; INPUT ->
10147                                   ;      AX = word (binary number)
10148                                   ; OUTPUT ->
10149                                   ;      EAX = hexadecimal string
10150                                   ;
10151 000032E4 53                       push  ebx
10152 000032E5 31DB                     xor   ebx, ebx
10153 000032E7 86E0                     xchg  ah, al
10154 000032E9 6650                     push  ax
10155 000032EB 88E3                     mov   bl, ah
10156 000032ED C0EB04                   shr   bl, 4
10157 000032F0 8A83[1B330000]           mov   al, [ebx+hexchrs]
10158 000032F6 88E3                     mov   bl, ah
10159 000032F8 80E30F                   and   bl, 0Fh
10160 000032FB 8AA3[1B330000]           mov   ah, [ebx+hexchrs]
10161 00003301 C1E010                   shl   eax, 16
10162 00003304 6658                     pop   ax
10163 00003306 5B                       pop   ebx
10164 00003307 EBC0                     jmp   short bytetohex
10165                                   ;mov  bl, al
10166                                   ;shr  bl, 4
10167                                   ;mov  bl, [ebx+hexchrs]
10168                                   ;xchg bl, al
10169                                   ;and  bl, 0Fh
10170                                   ;mov  ah, [ebx+hexchrs]
10171                                   ;pop  ebx
10172                                   ;retn
10173
10174                                  dwordtohex:
10175                                   ; INPUT ->
10176                                   ;      EAX = dword (binary number)
10177                                   ; OUTPUT ->
10178                                   ;      EDX:EAX = hexadecimal string
10179                                   ;
10180 00003309 50                       push  eax
10181 0000330A C1E810                   shr   eax, 16
10182 0000330D E8D2FFFFFF               call  wordtohex
10183 00003312 89C2                     mov   edx, eax
10184 00003314 58                       pop   eax
10185 00003315 E8CAFFFFFF               call  wordtohex
10186 0000331A C3                       retn
10187
10188                                  ; 10/05/2015
10189                                  hex_digits:
10190                                  hexchrs:
10191 0000331B 303132333435363738-      db '0123456789ABCDEF'
10192 00003324 39414243444546
10193
10194                                  ; Convert binary number to decimal/numeric string
10195                                  ; 06/11/2014
10196                                  ; Temporary Code
10197                                  ;
10198
```

```
10199                                  bintdstr:
10200                                        ; EAX = binary number
10201                                        ; ESI = decimal/numeric string address
10202                                        ; EBX = divisor (10)
10203                                        ; ECX = string length (<=10)
10204 0000332B 01CE                          add    esi, ecx
10205                                  btdstr0:
10206 0000332D 4E                            dec    esi
10207 0000332E 31D2                          xor    edx, edx
10208 00003330 F7F3                          div    ebx
10209 00003332 80C230                        add    dl, 30h
10210 00003335 8816                          mov    [esi], dl
10211 00003337 FEC9                          dec    cl
10212 00003339 740C                          jz     short btdstr2 ; 08/09/2016
10213 0000333B 09C0                          or     eax, eax
10214 0000333D 75EE                          jnz    short btdstr0
10215                                  btdstr1:
10216 0000333F 4E                            dec    esi
10217 00003340 C60620                          mov     byte [esi], 20h ; blank space
10218 00003343 FEC9                          dec    cl
10219 00003345 75F8                          jnz    short btdstr1
10220                                  btdstr2:
10221 00003347 C3                            retn
10222
10223                                  ; Calculate free memory pages on M.A.T.
10224                                  ; 06/11/2014
10225                                  ; Temporary Code
10226                                  ;
10227
10228                                  calc_free_mem:
10229 00003348 31D2                          xor    edx, edx
10230                                        ;xor   ecx, ecx
10231 0000334A 668B0D[E04D0100]              mov    cx, [mat_size] ; in pages
10232 00003351 C1E10A                        shl    ecx, 10      ; 1024 dwords per page
10233 00003354 BE00001000                    mov    esi, MEM_ALLOC_TBL
10234                                  cfm0:
10235 00003359 AD                            lodsd
10236 0000335A 51                            push   ecx
10237 0000335B B920000000                    mov    ecx, 32
10238                                  cfm1:
10239 00003360 D1E8                          shr    eax, 1
10240 00003362 7301                          jnc    short cfm2
10241 00003364 42                            inc    edx
10242                                  cfm2:
10243 00003365 E2F9                          loop   cfm1
10244 00003367 59                            pop    ecx
10245 00003368 E2EF                          loop   cfm0
10246 0000336A C3                            retn
10247
10248                                      %include 'diskio.s'  ; 07/03/2015
10249                                <1> ; ****************************************************************************
10250                                <1> ; TRDOS386.ASM (TRDOS 386 Kernel) - v2.0.0 - diskio.s
10251                                <1> ; ----------------------------------------------------------------------------
10252                                <1> ; Last Update: 15/01/2017
10253                                <1> ; ----------------------------------------------------------------------------
10254                                <1> ; Beginning: 24/01/2016
10255                                <1> ; ----------------------------------------------------------------------------
10256                                <1> ; Assembler: NASM version 2.11 (trdos386.s)
10257                                <1> ; ----------------------------------------------------------------------------
10258                                <1> ; Turkish Rational DOS
10259                                <1> ; Operating System Project v2.0 by ERDOGAN TAN (Beginning: 04/01/2016)
10260                                <1> ;
10261                                <1> ; Derived from 'Retro UNIX 386 Kernel - v0.2.1.0' source code by Erdogan Tan
10262                                <1> ; diskio.inc (22/08/2015)
10263                                <1> ;
10264                                <1> ; Derived from 'IBM PC-XT-286' BIOS source code (1986)
10265                                <1> ; ****************************************************************************
10266                                <1>
10267                                <1> ; Retro UNIX 386 v1 Kernel - DISKIO.INC
10268                                <1> ; Last Modification: 22/08/2015
10269                                <1> ;     (Initialized Disk Parameters Data is in 'DISKDATA.INC')
10270                                <1> ;     (Uninitialized Disk Parameters Data is in 'DISKBSS.INC')
10271                                <1>
10272                                <1> ; DISK I/O SYSTEM - Erdogan Tan (Retro UNIX 386 v1 project)
10273                                <1>
10274                                <1> ; ///////// DISK I/O SYSTEM ///////////////
10275                                <1>
10276                                <1> ; 06/02/2015
10277                                <1> diskette_io:
10278 0000336B 9C                    <1>       pushfd
10279 0000336C 0E                    <1>       push   cs
10280 0000336D E809000000            <1>       call   DISKETTE_IO_1
10281 00003372 C3                    <1>       retn
10282                                <1>
10283                                <1> ;;;;;; DISKETTE I/O ;;;;;;;;;;;;;;;;;;;; 06/02/2015 ;;;
10284                                <1> ;//////////////////////////////////////////////////////
10285                                <1>
10286                                <1> ; DISKETTE I/O - Erdogan Tan (Retro UNIX 386 v1 project)
10287                                <1> ; 20/02/2015
10288                                <1> ; 06/02/2015 (unix386.s)
10289                                <1> ; 16/12/2014 - 02/01/2015 (dsectrm2.s)
10290                                <1> ;
10291                                <1> ; Code (DELAY) modifications - AWARD BIOS 1999 (ADISK.EQU, COMMON.MAC)
10292                                <1> ;
10293                                <1> ; ADISK.EQU
10294                                <1>
10295                                <1> ;----- Wait control constants
10296                                <1>
10297                                <1> ;amount of time to wait while RESET is active.
10298                                <1>
10299                                <1> WAITCPU_RESET_ON   EQU    21          ;Reset on must last at least 14us
10300                                <1>                                       ;at 250 KBS xfer rate.
```

```
10301                                  <1>                                     ;see INTEL MCS, 1985, pg. 5-456
10302                                  <1>
10303                                  <1> WAITCPU_FOR_STATUS EQU    100       ;allow 30 microseconds for
10304                                  <1>                                     ;status register to become valid
10305                                  <1>                                     ;before re-reading.
10306                                  <1>
10307                                  <1> ;After sending a byte to NEC, status register may remain
10308                                  <1> ;incorrectly set for 24 us.
10309                                  <1>
10310                                  <1> WAITCPU_RQM_LOW           EQU    24           ;number of loops to check for
10311                                  <1>                                     ;RQM low.
10312                                  <1>
10313                                  <1> ; COMMON.MAC
10314                                  <1> ;
10315                                  <1> ;     Timing macros
10316                                  <1> ;
10317                                  <1>
10318                                  <1> %macro             SIODELAY 0               ; SHORT IODELAY
10319                                  <1>             jmp short $+2
10320                                  <1> %endmacro
10321                                  <1>
10322                                  <1> %macro             IODELAY  0               ; NORMAL IODELAY
10323                                  <1>             jmp short $+2
10324                                  <1>             jmp short $+2
10325                                  <1> %endmacro
10326                                  <1>
10327                                  <1> %macro             NEWIODELAY 0
10328                                  <1>             out    0ebh,al
10329                                  <1> %endmacro
10330                                  <1>
10331                                  <1> ; (According to) AWARD BIOS 1999 - ATORGS.ASM (dw -> equ, db -> equ)
10332                                  <1> ;;; WAIT_FOR_MEM
10333                                  <1> ;WAIT_FDU_INT_LO   equ    017798       ; 2.5 secs in 30 micro units.
10334                                  <1> ;WAIT_FDU_INT_HI   equ    1
10335                                  <1> WAIT_FDU_INT_LH           equ    83334        ; 27/02/2015 (2.5 seconds waiting)
10336                                  <1> ;;; WAIT_FOR_PORT
10337                                  <1> ;WAIT_FDU_SEND_LO  equ    16667        ; .5 secons in 30 us units.
10338                                  <1> ;WAIT_FDU_SEND_HI  equ    0
10339                                  <1> WAIT_FDU_SEND_LH   equ    16667        ; 27/02/2015
10340                                  <1> ;Time to wait while waiting for each byte of NEC results = .5
10341                                  <1> ;seconds.  .5 seconds = 500,000 micros.  500,000/30 = 16,667.
10342                                  <1> ;WAIT_FDU_RESULTS_LO      equ    16667        ; .5 seconds in 30 micro units.
10343                                  <1> ;WAIT_FDU_RESULTS_HI      equ    0
10344                                  <1> WAIT_FDU_RESULTS_LH       equ    16667  ; 27/02/2015
10345                                  <1> ;;; WAIT_REFRESH
10346                                  <1> ;amount of time to wait for head settle, per unit in parameter
10347                                  <1> ;table = 1 ms.
10348                                  <1> WAIT_FDU_HEAD_SETTLE      equ    33           ; 1 ms in 30 micro units.
10349                                  <1>
10350                                  <1>
10351                                  <1> ; ///////////////// DISKETTE I/O ////////////////
10352                                  <1>
10353                                  <1> ; 11/12/2014 (copy from IBM PC-XT Model 286 BIOS - POSTEQU.INC)
10354                                  <1>
10355                                  <1> ;----------------------------------------
10356                                  <1> ;     EQUATES USED BY POST AND BIOS    :
10357                                  <1> ;----------------------------------------
10358                                  <1>
10359                                  <1> ;--------- 8042 KEYBOARD INTERFACE AND DIAGNOSTIC CONTROL REGISTERS ------------
10360                                  <1> ;PORT_A            EQU    060H         ; 8042 KEYBOARD SCAN CODE/CONTROL PORT
10361                                  <1> ;PORT_B            EQU    061H         ; PORT B READ/WRITE DIAGNOSTIC REGISTER
10362                                  <1> ;REFRESH_BIT EQU    00010000B    ; REFRESH TEST BIT
10363                                  <1>
10364                                  <1> ;----------------------------------------
10365                                  <1> ;     CMOS EQUATES FOR THIS SYSTEM     :
10366                                  <1> ;--------------------------------------------------------------------------------
10367                                  <1> ;CMOS_PORT   EQU    070H         ; I/O ADDRESS OF CMOS ADDRESS PORT
10368                                  <1> ;CMOS_DATA   EQU    071H         ; I/O ADDRESS OF CMOS DATA PORT
10369                                  <1> ;NMI         EQU    10000000B    ; DISABLE NMI INTERRUPTS MASK -
10370                                  <1>                                 ;  HIGH BIT OF CMOS LOCATION ADDRESS
10371                                  <1>
10372                                  <1> ;---------- CMOS TABLE LOCATION ADDRESS'S ## ---------------------------------
10373                                  <1> CMOS_DISKETTE      EQU    010H         ; DISKETTE DRIVE TYPE BYTE      ;
10374                                  <1> ;           EQU    011H         ; - RESERVED                    ;C
10375                                  <1> CMOS_DISK    EQU    012H         ; FIXED DISK TYPE BYTE                  ;H
10376                                  <1> ;           EQU    013H         ; - RESERVED                    ;E
10377                                  <1> CMOS_EQUIP   EQU    014H         ; EQUIPMENT WORD LOW BYTE       ;C
10378                                  <1>
10379                                  <1> ;---------- DISKETTE EQUATES --------------------------------------------------
10380                                  <1> INT_FLAG     EQU    10000000B    ; INTERRUPT OCCURRENCE FLAG
10381                                  <1> DSK_CHG      EQU    10000000B    ; DISKETTE CHANGE FLAG MASK BIT
10382                                  <1> DETERMINED   EQU    00010000B    ; SET STATE DETERMINED IN STATE BITS
10383                                  <1> HOME         EQU    00010000B    ; TRACK 0 MASK
10384                                  <1> SENSE_DRV_ST EQU    00000100B    ; SENSE DRIVE STATUS COMMAND
10385                                  <1> TRK_SLAP     EQU    030H         ; CRASH STOP (48 TPI DRIVES)
10386                                  <1> QUIET_SEEK   EQU    00AH         ; SEEK TO TRACK 10
10387                                  <1> ;MAX_DRV     EQU    2            ; MAX NUMBER OF DRIVES
10388                                  <1> HD12_SETTLE  EQU    15           ; 1.2 M HEAD SETTLE TIME
10389                                  <1> HD320_SETTLE EQU    20           ; 320 K HEAD SETTLE TIME
10390                                  <1> MOTOR_WAIT   EQU    37           ; 2 SECONDS OF COUNTS FOR MOTOR TURN OFF
10391                                  <1>
10392                                  <1> ;---------- DISKETTE ERRORS ---------------------------------------------------
10393                                  <1> ;TIME_OUT    EQU    080H         ; ATTACHMENT FAILED TO RESPOND
10394                                  <1> ;BAD_SEEK    EQU    040H         ; SEEK OPERATION FAILED
10395                                  <1> BAD_NEC      EQU    020H         ; DISKETTE CONTROLLER HAS FAILED
10396                                  <1> BAD_CRC      EQU    010H         ; BAD CRC ON DISKETTE READ
10397                                  <1> MED_NOT_FND  EQU    00CH         ; MEDIA TYPE NOT FOUND
10398                                  <1> DMA_BOUNDARY EQU    009H         ; ATTEMPT TO DMA ACROSS 64K BOUNDARY
10399                                  <1> BAD_DMA      EQU    008H         ; DMA OVERRUN ON OPERATION
10400                                  <1> MEDIA_CHANGE EQU    006H         ; MEDIA REMOVED ON DUAL ATTACH CARD
10401                                  <1> RECORD_NOT_FND     EQU    004H         ; REQUESTED SECTOR NOT FOUND
10402                                  <1> WRITE_PROTECT      EQU    003H         ; WRITE ATTEMPTED ON WRITE PROTECT DISK
```

```
10403                                  <1> BAD_ADDR_MARK      EQU    002H         ; ADDRESS MARK NOT FOUND
10404                                  <1> BAD_CMD     EQU    001H         ; BAD COMMAND PASSED TO DISKETTE I/O
10405                                  <1>
10406                                  <1> ;---------- DISK CHANGE LINE EQUATES ----------------------------------------
10407                                  <1> NOCHGLN     EQU    001H         ; NO DISK CHANGE LINE AVAILABLE
10408                                  <1> CHGLN       EQU    002H         ; DISK CHANGE LINE AVAILABLE
10409                                  <1>
10410                                  <1> ;---------- MEDIA/DRIVE STATE INDICATORS ------------------------------------
10411                                  <1> TRK_CAPA    EQU    00000001B    ; 80 TRACK CAPABILITY
10412                                  <1> FMT_CAPA    EQU    00000010B    ; MULTIPLE FORMAT CAPABILITY (1.2M)
10413                                  <1> DRV_DET     EQU    00000100B    ; DRIVE DETERMINED
10414                                  <1> MED_DET     EQU    00010000B    ; MEDIA DETERMINED BIT
10415                                  <1> DBL_STEP    EQU    00100000B    ; DOUBLE STEP BIT
10416                                  <1> RATE_MSK    EQU    11000000B    ; MASK FOR CLEARING ALL BUT RATE
10417                                  <1> RATE_500    EQU    00000000B    ; 500 KBS DATA RATE
10418                                  <1> RATE_300    EQU    01000000B    ; 300 KBS DATA RATE
10419                                  <1> RATE_250    EQU    10000000B    ; 250 KBS DATA RATE
10420                                  <1> STRT_MSK    EQU    00001100B    ; OPERATION START RATE MASK
10421                                  <1> SEND_MSK    EQU    11000000B    ; MASK FOR SEND RATE BITS
10422                                  <1>
10423                                  <1> ;---------- MEDIA/DRIVE STATE INDICATORS COMPATIBILITY ------------------------
10424                                  <1> M3D3U       EQU    00000000B    ; 360 MEDIA/DRIVE NOT ESTABLISHED
10425                                  <1> M3D1U       EQU    00000001B    ; 360 MEDIA,1.2DRIVE NOT ESTABLISHED
10426                                  <1> M1D1U       EQU    00000010B    ; 1.2 MEDIA/DRIVE NOT ESTABLISHED
10427                                  <1> MED_UNK     EQU    00000111B    ; NONE OF THE ABOVE
10428                                  <1>
10429                                  <1> ;---------- INTERRUPT EQUATES ------------------------------------------------
10430                                  <1> ;EOI        EQU    020H         ; END OF INTERRUPT COMMAND TO 8259
10431                                  <1> ;INTA00            EQU    020H         ; 8259 PORT
10432                                  <1> INTA01             EQU    021H         ; 8259 PORT
10433                                  <1> INTB00             EQU    0A0H         ; 2ND 8259
10434                                  <1> INTB01             EQU    0A1H         ;
10435                                  <1>
10436                                  <1> ;--------------------------------------------------------------------------------
10437                                  <1> DMA08       EQU    008H         ; DMA STATUS REGISTER PORT ADDRESS
10438                                  <1> DMA         EQU    000H         ; DMA CH.0 ADDRESS REGISTER PORT ADDRESS
10439                                  <1> DMA18       EQU    0D0H         ; 2ND DMA STATUS PORT ADDRESS
10440                                  <1> DMA1        EQU    0C0H         ; 2ND DMA CH.0 ADDRESS REGISTER ADDRESS
10441                                  <1> ;--------------------------------------------------------------------------------
10442                                  <1> ;TIMER             EQU    040H         ; 8254 TIMER - BASE ADDRESS
10443                                  <1>
10444                                  <1> ;--------------------------------------------------------------------------------
10445                                  <1> DMA_PAGE    EQU    081H         ; START OF DMA PAGE REGISTERS
10446                                  <1>
10447                                  <1> ; 06/02/2015 (unix386.s, protected mode modifications)
10448                                  <1> ; (unix386.s <-- dsectrm2.s)
10449                                  <1> ; 11/12/2014 (copy from IBM PC-XT Model 286 BIOS - DSEG.INC)
10450                                  <1>
10451                                  <1> ; 27/05/2016 - TRDOS 386 (TRDOS v2.0)
10452                                  <1> ; 10/12/2014
10453                                  <1> ;
10454                                  <1> ;int40h:
10455                                  <1> ;     pushf
10456                                  <1> ;     push   cs
10457                                  <1> ;     ;cli
10458                                  <1> ;     call   DISKETTE_IO_1
10459                                  <1> ;     retn
10460                                  <1>
10461                                  <1> ; DSKETTE ----- 04/21/86 DISKETTE BIOS
10462                                  <1> ; (IBM PC XT Model 286 System BIOS Source Code, 04-21-86)
10463                                  <1> ;
10464                                  <1>
10465                                  <1> ;-- INT13H --------------------------------------------------------------------
10466                                  <1> ; DISKETTE I/O
10467                                  <1> ;     THIS INTERFACE PROVIDES ACCESS TO THE 5 1/4 INCH 360 KB,
10468                                  <1> ;     1.2 MB, 720 KB AND 1.44 MB DISKETTE DRIVES.
10469                                  <1> ; INPUT
10470                                  <1> ;     (AH) =  00H RESET DISKETTE SYSTEM
10471                                  <1> ;             HARD RESET TO NEC, PREPARE COMMAND, RECALIBRATE REQUIRED
10472                                  <1> ;             ON ALL DRIVES
10473                                  <1> ;--------------------------------------------------------------------------------
10474                                  <1> ;     (AH)= 01H  READ THE STATUS OF THE SYSTEM INTO (AH)
10475                                  <1> ;             @DISKETTE_STATUS FROM LAST OPERATION IS USED
10476                                  <1> ;--------------------------------------------------------------------------------
10477                                  <1> ;     REGISTERS FOR READ/WRITE/VERIFY/FORMAT
10478                                  <1> ;     (DL) - DRIVE NUMBER (0-1 ALLOWED, VALUE CHECKED)
10479                                  <1> ;     (DH) - HEAD NUMBER (0-1 ALLOWED, NOT VALUE CHECKED)
10480                                  <1> ;     (CH) - TRACK NUMBER (NOT VALUE CHECKED)
10481                                  <1> ;             MEDIA  DRIVE  TRACK NUMBER
10482                                  <1> ;             320/360      320/360          0-39
10483                                  <1> ;             320/360      1.2M       0-39
10484                                  <1> ;             1.2M   1.2M       0-79
10485                                  <1> ;             720K   720K       0-79
10486                                  <1> ;             1.44M  1.44M      0-79
10487                                  <1> ;     (CL) -       SECTOR NUMBER (NOT VALUE CHECKED, NOT USED FOR FORMAT)
10488                                  <1> ;             MEDIA  DRIVE  SECTOR NUMBER
10489                                  <1> ;             320/360      320/360           1-8/9
10490                                  <1> ;             320/360      1.2M        1-8/9
10491                                  <1> ;             1.2M   1.2M       1-15
10492                                  <1> ;             720K   720K       1-9
10493                                  <1> ;             1.44M  1.44M      1-18
10494                                  <1> ;     (AL)   NUMBER OF SECTORS (NOT VALUE CHECKED)
10495                                  <1> ;             MEDIA  DRIVE  MAX NUMBER OF SECTORS
10496                                  <1> ;             320/360      320/360              8/9
10497                                  <1> ;             320/360      1.2M           8/9
10498                                  <1> ;             1.2M   1.2M         15
10499                                  <1> ;             720K   720K         9
10500                                  <1> ;             1.44M  1.44M        18
10501                                  <1> ;
10502                                  <1> ;     (ES:BX) - ADDRESS OF BUFFER (NOT REQUIRED FOR VERIFY)
10503                                  <1> ;
10504                                  <1> ;--------------------------------------------------------------------------------
```

```
10505                           <1> ;     (AH)= 02H  READ THE DESIRED SECTORS INTO MEMORY
10506                           <1> ;-------------------------------------------------------------------------------
10507                           <1> ;     (AH)= 03H  WRITE THE DESIRED SECTORS FROM MEMORY
10508                           <1> ;-------------------------------------------------------------------------------
10509                           <1> ;     (AH)= 04H  VERIFY THE DESIRED SECTORS
10510                           <1> ;-------------------------------------------------------------------------------
10511                           <1> ;     (AH)= 05H  FORMAT THE DESIRED TRACK
10512                           <1> ;           (ES,BX) MUST POINT TO THE COLLECTION OF DESIRED ADDRESS FIELDS
10513                           <1> ;           FOR THE      TRACK. EACH FIELD IS COMPOSED OF 4 BYTES, (C,H,R,N),
10514                           <1> ;           WHERE C = TRACK NUMBER, H=HEAD NUMBER, R = SECTOR NUMBER,
10515                           <1> ;           N= NUMBER OF BYTES PER SECTOR (00=128,01=256,02=512,03=1024),
10516                           <1> ;           THERE MUST BE ONE ENTRY FOR EVERY SECTOR ON THE TRACK.
10517                           <1> ;           THIS INFORMATION IS USED TO FIND THE REQUESTED SECTOR DURING
10518                           <1> ;           READ/WRITE ACCESS.
10519                           <1> ;           PRIOR TO FORMATTING A DISKETTE, IF THERE EXISTS MORE THAN
10520                           <1> ;           ONE SUPPORTED MEDIA FORMAT TYPE WITHIN THE DRIVE IN QUESTION,
10521                           <1> ;           THEN "SET DASD TYPE" (INT 13H, AH = 17H) OR 'SET MEDIA TYPE'
10522                           <1> ;           (INT 13H, AH =  18H) MUST BE CALLED TO SET THE DISKETTE TYPE
10523                           <1> ;           THAT IS TO BE FORMATTED. IF "SET DASD TYPE" OR "SET MEDIA TYPE"
10524                           <1> ;           IS NOT CALLED, THE FORMAT ROUTINE WILL ASSUME THE
10525                           <1> ;           MEDIA FORMAT TO BE THE MAXIMUM CAPACITY OF THE DRIVE.
10526                           <1> ;
10527                           <1> ;           THESE PARAMETERS OF DISK BASE MUST BE CHANGED IN ORDER TO
10528                           <1> ;           FORMAT THE FOLLOWING MEDIAS:
10529                           <1> ;           -----------------------------------------
10530                           <1> ;           : MEDIA  :     DRIVE      : PARM 1 : PARM 2 :
10531                           <1> ;           -----------------------------------------
10532                           <1> ;           : 320K  : 320K/360K/1.2M :  50H   :   8    :
10533                           <1> ;           : 360K  : 320K/360K/1.2M :  50H   :   9    :
10534                           <1> ;           : 1.2M  : 1.2M           :  54H   :  15    :
10535                           <1> ;           : 720K  : 720K/1.44M     :  50H   :   9    :
10536                           <1> ;           : 1.44M       : 1.44M          :  6CH   :  18    :
10537                           <1> ;           -----------------------------------------
10538                           <1> ;           NOTES: - PARM 1 = GAP LENGTH FOR FORMAT
10539                           <1> ;                  - PARM 2 = EOT (LAST SECTOR ON TRACK)
10540                           <1> ;                  - DISK BASE IS POINTED BY DISK POINTER LOCATED
10541                           <1> ;                   AT ABSOLUTE ADDRESS 0:78.
10542                           <1> ;                  - WHEN FORMAT OPERATIONS ARE COMPLETE, THE PARAMETERS
10543                           <1> ;                   SHOULD BE RESTORED TO THEIR RESPECTIVE INITIAL VALUES.
10544                           <1> ;-------------------------------------------------------------------------------
10545                           <1> ;     (AH) = 08H READ DRIVE PARAMETERS
10546                           <1> ;     REGISTERS
10547                           <1> ;       INPUT
10548                           <1> ;         (DL) - DRIVE NUMBER (0-1 ALLOWED, VALUE CHECKED)
10549                           <1> ;          ** 27/05/2016 - TRDOS 386 (TRDOS v2.0) **
10550                           <1> ;            ** EBX = Buffer address for floppy disk parameters table **
10551                           <1> ;       OUTPUT
10552                           <1> ;         (ES:DI) POINTS TO DRIVE PARAMETER TABLE
10553                           <1> ;         *** TRDOS 386 note: floppy disk parameter table (16 bytes)
10554                           <1> ;         will be returned to user in EBX, buffer address *** 27/05/2016 ***
10555                           <1> ;
10556                           <1> ;         (CH) - LOW ORDER 8 OF 10 BITS MAXIMUM NUMBER OF TRACKS
10557                           <1> ;         (CL) - BITS 7 & 6 - HIGH ORDER TWO BITS OF MAXIMUM TRACKS
10558                           <1> ;                BITS 5 THRU 0 - MAXIMUM SECTORS PER TRACK
10559                           <1> ;         (DH) - MAXIMUM HEAD NUMBER
10560                           <1> ;         (DL) - NUMBER OF DISKETTE DRIVES INSTALLED
10561                           <1> ;         (BH) - 0
10562                           <1> ;         (BL) - BITS 7 THRU 4 - 0
10563                           <1> ;                BITS 3 THRU 0 - VALID DRIVE TYPE VALUE IN CMOS
10564                           <1> ;         (AX) - 0
10565                           <1> ;      UNDER THE FOLLOWING CIRCUMSTANCES:
10566                           <1> ;         (1) THE DRIVE NUMBER IS INVALID,
10567                           <1> ;         (2) THE DRIVE TYPE IS UNKNOWN AND CMOS IS NOT PRESENT,
10568                           <1> ;         (3) THE DRIVE TYPE IS UNKNOWN AND CMOS IS BAD,
10569                           <1> ;         (4) OR THE DRIVE TYPE IS UNKNOWN AND THE CMOS DRIVE TYPE IS INVALID
10570                           <1> ;         THEN ES,AX,BX,CX,DH,DI=0 ; DL=NUMBER OF DRIVES.
10571                           <1> ;         IF NO DRIVES ARE PRESENT THEN: ES,AX,BX,CX,DX,DI=0.
10572                           <1> ;         @DISKETTE_STATUS = 0 AND CY IS RESET.
10573                           <1> ;-------------------------------------------------------------------------------
10574                           <1> ;     (AH)= 15H  READ DASD TYPE
10575                           <1> ;     OUTPUT REGISTERS
10576                           <1> ;     (AH) - ON RETURN IF CARRY FLAG NOT SET, OTHERWISE ERROR
10577                           <1> ;            00 - DRIVE NOT PRESENT
10578                           <1> ;            01 - DISKETTE, NO CHANGE LINE AVAILABLE
10579                           <1> ;            02 - DISKETTE, CHANGE LINE AVAILABLE
10580                           <1> ;            03 - RESERVED (FIXED DISK)
10581                           <1> ;     (DL) - DRIVE NUMBER (0-1 ALLOWED, VALUE CHECKED)
10582                           <1> ;-------------------------------------------------------------------------------
10583                           <1> ;     (AH)= 16H  DISK CHANGE LINE STATUS
10584                           <1> ;     OUTPUT REGISTERS
10585                           <1> ;     (AH) - 00 - DISK CHANGE LINE NOT ACTIVE
10586                           <1> ;            06 - DISK CHANGE LINE ACTIVE & CARRY BIT ON
10587                           <1> ;     (DL) - DRIVE NUMBER (0-1 ALLOWED, VALUE CHECKED)
10588                           <1> ;-------------------------------------------------------------------------------
10589                           <1> ;     (AH)= 17H  SET DASD TYPE FOR FORMAT
10590                           <1> ;     INPUT REGISTERS
10591                           <1> ;     (AL) - 00 - NOT USED
10592                           <1> ;            01 - DISKETTE 320/360K IN 360K DRIVE
10593                           <1> ;            02 - DISKETTE 360K IN 1.2M DRIVE
10594                           <1> ;            03 - DISKETTE 1.2M IN 1.2M DRIVE
10595                           <1> ;            04 - DISKETTE 720K IN 720K DRIVE
10596                           <1> ;     (DL) - DRIVE NUMBER (0-1 ALLOWED, VALUE CHECKED:
10597                           <1> ;            (DO NOT USE WHEN DISKETTE ATTACH CARD USED)
10598                           <1> ;-------------------------------------------------------------------------------
10599                           <1> ;     (AH)= 18H  SET MEDIA TYPE FOR FORMAT
10600                           <1> ;     INPUT REGISTERS
10601                           <1> ;     (CH) - LOW ORDER 8 OF 10 BITS MAXIMUM TRACKS
10602                           <1> ;     (CL) - BITS 7 & 6 - HIGH ORDER TWO BITS OF MAXIMUM TRACKS
10603                           <1> ;            BITS 5 THRU 0 - MAXIMUM SECTORS PER TRACK
10604                           <1> ;     (DL) - DRIVE NUMBER (0-1 ALLOWED, VALUE CHACKED)
10605                           <1> ;     OUTPUT REGISTERS:
10606                           <1> ;     (ES:DI) - POINTER TO DRIVE PARAMETERS TABLE FOR THIS MEDIA TYPE,
```

```
10607                                  <1> ;             UNCHANGED IF (AH) IS NON-ZERO
10608                                  <1> ;     (AH) - 00H, CY = 0, TRACK AND SECTORS/TRACK COMBINATION IS SUPPORTED
10609                                  <1> ;          - 01H, CY = 1, FUNCTION IS NOT AVAILABLE
10610                                  <1> ;          - 0CH, CY = 1, TRACK AND SECTORS/TRACK COMBINATION IS NOT SUPPORTED
10611                                  <1> ;          - 80H, CY = 1, TIME OUT (DISKETTE NOT PRESENT)
10612                                  <1> ;--------------------------------------------------------------------------------
10613                                  <1> ;     DISK CHANGE STATUS IS ONLY CHECKED WHEN A MEDIA SPECIFIED IS OTHER
10614                                  <1> ;     THAN 360 KB DRIVE. IF THE DISK CHANGE LINE IS FOUND TO BE
10615                                  <1> ;     ACTIVE THE FOLLOWING ACTIONS TAKE PLACE:
10616                                  <1> ;           ATTEMPT TO RESET DISK CHANGE LINE TO INACTIVE STATE.
10617                                  <1> ;           IF ATTEMPT SUCCEEDS SET DASD TYPE FOR FORMAT AND RETURN DISK
10618                                  <1> ;           CHANGE ERROR CODE
10619                                  <1> ;           IF ATTEMPT FAILS RETURN TIMEOUT ERROR CODE AND SET DASD TYPE
10620                                  <1> ;           TO A PREDETERMINED STATE INDICATING MEDIA TYPE UNKNOWN.
10621                                  <1> ;     IF THE DISK CHANGE LINE IN INACTIVE PERFORM SET DASD TYPE FOR FORMAT.
10622                                  <1> ;
10623                                  <1> ; DATA VARIABLE -- @DISK_POINTER
10624                                  <1> ;     DOUBLE WORD POINTER TO THE CURRENT SET OF DISKETTE PARAMETERS
10625                                  <1> ;--------------------------------------------------------------------------------
10626                                  <1> ; OUTPUT FOR ALL FUNCTIONS
10627                                  <1> ;     AH = STATUS OF OPERATION
10628                                  <1> ;           STATUS BITS ARE DEFINED IN THE EQUATES FOR @DISKETTE_STATUS
10629                                  <1> ;           VARIABLE IN THE DATA SEGMENT OF THIS MODULE
10630                                  <1> ;     CY = 0 SUCCESSFUL OPERATION (AH=0 ON RETURN, EXCEPT FOR READ DASD
10631                                  <1> ;           TYPE AH=(15)).
10632                                  <1> ;     CY = 1 FAILED OPERATION (AH HAS ERROR REASON)
10633                                  <1> ;     FOR READ/WRITE/VERIFY
10634                                  <1> ;           DS,BX,DX,CX PRESERVED
10635                                  <1> ;     NOTE: IF AN ERROR IS REPORTED BY THE DISKETTE CODE, THE APPROPRIATE
10636                                  <1> ;           ACTION IS TO RESET THE DISKETTE, THEN RETRY THE OPERATION.
10637                                  <1> ;           ON READ ACCESSES, NO MOTOR START DELAY IS TAKEN, SO THAT
10638                                  <1> ;           THREE RETRIES ARE REQUIRED ON READS TO ENSURE THAT THE
10639                                  <1> ;           PROBLEM IS NOT DUE TO MOTOR START-UP.
10640                                  <1> ;--------------------------------------------------------------------------------
10641                                  <1> ;
10642                                  <1> ; DISKETTE STATE MACHINE - ABSOLUTE ADDRESS 40:90 (DRIVE A) & 91 (DRIVE B)
10643                                  <1> ;
10644                                  <1> ;   ---------------------------------------------------------------
10645                                  <1> ;   |       |       |       |       |       |       |       |       |
10646                                  <1> ;   |   7   |   6   |   5   |   4   |   3   |   2   |   1   |   0   |
10647                                  <1> ;   |       |       |       |       |       |       |       |       |
10648                                  <1> ;   ---------------------------------------------------------------
10649                                  <1> ;     |       |       |       |       |       |       |       |
10650                                  <1> ;     |       |       |       |       |       -----------------
10651                                  <1> ;     |       |       |       |       |               |
10652                                  <1> ;     |       |       |       |    RESERVED           |
10653                                  <1> ;     |       |       |       |                 PRESENT STATE
10654                                  <1> ;     |       |       |       |       000: 360K IN 360K DRIVE UNESTABLISHED
10655                                  <1> ;     |       |       |       |       001: 360K IN 1.2M DRIVE UNESTABLISHED
10656                                  <1> ;     |       |       |       |       010: 1.2M IN 1.2M DRIVE UNESTABLISHED
10657                                  <1> ;     |       |       |       |       011: 360K IN 360K DRIVE ESTABLISHED
10658                                  <1> ;     |       |       |       |       100: 360K IN 1.2M DRIVE ESTABLISHED
10659                                  <1> ;     |       |       |       |       101: 1.2M IN 1.2M DRIVE ESTABLISHED
10660                                  <1> ;     |       |       |       |       110: RESERVED
10661                                  <1> ;     |       |       |       |       111: NONE OF THE ABOVE
10662                                  <1> ;     |       |       |       |
10663                                  <1> ;     |       |       |       ------>       MEDIA/DRIVE ESTABLISHED
10664                                  <1> ;     |       |       |
10665                                  <1> ;     |       |       --------------->     DOUBLE STEPPING REQUIRED (360K IN 1.2M
10666                                  <1> ;     |       |                       DRIVE)
10667                                  <1> ;     |       |
10668                                  <1> ;     --------------------------------->  DATA TRANSFER RATE FOR THIS DRIVE:
10669                                  <1> ;
10670                                  <1> ;                                         00: 500 KBS
10671                                  <1> ;                                         01: 300 KBS
10672                                  <1> ;                                         10: 250 KBS
10673                                  <1> ;                                         11: RESERVED
10674                                  <1> ;
10675                                  <1> ;
10676                                  <1> ;--------------------------------------------------------------------------------
10677                                  <1> ; STATE OPERATION STARTED - ABSOLUTE ADDRESS 40:92 (DRIVE A) & 93 (DRIVE B)
10678                                  <1> ;--------------------------------------------------------------------------------
10679                                  <1> ; PRESENT CYLINDER NUMBER - ABSOLUTE ADDRESS 40:94 (DRIVE A) & 95 (DRIVE B)
10680                                  <1> ;--------------------------------------------------------------------------------
10681                                  <1>
10682                                  <1> struc MD
10683 00000000 <res 00000001>         <1>       .SPEC1       resb   1      ; SRT=D, HD UNLOAD=0F - 1ST SPECIFY BYTE
10684 00000001 <res 00000001>         <1>       .SPEC2       resb   1      ; HD LOAD=1, MODE=DMA - 2ND SPECIFY BYTE
10685 00000002 <res 00000001>         <1>       .OFF_TIM     resb   1      ; WAIT TIME AFTER OPERATION TILL MOTOR OFF
10686 00000003 <res 00000001>         <1>       .BYT_SEC     resb   1      ; 512 BYTES/SECTOR
10687 00000004 <res 00000001>         <1>       .SEC_TRK     resb   1      ; EOT (LAST SECTOR ON TRACK)
10688 00000005 <res 00000001>         <1>       .GAP         resb   1      ; GAP LENGTH
10689 00000006 <res 00000001>         <1>       .DTL         resb   1      ; DTL
10690 00000007 <res 00000001>         <1>       .GAP3        resb   1      ; GAP LENGTH FOR FORMAT
10691 00000008 <res 00000001>         <1>       .FIL_BYT     resb   1      ; FILL BYTE FOR FORMAT
10692 00000009 <res 00000001>         <1>       .HD_TIM              resb   1      ; HEAD SETTLE TIME (MILLISECONDS)
10693 0000000A <res 00000001>         <1>       .STR_TIM     resb   1      ; MOTOR START TIME (1/8 SECONDS)
10694 0000000B <res 00000001>         <1>       .MAX_TRK     resb   1      ; MAX. TRACK NUMBER
10695 0000000C <res 00000001>         <1>       .RATE        resb   1      ; DATA TRANSFER RATE
10696                                 <1> endstruc
10697                                 <1>
10698                                 <1> BIT7OFF     EQU    7FH
10699                                 <1> BIT7ON      EQU    80H
10700                                 <1>
10701                                 <1> ;;int13h: ; 16/02/2015
10702                                 <1> ;; 16/02/2015 - 21/02/2015
10703                                 <1> int40h:
10704 00003373 9C                     <1>       pushfd
10705 00003374 0E                     <1>       push   cs
10706 00003375 E801000000             <1>       call   DISKETTE_IO_1
10707 0000337A C3                     <1>       retn
10708                                 <1>
```

```
10709                                  <1> DISKETTE_IO_1:
10710                                  <1>
10711 0000337B FB                      <1>       STI                     ; INTERRUPTS BACK ON
10712 0000337C 55                      <1>       PUSH   eBP              ; USER REGISTER
10713 0000337D 57                      <1>       PUSH   eDI              ; USER REGISTER
10714 0000337E 52                      <1>       PUSH   eDX              ; HEAD #, DRIVE # OR USER REGISTER
10715 0000337F 53                      <1>       PUSH   eBX              ; BUFFER OFFSET PARAMETER OR REGISTER
10716 00003380 51                      <1>       PUSH   eCX              ; TRACK #-SECTOR # OR USER REGISTER
10717 00003381 89E5                    <1>       MOV    eBP,eSP                  ; BP     => PARAMETER LIST DEP. ON AH
10718                                  <1>                               ; [BP]   = SECTOR #
10719                                  <1>                               ; [BP+1] = TRACK #
10720                                  <1>                               ; [BP+2] = BUFFER OFFSET
10721                                  <1>                               ; FOR RETURN OF DRIVE PARAMETERS:
10722                                  <1>                               ; CL/[BP] = BITS 7&6 HI BITS OF MAX CYL
10723                                  <1>                               ;           BITS 0-5 MAX SECTORS/TRACK
10724                                  <1>                               ; CH/[BP+1] = LOW 8 BITS OF MAX CYL.
10725                                  <1>                               ; BL/[BP+2] = BITS 7-4 = 0
10726                                  <1>                               ;             BITS 3-0 = VALID CMOS TYPE
10727                                  <1>                               ; BH/[BP+3] = 0
10728                                  <1>                               ; DL/[BP+4] = # DRIVES INSTALLED
10729                                  <1>                               ; DH/[BP+5] = MAX HEAD #
10730                                  <1>                               ; DI/[BP+6] = OFFSET TO DISK BASE
10731 00003383 06                      <1>       push   es ; 06/02/2015
10732 00003384 1E                      <1>       PUSH   DS               ; BUFFER SEGMENT PARM OR USER REGISTER
10733 00003385 56                      <1>       PUSH   eSI              ; USER REGISTERS
10734                                  <1>       ;CALL  DDS              ; SEGMENT OF BIOS DATA AREA TO DS
10735                                  <1>       ;mov   cx, cs
10736                                  <1>       ;mov   ds, cx
10737 00003386 66B91000                <1>       mov    cx, KDATA
10738 0000338A 8ED9                    <1>         mov     ds, cx
10739 0000338C 8EC1                    <1>         mov     es, cx
10740                                  <1>
10741                                  <1>       ;CMP   AH,(FNC_TAE-FNC_TAB)/2    ; CHECK FOR > LARGEST FUNCTION
10742 0000338E 80FC19                  <1>       cmp    ah,(FNC_TAE-FNC_TAB)/4  ; 18/02/2015
10743 00003391 7202                    <1>       JB     short OK_FUNC       ; FUNCTION OK
10744 00003393 B414                    <1>       MOV    AH,14H              ; REPLACE WITH KNOWN INVALID FUNCTION
10745                                  <1> OK_FUNC:
10746 00003395 80FC01                  <1>       CMP    AH,1                ; RESET OR STATUS ?
10747 00003398 760C                    <1>       JBE    short OK_DRV        ; IF RESET OR STATUS DRIVE ALWAYS OK
10748 0000339A 80FC08                  <1>       CMP    AH,8                ; READ DRIVE PARMS ?
10749 0000339D 7407                    <1>       JZ     short OK_DRV        ; IF SO DRIVE CHECKED LATER
10750 0000339F 80FA01                  <1>       CMP    DL,1                ; DRIVES 0 AND 1 OK
10751 000033A2 7602                    <1>       JBE    short OK_DRV        ; IF 0 OR 1 THEN JUMP
10752 000033A4 B414                    <1>       MOV    AH,14H              ; REPLACE WITH KNOWN INVALID FUNCTION
10753                                  <1> OK_DRV:
10754 000033A6 31C9                    <1>       xor    ecx, ecx
10755                                  <1>       ;mov   esi, ecx ; 08/02/2015
10756 000033A8 89CF                    <1>       mov    edi, ecx ; 08/02/2015
10757 000033AA 88E1                    <1>       MOV    CL,AH               ; CL = FUNCTION
10758                                  <1>       ;XOR   CH,CH               ; CX = FUNCTION
10759                                  <1>       ;SHL   CL, 1               ; FUNCTION TIMES 2
10760 000033AC C0E102                  <1>       SHL    CL, 2 ; 20/02/2015  ; FUNCTION TIMES 4 (for 32 bit offset)
10761 000033AF BB[E7330000]            <1>       MOV    eBX,FNC_TAB         ; LOAD START OF FUNCTION TABLE
10762 000033B4 01CB                    <1>       ADD    eBX,eCX                  ; ADD OFFSET INTO TABLE => ROUTINE
10763 000033B6 88F4                    <1>       MOV    AH,DH               ; AX = HEAD #,# OF SECTORS OR DASD TYPE
10764 000033B8 30F6                    <1>       XOR    DH,DH               ; DX = DRIVE #
10765 000033BA 6689C6                  <1>       MOV    SI,AX               ; SI = HEAD #,# OF SECTORS OR DASD TYPE
10766 000033BD 6689D7                  <1>       MOV     DI,DX                   ; DI = DRIVE #
10767                                  <1>       ;
10768                                  <1>       ; 11/12/2014
10769 000033C0 8815[E55C0000]          <1>         mov     [cfd], dl             ; current floppy drive (for 'GET_PARM')
10770                                  <1>       ;
10771 000033C6 8A25[504E0100]          <1>       MOV    AH, [DSKETTE_STATUS]      ; LOAD STATUS TO AH FOR STATUS FUNCTION
10772 000033CC C605[504E0100]00        <1>       MOV    byte [DSKETTE_STATUS],0   ; INITIALIZE FOR ALL OTHERS
10773                                  <1>
10774                                  <1> ;     THROUGHOUT THE DISKETTE BIOS, THE FOLLOWING INFORMATION IS CONTAINED IN
10775                                  <1> ;     THE FOLLOWING MEMORY LOCATIONS AND REGISTERS. NOT ALL DISKETTE BIOS
10776                                  <1> ;     FUNCTIONS REQUIRE ALL OF THESE PARAMETERS.
10777                                  <1> ;
10778                                  <1> ;           DI     : DRIVE #
10779                                  <1> ;           SI-HI  : HEAD #
10780                                  <1> ;           SI-LOW : # OF SECTORS OR DASD TYPE FOR FORMAT
10781                                  <1> ;           ES     : BUFFER SEGMENT
10782                                  <1> ;           [BP]   : SECTOR #
10783                                  <1> ;           [BP+1] : TRACK #
10784                                  <1> ;           [BP+2] : BUFFER OFFSET
10785                                  <1> ;
10786                                  <1> ;     ACROSS CALLS TO SUBROUTINES THE CARRY FLAG (CY=1), WHERE INDICATED IN
10787                                  <1> ;     SUBROUTINE PROLOGUES, REPRESENTS AN EXCEPTION RETURN (NORMALLY AN ERROR
10788                                  <1> ;     CONDITION). IN MOST CASES, WHEN CY = 1, @DSKETTE_STATUS CONTAINS THE
10789                                  <1> ;     SPECIFIC ERROR CODE.
10790                                  <1> ;
10791                                  <1>                                   ; (AH) = @DSKETTE_STATUS
10792 000033D3 FF13                    <1>       CALL   dWORD [eBX]          ; CALL THE REQUESTED FUNCTION
10793 000033D5 5E                      <1>       POP    eSI                  ; RESTORE ALL REGISTERS
10794 000033D6 1F                      <1>       POP    DS
10795 000033D7 07                      <1>       pop    es     ; 06/02/2015
10796 000033D8 59                      <1>       POP    eCX
10797 000033D9 5B                      <1>       POP    eBX
10798 000033DA 5A                      <1>       POP    eDX
10799 000033DB 5F                      <1>       POP    eDI
10800 000033DC 89E5                    <1>       MOV    eBP, eSP
10801 000033DE 50                      <1>       PUSH   eAX
10802 000033DF 9C                      <1>       PUSHFd
10803 000033E0 58                      <1>       POP    eAX
10804                                  <1>       ;MOV   [BP+6], AX
10805 000033E1 89450C                  <1>       mov    [ebp+12], eax  ; 18/02/2015, flags
10806 000033E4 58                      <1>       POP    eAX
10807 000033E5 5D                      <1>       POP    eBP
10808 000033E6 CF                      <1>       IRETd
10809                                  <1>
10810                                  <1> ;---------------------------------------------------------------------------
```

```
10811                                   <1> ; DW --> dd (06/02/2015)
10812 000033E7 [4B340000]               <1> FNC_TAB     dd     DSK_RESET           ; AH = 00H; RESET
10813 000033EB [C4340000]               <1>             dd     DSK_STATUS          ; AH = 01H; STATUS
10814 000033EF [D5340000]               <1>             dd     DSK_READ            ; AH = 02H; READ
10815 000033F3 [E6340000]               <1>             dd     DSK_WRITE           ; AH = 03H; WRITE
10816 000033F7 [F7340000]               <1>             dd     DSK_VERF            ; AH = 04H; VERIFY
10817 000033FB [08350000]               <1>             dd     DSK_FORMAT          ; AH = 05H; FORMAT
10818 000033FF [8D350000]               <1>             dd     FNC_ERR                   ; AH = 06H; INVALID
10819 00003403 [8D350000]               <1>             dd     FNC_ERR                   ; AH = 07H; INVALID
10820 00003407 [9A350000]               <1>             dd     DSK_PARMS           ; AH = 08H; READ DRIVE PARAMETERS
10821 0000340B [8D350000]               <1>             dd     FNC_ERR                   ; AH = 09H; INVALID
10822 0000340F [8D350000]               <1>             dd     FNC_ERR                   ; AH = 0AH; INVALID
10823 00003413 [8D350000]               <1>             dd     FNC_ERR                   ; AH = 0BH; INVALID
10824 00003417 [8D350000]               <1>             dd     FNC_ERR                   ; AH = 0CH; INVALID
10825 0000341B [8D350000]               <1>             dd     FNC_ERR                   ; AH = 0DH; INVALID
10826 0000341F [8D350000]               <1>             dd     FNC_ERR                   ; AH = 0EH; INVALID
10827 00003423 [8D350000]               <1>             dd     FNC_ERR                   ; AH = 0FH; INVALID
10828 00003427 [8D350000]               <1>             dd     FNC_ERR                   ; AH = 10H; INVALID
10829 0000342B [8D350000]               <1>             dd     FNC_ERR                   ; AH = 11H; INVALID
10830 0000342F [8D350000]               <1>             dd     FNC_ERR                   ; AH = 12H; INVALID
10831 00003433 [8D350000]               <1>             dd     FNC_ERR                   ; AH = 13H; INVALID
10832 00003437 [8D350000]               <1>             dd     FNC_ERR                   ; AH = 14H; INVALID
10833 0000343B [72360000]               <1>             dd     DSK_TYPE            ; AH = 15H; READ DASD TYPE
10834 0000343F [9D360000]               <1>             dd     DSK_CHANGE          ; AH = 16H; CHANGE STATUS
10835 00003443 [D7360000]               <1>             dd     FORMAT_SET          ; AH = 17H; SET DASD TYPE
10836 00003447 [5A370000]               <1>             dd     SET_MEDIA           ; AH = 18H; SET MEDIA TYPE
10837                                   <1> FNC_TAE EQU     $                          ; END
10838                                   <1>
10839                                   <1> ;--------------------------------------------------------------------------
10840                                   <1> ; DISK_RESET (AH = 00H)
10841                                   <1> ;           RESET THE DISKETTE SYSTEM.
10842                                   <1> ;
10843                                   <1> ; ON EXIT:  @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION
10844                                   <1> ;--------------------------------------------------------------------------
10845                                   <1> DSK_RESET:
10846 0000344B 66BAF203                 <1>             MOV    DX,03F2H            ; ADAPTER CONTROL PORT
10847 0000344F FA                       <1>             CLI                        ; NO INTERRUPTS
10848 00003450 A0[4E4E0100]             <1>             MOV    AL,[MOTOR_STATUS]   ; GET DIGITAL OUTPUT REGISTER REFLECTION
10849 00003455 243F                     <1>             AND    AL,00111111B        ; KEEP SELECTED AND MOTOR ON BITS
10850 00003457 C0C004                   <1>             ROL    AL,4                ; MOTOR VALUE TO HIGH NIBBLE
10851                                   <1>                                        ; DRIVE SELECT TO LOW NIBBLE
10852 0000345A 0C08                     <1>             OR     AL,00001000B        ; TURN ON INTERRUPT ENABLE
10853 0000345C EE                       <1>             OUT    DX,AL               ; RESET THE ADAPTER
10854 0000345D C605[4D4E0100]00         <1>             MOV    byte [SEEK_STATUS],0      ; SET RECALIBRATE REQUIRED ON ALL DRIVES
10855                                   <1>             ;JMP   $+2                 ; WAIT FOR I/O
10856                                   <1>             ;JMP   $+2                 ; WAIT FOR I/O (TO INSURE MINIMUM
10857                                   <1>                                        ;      PULSE WIDTH)
10858                                   <1>             ; 19/12/2014
10859                                   <1>             NEWIODELAY
10860 00003464 E6EB                     <2>  out 0ebh,al
10861                                   <1>
10862                                   <1>             ; 17/12/2014
10863                                   <1>             ; AWARD BIOS 1999 - RESETDRIVES (ADISK.ASM)
10864 00003466 B915000000               <1>             mov    ecx, WAITCPU_RESET_ON     ; cx = 21 -- Min. 14 micro seconds !?
10865                                   <1> wdw1:
10866                                   <1>             NEWIODELAY   ; 27/02/2015
10867 0000346B E6EB                     <2>  out 0ebh,al
10868 0000346D E2FC                     <1>             loop   wdw1
10869                                   <1>             ;
10870 0000346F 0C04                     <1>             OR     AL,00000100B        ; TURN OFF RESET BIT
10871 00003471 EE                       <1>             OUT    DX,AL               ; RESET THE ADAPTER
10872                                   <1>             ; 16/12/2014
10873                                   <1>             IODELAY
10874 00003472 EB00                     <2>  jmp short $+2
10875 00003474 EB00                     <2>  jmp short $+2
10876                                   <1>             ;
10877                                   <1>             ;STI                       ; ENABLE THE INTERRUPTS
10878 00003476 E83C0C0000               <1>             CALL   WAIT_INT            ; WAIT FOR THE INTERRUPT
10879 0000347B 723E                     <1>             JC     short DR_ERR        ; IF ERROR, RETURN IT
10880 0000347D 66B9C000                 <1>             MOV    CX,11000000B        ; CL = EXPECTED @NEC_STATUS
10881                                   <1> NXT_DRV:
10882 00003481 6651                     <1>             PUSH   CX                  ; SAVE FOR CALL
10883 00003483 B8[B9340000]             <1>             MOV    eAX, DR_POP_ERR     ; LOAD NEC_OUTPUT ERROR ADDRESS
10884 00003488 50                       <1>             PUSH   eAX                 ; "
10885 00003489 B408                     <1>             MOV    AH,08H              ; SENSE INTERRUPT STATUS COMMAND
10886 0000348B E81A0B0000               <1>             CALL   NEC_OUTPUT
10887 00003490 58                       <1>             POP    eAX                 ; THROW AWAY ERROR RETURN
10888 00003491 E8510C0000               <1>             CALL   RESULTS                  ; READ IN THE RESULTS
10889 00003496 6659                     <1>             POP    CX                  ; RESTORE AFTER CALL
10890 00003498 7221                     <1>             JC     short DR_ERR        ; ERROR RETURN
10891 0000349A 3A0D[514E0100]           <1>             CMP    CL, [NEC_STATUS]    ; TEST FOR DRIVE READY TRANSITION
10892 000034A0 7519                     <1>             JNZ    short DR_ERR        ; EVERYTHING OK
10893 000034A2 FEC1                     <1>             INC    CL                  ; NEXT EXPECTED @NEC_STATUS
10894 000034A4 80F9C3                   <1>             CMP    CL,11000011B        ; ALL POSSIBLE DRIVES CLEARED
10895 000034A7 76D8                     <1>             JBE    short NXT_DRV       ; FALL THRU IF 11000100B OR >
10896                                   <1>             ;
10897 000034A9 E869030000               <1>             CALL   SEND_SPEC           ; SEND SPECIFY COMMAND TO NEC
10898                                   <1> RESBAC:
10899 000034AE E81D090000               <1>             CALL   SETUP_END           ; VARIOUS CLEANUPS
10900 000034B3 6689F3                   <1>             MOV    BX,SI               ; GET SAVED AL TO BL
10901 000034B6 88D8                     <1>             MOV    AL,BL               ; PUT BACK FOR RETURN
10902 000034B8 C3                       <1>             RETn
10903                                   <1> DR_POP_ERR:
10904 000034B9 6659                     <1>             POP    CX                  ; CLEAR STACK
10905                                   <1> DR_ERR:
10906 000034BB 800D[504E0100]20         <1>             OR     byte [DSKETTE_STATUS],BAD_NEC ; SET ERROR CODE
10907 000034C2 EBEA                     <1>             JMP    SHORT RESBAC        ; RETURN FROM RESET
10908                                   <1>
10909                                   <1> ;--------------------------------------------------------------------------
10910                                   <1> ; DISK_STATUS      (AH = 01H)
10911                                   <1> ;     DISKETTE STATUS.
10912                                   <1> ;
```

```
10913                                  <1> ; ON ENTRY:  AH : STATUS OF PREVIOUS OPERATION
10914                                  <1> ;
10915                                  <1> ; ON EXIT:   AH, @DSKETTE_STATUS, CY REFLECT STATUS OF PREVIOUS OPERATION.
10916                                  <1> ;--------------------------------------------------------------------------------
10917                                  <1> DSK_STATUS:
10918 000034C4 8825[504E0100]          <1>       MOV    [DSKETTE_STATUS],AH ; PUT BACK FOR SETUP END
10919 000034CA E801090000              <1>       CALL   SETUP_END           ; VARIOUS CLEANUPS
10920 000034CF 6689F3                  <1>       MOV    BX,SI               ; GET SAVED AL TO BL
10921 000034D2 88D8                    <1>       MOV    AL,BL               ; PUT BACK FOR RETURN
10922 000034D4 C3                      <1>       RETn
10923                                  <1>
10924                                  <1> ;--------------------------------------------------------------------------------
10925                                  <1> ; DISK_READ  (AH = 02H)
10926                                  <1> ;     DISKETTE READ.
10927                                  <1> ;
10928                                  <1> ; ON ENTRY:  DI     : DRIVE #
10929                                  <1> ;            SI-HI  : HEAD #
10930                                  <1> ;            SI-LOW : # OF SECTORS
10931                                  <1> ;            ES     : BUFFER SEGMENT
10932                                  <1> ;            [BP]   : SECTOR #
10933                                  <1> ;            [BP+1] : TRACK #
10934                                  <1> ;            [BP+2] : BUFFER OFFSET
10935                                  <1> ;
10936                                  <1> ; ON EXIT:   @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION
10937                                  <1> ;--------------------------------------------------------------------------------
10938                                  <1>
10939                                  <1> ; 06/02/2015, ES:BX -> EBX (unix386.s)
10940                                  <1>
10941                                  <1> DSK_READ:
10942 000034D5 8025[4E4E0100]7F        <1>       AND    byte [MOTOR_STATUS],01111111B ; INDICATE A READ OPERATION
10943 000034DC 66B846E6                <1>       MOV    AX,0E646H           ; AX = NEC COMMAND, DMA COMMAND
10944 000034E0 E83C040000              <1>       CALL   RD_WR_VF            ; COMMON READ/WRITE/VERIFY
10945 000034E5 C3                      <1>       RETn
10946                                  <1>
10947                                  <1> ;--------------------------------------------------------------------------------
10948                                  <1> ; DISK_WRITE (AH = 03H)
10949                                  <1> ;     DISKETTE WRITE.
10950                                  <1> ;
10951                                  <1> ; ON ENTRY:  DI     : DRIVE #
10952                                  <1> ;            SI-HI  : HEAD #
10953                                  <1> ;            SI-LOW : # OF SECTORS
10954                                  <1> ;            ES     : BUFFER SEGMENT
10955                                  <1> ;            [BP]   : SECTOR #
10956                                  <1> ;            [BP+1] : TRACK #
10957                                  <1> ;            [BP+2] : BUFFER OFFSET
10958                                  <1> ;
10959                                  <1> ; ON EXIT:   @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION
10960                                  <1> ;--------------------------------------------------------------------------------
10961                                  <1>
10962                                  <1> ; 06/02/2015, ES:BX -> EBX (unix386.s)
10963                                  <1>
10964                                  <1> DSK_WRITE:
10965 000034E6 66B84AC5                <1>       MOV    AX,0C54AH           ; AX = NEC COMMAND, DMA COMMAND
10966 000034EA 800D[4E4E0100]80        <1>         OR      byte [MOTOR_STATUS],10000000B ; INDICATE WRITE OPERATION
10967 000034F1 E82B040000              <1>       CALL   RD_WR_VF            ; COMMON READ/WRITE/VERIFY
10968 000034F6 C3                      <1>       RETn
10969                                  <1>
10970                                  <1> ;--------------------------------------------------------------------------------
10971                                  <1> ; DISK_VERF  (AH = 04H)
10972                                  <1> ;     DISKETTE VERIFY.
10973                                  <1> ;
10974                                  <1> ; ON ENTRY:  DI     : DRIVE #
10975                                  <1> ;            SI-HI  : HEAD #
10976                                  <1> ;            SI-LOW : # OF SECTORS
10977                                  <1> ;            ES     : BUFFER SEGMENT
10978                                  <1> ;            [BP]   : SECTOR #
10979                                  <1> ;            [BP+1] : TRACK #
10980                                  <1> ;            [BP+2] : BUFFER OFFSET
10981                                  <1> ;
10982                                  <1> ; ON EXIT:   @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION
10983                                  <1> ;--------------------------------------------------------------------------------
10984                                  <1> DSK_VERF:
10985 000034F7 8025[4E4E0100]7F        <1>       AND    byte [MOTOR_STATUS],01111111B ; INDICATE A READ OPERATION
10986 000034FE 66B842E6                <1>       MOV    AX,0E642H           ; AX = NEC COMMAND, DMA COMMAND
10987 00003502 E81A040000              <1>       CALL   RD_WR_VF            ; COMMON READ/WRITE/VERIFY
10988 00003507 C3                      <1>       RETn
10989                                  <1>
10990                                  <1> ;--------------------------------------------------------------------------------
10991                                  <1> ; DISK_FORMAT      (AH = 05H)
10992                                  <1> ;     DISKETTE FORMAT.
10993                                  <1> ;
10994                                  <1> ; ON ENTRY:  DI     : DRIVE #
10995                                  <1> ;            SI-HI  : HEAD #
10996                                  <1> ;            SI-LOW : # OF SECTORS
10997                                  <1> ;            ES     : BUFFER SEGMENT
10998                                  <1> ;            [BP]   : SECTOR #
10999                                  <1> ;            [BP+1] : TRACK #
11000                                  <1> ;            [BP+2] : BUFFER OFFSET
11001                                  <1> ;            @DISK_POINTER POINTS TO THE PARAMETER TABLE OF THIS DRIVE
11002                                  <1> ;
11003                                  <1> ; ON EXIT:   @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION
11004                                  <1> ;--------------------------------------------------------------------------------
11005                                  <1> DSK_FORMAT:
11006 00003508 E853030000              <1>       CALL   XLAT_NEW            ; TRANSLATE STATE TO PRESENT ARCH.
11007 0000350D E84F050000              <1>       CALL   FMT_INIT            ; ESTABLISH STATE IF UNESTABLISHED
11008 00003512 800D[4E4E0100]80        <1>         OR      byte [MOTOR_STATUS], 10000000B ; INDICATE WRITE OPERATION
11009 00003519 E897050000              <1>       CALL   MED_CHANGE          ; CHECK MEDIA CHANGE AND RESET IF SO
11010 0000351E 725D                    <1>         JC      short FM_DON             ; MEDIA CHANGED, SKIP
11011 00003520 E8F2020000              <1>       CALL   SEND_SPEC           ; SEND SPECIFY COMMAND TO NEC
11012 00003525 E8FD050000              <1>       CALL   CHK_LASTRATE        ; ZF=1 ATTEMPT RATE IS SAME AS LAST RATE
11013 0000352A 7405                    <1>         JZ      short FM_WR              ; YES, SKIP SPECIFY COMMAND
11014 0000352C E8D4050000              <1>       CALL   SEND_RATE           ; SEND DATA RATE TO CONTROLLER
```

```
11015                                  <1> FM_WR:
11016 00003531 E88A060000              <1>       CALL   FMTDMA_SET          ; SET UP THE DMA FOR FORMAT
11017 00003536 7245                    <1>         JC     short FM_DON            ; RETURN WITH ERROR
11018 00003538 B44D                    <1>       MOV    AH,04DH                ; ESTABLISH THE FORMAT COMMAND
11019 0000353A E8E7060000              <1>       CALL   NEC_INIT            ; INITIALIZE THE NEC
11020 0000353F 723C                    <1>         JC     short FM_DON            ; ERROR - EXIT
11021 00003541 B8[7D350000]            <1>         MOV    eAX, FM_DON             ; LOAD ERROR ADDRESS
11022 00003546 50                      <1>       PUSH   eAX                ; PUSH NEC_OUT ERROR RETURN
11023 00003547 B203                    <1>       MOV    DL,3               ; BYTES/SECTOR VALUE TO NEC
11024 00003549 E856090000              <1>       CALL   GET_PARM
11025 0000354E E8570A0000              <1>       CALL   NEC_OUTPUT
11026 00003553 B204                    <1>       MOV    DL,4               ; SECTORS/TRACK VALUE TO NEC
11027 00003555 E84A090000              <1>       CALL   GET_PARM
11028 0000355A E84B0A0000              <1>       CALL   NEC_OUTPUT
11029 0000355F B207                    <1>       MOV    DL,7               ; GAP LENGTH VALUE TO NEC
11030 00003561 E83E090000              <1>       CALL   GET_PARM
11031 00003566 E83F0A0000              <1>       CALL   NEC_OUTPUT
11032 0000356B B208                    <1>       MOV    DL,8               ; FILLER BYTE TO NEC
11033 0000356D E832090000              <1>       CALL   GET_PARM
11034 00003572 E8330A0000              <1>       CALL   NEC_OUTPUT
11035 00003577 58                      <1>       POP    eAX                ; THROW AWAY ERROR
11036 00003578 E827070000              <1>       CALL   NEC_TERM           ; TERMINATE, RECEIVE STATUS, ETC,
11037                                  <1> FM_DON:
11038 0000357D E80F030000              <1>       CALL   XLAT_OLD           ; TRANSLATE STATE TO COMPATIBLE MODE
11039 00003582 E849080000              <1>       CALL   SETUP_END          ; VARIOUS CLEANUPS
11040 00003587 6689F3                  <1>       MOV    BX,SI              ; GET SAVED AL TO BL
11041 0000358A 88D8                    <1>       MOV    AL,BL              ; PUT BACK FOR RETURN
11042 0000358C C3                      <1>       RETn
11043                                  <1>
11044                                  <1> ;--------------------------------------------------------------------------
11045                                  <1> ; FNC_ERR
11046                                  <1> ;     INVALID FUNCTION REQUESTED OR INVALID DRIVE:
11047                                  <1> ;     SET BAD COMMAND IN STATUS.
11048                                  <1> ;
11049                                  <1> ; ON EXIT:  @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION
11050                                  <1> ;--------------------------------------------------------------------------
11051                                  <1> FNC_ERR:                       ; INVALID FUNCTION REQUEST
11052 0000358D 6689F0                  <1>       MOV    AX,SI              ; RESTORE AL
11053 00003590 B401                    <1>       MOV    AH,BAD_CMD         ; SET BAD COMMAND ERROR
11054 00003592 8825[504E0100]          <1>       MOV    [DSKETTE_STATUS],AH ; STORE IN DATA AREA
11055 00003598 F9                      <1>       STC                       ; SET CARRY INDICATING ERROR
11056 00003599 C3                      <1>       RETn
11057                                  <1>
11058                                  <1> ; 01/06/2016
11059                                  <1> ; 28/05/2016
11060                                  <1> ; 27/05/2016 - TRDOS 386 (TRDOS v.2.0)
11061                                  <1> ;--------------------------------------------------------------------------
11062                                  <1> ; DISK_PARMS (AH = 08H)
11063                                  <1> ;     READ DRIVE PARAMETERS.
11064                                  <1> ;
11065                                  <1> ; ON ENTRY:  DI : DRIVE #
11066                                  <1> ;            ; 27/05/2016
11067                                  <1> ;            EBX = Buffer Address for floppy disk parameters table (16 bytes)
11068                                  <1> ;
11069                                  <1> ; ON EXIT:   CL/[BP]   = BITS 7 & 6 HI 2 BITS OF MAX CYLINDER
11070                                  <1> ;                        BITS 0-5 MAX SECTORS/TRACK
11071                                  <1> ;            CH/[BP+1] = LOW 8 BITS OF MAX CYLINDER
11072                                  <1> ;            BL/[BP+2] = BITS 7-4 = 0
11073                                  <1> ;                        BITS 3-0 = VALID CMOS DRIVE TYPE
11074                                  <1> ;            BH/[BP+3] = 0
11075                                  <1> ;            DL/[BP+4] = # DRIVES INSTALLED (VALUE CHECKED)
11076                                  <1> ;            DH/[BP+5] = MAX HEAD #
11077                                  <1> ;          ** 27/05/2016 - TRDOS 386 (TRDOS v2.0) **
11078                                  <1> ;            ** EBX = Buffer address for floppy disk parameters table **
11079                                  <1> ;            ;DI/[BP+6] = OFFSET TO DISK_BASE
11080                                  <1> ;            ;ES        = SEGMENT OF DISK_BASE
11081                                  <1> ;
11082                                  <1> ;            AX        = 0
11083                                  <1> ;
11084                                  <1> ;            NOTE : THE ABOVE INFORMATION IS STORED IN THE USERS STACK AT
11085                                  <1> ;                   THE LOCATIONS WHERE THE MAIN ROUTINE WILL POP THEM
11086                                  <1> ;                   INTO THE APPROPRIATE REGISTERS BEFORE RETURNING TO THE
11087                                  <1> ;                   CALLER.
11088                                  <1> ;--------------------------------------------------------------------------
11089                                  <1> DSK_PARMS:
11090 0000359A E8C1020000              <1>       CALL   XLAT_NEW           ; TRANSLATE STATE TO PRESENT ARCH,
11091                                  <1>      ;       MOV    WORD [BP+2],0       ; DRIVE TYPE = 0
11092                                  <1>      ;  MOV     AX, [EQUIP_FLAG]        ; LOAD EQUIPMENT FLAG FOR # DISKETTES
11093                                  <1>      ;  AND     AL,11000001B            ; KEEP DISKETTE DRIVE BITS
11094                                  <1>      ;  MOV     DL,2                    ; DISKETTE DRIVES = 2
11095                                  <1>      ;  CMP     AL,01000001B            ; 2 DRIVES INSTALLED ?
11096                                  <1>      ;  JZ      short STO_DL            ; IF YES JUMP
11097                                  <1>      ;  DEC     DL                      ; DISKETTE DRIVES = 1
11098                                  <1>      ;  CMP     AL,00000001B            ; 1 DRIVE INSTALLED ?
11099                                  <1>      ;  JNZ     short NON_DRV           ; IF NO JUMP
11100 0000359F 29D2                    <1>       sub    edx, edx
11101 000035A1 66A1[F65C0000]          <1>       mov     ax, [fd0_type]
11102 000035A7 6621C0                  <1>       and     ax, ax
11103 000035AA 0F848A000000            <1>         jz      NON_DRV
11104 000035B0 FEC2                    <1>       inc     dl
11105 000035B2 20E4                    <1>       and     ah, ah
11106 000035B4 7402                    <1>       jz      short STO_DL
11107 000035B6 FEC2                    <1>       inc     dl
11108                                  <1> STO_DL:
11109                                  <1>       ;MOV   [BP+4],DL          ; STORE NUMBER OF DRIVES
11110 000035B8 895508                  <1>       mov    [ebp+8], edx ; 20/02/2015
11111 000035BB 6683FF01                <1>       CMP    DI,1               ; CHECK FOR VALID DRIVE
11112 000035BF 777C                    <1>       JA     short NON_DRV1            ; DRIVE INVALID
11113                                  <1>       ;MOV   BYTE [BP+5],1      ; MAXIMUM HEAD NUMBER =   1
11114 000035C1 C6450901                <1>       mov    byte [ebp+9], 1  ; 20/02/2015
11115 000035C5 E8D1080000              <1>       CALL   CMOS_TYPE          ; RETURN DRIVE TYPE IN AL
11116                                  <1>       ;;20/02/2015
```

```
11117                                  <1>       ;;JC   short CHK_EST       ; IF CMOS BAD CHECKSUM ESTABLISHED
11118                                  <1>       ;;OR   AL,AL               ; TEST FOR NO DRIVE TYPE
11119 000035CA 740F                    <1>       JZ     short CHK_EST       ; JUMP IF SO
11120 000035CC E81B020000              <1>       CALL   DR_TYPE_CHECK       ; RTN CS:BX = MEDIA/DRIVE PARAM TBL
11121 000035D1 7208                    <1>       JC     short CHK_EST       ; TYPE NOT IN TABLE (POSSIBLE BAD CMOS)
11122                                  <1>       ;MOV   [BP+2],AL           ; STORE VALID CMOS DRIVE TYPE
11123                                  <1>         ;mov [ebp+4], al ; 06/02/2015
11124 000035D3 8A4B04                  <1>       MOV     CL, [eBX+MD.SEC_TRK]     ; GET SECTOR/TRACK
11125 000035D6 8A6B0B                  <1>         MOV     CH, [eBX+MD.MAX_TRK]     ; GET MAX. TRACK NUMBER
11126 000035D9 EB36                    <1>       JMP    SHORT STO_CX        ; CMOS GOOD, USE CMOS
11127                                  <1> CHK_EST:
11128 000035DB 8AA7[5D4E0100]          <1>       MOV    AH, [DSK_STATE+eDI] ; LOAD STATE FOR THIS DRIVE
11129 000035E1 F6C410                  <1>       TEST   AH,MED_DET          ; CHECK FOR ESTABLISHED STATE
11130 000035E4 7457                    <1>       JZ     short NON_DRV1            ; CMOS BAD/INVALID OR UNESTABLISHED
11131                                  <1> USE_EST:
11132 000035E6 80E4C0                  <1>       AND    AH,RATE_MSK         ; ISOLATE STATE
11133 000035E9 80FC80                  <1>       CMP    AH,RATE_250         ; RATE 250 ?
11134 000035EC 7570                    <1>       JNE    short USE_EST2            ; NO, GO CHECK OTHER RATE
11135                                  <1>
11136                                  <1> ;-----       DATA RATE IS 250 KBS, TRY 360 KB TABLE FIRST
11137                                  <1>
11138 000035EE B001                    <1>       MOV    AL,01               ; DRIVE TYPE 1 (360KB)
11139 000035F0 E8F7010000              <1>       CALL   DR_TYPE_CHECK       ; RTN CS:BX = MEDIA/DRIVE PARAM TBL
11140 000035F5 8A4B04                  <1>         MOV     CL, [eBX+MD.SEC_TRK]     ; GET SECTOR/TRACK
11141 000035F8 8A6B0B                  <1>         MOV     CH, [eBX+MD.MAX_TRK]     ; GET MAX. TRACK NUMBER
11142 000035FB F687[5D4E0100]01        <1>       TEST   byte [DSK_STATE+eDI],TRK_CAPA ; 80 TRACK ?
11143 00003602 740D                    <1>       JZ     short STO_CX        ; MUST BE 360KB DRIVE
11144                                  <1>
11145                                  <1> ;-----       IT IS 1.44 MB DRIVE
11146                                  <1>
11147                                  <1> PARM144:
11148 00003604 B004                    <1>       MOV    AL,04               ; DRIVE TYPE 4 (1.44MB)
11149 00003606 E8E1010000              <1>       CALL   DR_TYPE_CHECK       ; RTN CS:BX = MEDIA/DRIVE PARAM TBL
11150 0000360B 8A4B04                  <1>         MOV     CL, [eBX+MD.SEC_TRK]     ; GET SECTOR/TRACK
11151 0000360E 8A6B0B                  <1>         MOV     CH, [eBX+MD.MAX_TRK]     ; GET MAX. TRACK NUMBER
11152                                  <1> STO_CX:
11153 00003611 894D00                  <1>       MOV    [eBP],eCX           ; SAVE POINTER IN STACK FOR RETURN
11154                                  <1> ES_DI:
11155                                  <1>       ;MOV   [BP+6],BX           ; ADDRESS OF MEDIA/DRIVE PARM TABLE
11156                                  <1>       ;mov   [ebp+12], ebx ; 06/02/2015
11157                                  <1>       ;MOV   AX,CS               ; SEGMENT MEDIA/DRIVE PARAMETER TABLE
11158                                  <1>       ;MOV   ES,AX               ; ES IS SEGMENT OF TABLE
11159                                  <1>       ;
11160                                  <1>       ; 28/05/2016
11161                                  <1>       ; 27/05/2016
11162                                  <1>       ; return floppy disk parameters table to user
11163                                  <1>       ; in user's buffer, which is pointed by EBX
11164                                  <1>       ;
11165 00003614 57                      <1>       push   edi
11166 00003615 8B7D04                  <1>       mov    edi, [ebp+4]               ; ebx (input), user's buffer address
11167 00003618 0FB6C0                  <1>       movzx  eax, al
11168 0000361B 894504                  <1>         mov  [ebp+4], eax   ; ebx      ; drive type (for floppy drives)
11169                                  <1>       ; 01/06/2016 (INT 33h, disk type return for floppy disks, in BL)
11170 0000361E A3[585B0100]            <1>       mov    [user_buffer], eax  ; 01/06/2016 (overwrite ebx return value)
11171                                  <1>       ;(INT 33h, Function 08h will replace user's buffer addr with disk type!)
11172                                  <1>       ;
11173 00003623 89DE                    <1>       mov    esi, ebx            ; floppy disk parameter table (16 bytes)
11174 00003625 B910000000              <1>       mov    ecx, 16 ; 16 bytes
11175 0000362A E853B20000              <1>         call    transfer_to_user_buffer ; trdosk6.s (16/05/2016)
11176 0000362F 5F                      <1>       pop    edi
11177                                  <1> DP_OUT:
11178 00003630 E85C020000              <1>       CALL   XLAT_OLD            ; TRANSLATE STATE TO COMPATIBLE MODE
11179 00003635 6631C0                  <1>       XOR    AX,AX               ; CLEAR
11180 00003638 F8                      <1>       CLC
11181 00003639 C3                      <1>       RETn
11182                                  <1>
11183                                  <1> ;-----       NO DRIYE PRESENT HANDLER
11184                                  <1>
11185                                  <1> NON_DRV:
11186                                  <1>       ;MOV   BYTE [BP+4],0       ; CLEAR NUMBER OF DRIVES
11187 0000363A 895508                  <1>       mov    [ebp+8], edx ; 0 ; 20/02/2015
11188                                  <1> NON_DRV1:
11189 0000363D 6681FF8000              <1>       CMP    DI,80H              ; CHECK FOR FIXED MEDIA TYPE REQUEST
11190 00003642 720C                    <1>       JB     short NON_DRV2            ; CONTINUE IF NOT REQUEST FALL THROUGH
11191                                  <1>
11192                                  <1> ;-----       FIXED DISK REQUEST FALL THROUGH ERROR
11193                                  <1>
11194 00003644 E848020000              <1>       CALL   XLAT_OLD            ; ELSE TRANSLATE TO COMPATIBLE MODE
11195 00003649 6689F0                  <1>       MOV    AX,SI               ; RESTORE AL
11196 0000364C B401                    <1>       MOV    AH,BAD_CMD          ; SET BAD COMMAND ERROR
11197 0000364E F9                      <1>       STC
11198 0000364F C3                      <1>       RETn
11199                                  <1>
11200                                  <1> NON_DRV2:
11201                                  <1>       ;XOR   AX,AX               ; CLEAR PARMS IF NO DRIVES OR CMOS BAD
11202 00003650 31C0                    <1>       xor    eax, eax
11203 00003652 66894500                <1>       MOV    [eBP],AX            ; TRACKS, SECTORS/TRACK = 0
11204                                  <1>       ;MOV   [BP+5],AH           ; HEAD = 0
11205 00003656 886509                  <1>       mov    [ebp+9], ah ; 06/02/2015
11206                                  <1>       ;MOV   [BP+6],AX           ; OFFSET TO DISK_BASE = 0
11207 00003659 89450C                  <1>       mov    [ebp+12], eax
11208                                  <1>       ;MOV   ES,AX               ; ES IS SEGMENT OF TABLE
11209 0000365C EBD2                    <1>       JMP    SHORT DP_OUT
11210                                  <1>
11211                                  <1> ;-----       DATA RATE IS EITHER 300 KBS OR 500 KBS, TRY 1.2 MB TABLE FIRST
11212                                  <1>
11213                                  <1> USE_EST2:
11214 0000365E B002                    <1>       MOV    AL,02               ; DRIVE TYPE 2 (1.2MB)
11215 00003660 E887010000              <1>       CALL   DR_TYPE_CHECK       ; RTN CS:BX = MEDIA/DRIVE PARAM TBL
11216 00003665 8A4B04                  <1>         MOV     CL, [eBX+MD.SEC_TRK]     ; GET SECTOR/TRACK
11217 00003668 8A6B0B                  <1>         MOV     CH, [eBX+MD.MAX_TRK]     ; GET MAX. TRACK NUMBER
11218 0000366B 80FC40                  <1>       CMP    AH,RATE_300         ; RATE 300 ?
```

```
11219 0000366E 74A1              <1>       JZ    short STO_CX        ; MUST BE 1.2MB DRIVE
11220 00003670 EB92              <1>       JMP   SHORT PARM144       ; ELSE, IT IS 1.44MB DRIVE
11221                            <1>
11222                            <1> ;--------------------------------------------------------------------------
11223                            <1> ; DISK_TYPE (AH = 15H)
11224                            <1> ;     THIS ROUTINE RETURNS THE TYPE OF MEDIA INSTALLED.
11225                            <1> ;
11226                            <1> ;  ON ENTRY: DI = DRIVE #
11227                            <1> ;
11228                            <1> ;  ON EXIT:  AH = DRIVE TYPE, CY=0
11229                            <1> ;--------------------------------------------------------------------------
11230                            <1> DSK_TYPE:
11231 00003672 E8E9010000        <1>       CALL  XLAT_NEW            ; TRANSLATE STATE TO PRESENT ARCH.
11232 00003677 8A87[5D4E0100]    <1>       MOV   AL, [DSK_STATE+eDI] ; GET PRESENT STATE INFORMATION
11233 0000367D 08C0              <1>       OR    AL,AL               ; CHECK FOR NO DRIVE
11234 0000367F 7418              <1>       JZ    short NO_DRV
11235 00003681 B401              <1>       MOV   AH,NOCHGLN          ; NO CHANGE LINE FOR 40 TRACK DRIVE
11236 00003683 A801              <1>       TEST  AL,TRK_CAPA         ; IS THIS DRIVE AN 80 TRACK DRIVE?
11237 00003685 7402              <1>       JZ    short DT_BACK             ; IF NO JUMP
11238 00003687 B402              <1>       MOV   AH,CHGLN            ; CHANGE LINE FOR 80 TRACK DRIVE
11239                            <1> DT_BACK:
11240 00003689 6650              <1>       PUSH  AX                  ; SAVE RETURN VALUE
11241 0000368B E801020000        <1>       CALL  XLAT_OLD            ; TRANSLATE STATE TO COMPATIBLE MODE
11242 00003690 6658              <1>       POP   AX                  ; RESTORE RETURN VALUE
11243 00003692 F8                <1>       CLC                       ; NO ERROR
11244 00003693 6689F3            <1>       MOV   BX,SI               ; GET SAVED AL TO BL
11245 00003696 88D8              <1>       MOV   AL,BL               ; PUT BACK FOR RETURN
11246 00003698 C3                <1>       RETn
11247                            <1> NO_DRV:
11248 00003699 30E4              <1>       XOR   AH,AH               ; NO DRIVE PRESENT OR UNKNOWN
11249 0000369B EBEC              <1>       JMP   SHORT DT_BACK
11250                            <1>
11251                            <1> ;--------------------------------------------------------------------------
11252                            <1> ; DISK_CHANGE      (AH = 16H)
11253                            <1> ;     THIS ROUTINE RETURNS THE STATE OF THE DISK CHANGE LINE.
11254                            <1> ;
11255                            <1> ; ON ENTRY:  DI = DRIVE #
11256                            <1> ;
11257                            <1> ; ON EXIT:   AH = @DSKETTE_STATUS
11258                            <1> ;                 00 - DISK CHANGE LINE INACTIVE, CY = 0
11259                            <1> ;                 06 - DISK CHANGE LINE ACTIVE, CY = 1
11260                            <1> ;--------------------------------------------------------------------------
11261                            <1> DSK_CHANGE:
11262 0000369D E8BE010000        <1>       CALL  XLAT_NEW            ; TRANSLATE STATE TO PRESENT ARCH.
11263 000036A2 8A87[5D4E0100]    <1>       MOV   AL, [DSK_STATE+eDI] ; GET MEDIA STATE INFORMATION
11264 000036A8 08C0              <1>       OR    AL,AL               ; DRIVE PRESENT ?
11265 000036AA 7422              <1>       JZ    short DC_NON        ; JUMP IF NO DRIVE
11266 000036AC A801              <1>       TEST  AL,TRK_CAPA         ; 80 TRACK DRIVE ?
11267 000036AE 7407              <1>       JZ    short SETIT         ; IF SO , CHECK CHANGE LINE
11268                            <1> DC0:
11269 000036B0 E88D0A0000        <1>         CALL    READ_DSKCHNG            ; GO CHECK STATE OF DISK CHANGE LINE
11270 000036B5 7407              <1>       JZ    short FINIS         ; CHANGE LINE NOT ACTIVE
11271                            <1>
11272 000036B7 C605[504E0100]06  <1> SETIT:      MOV   byte [DSKETTE_STATUS], MEDIA_CHANGE ; INDICATE MEDIA REMOVED
11273                            <1>
11274 000036BE E8CE010000        <1> FINIS:      CALL  XLAT_OLD            ; TRANSLATE STATE TO COMPATIBLE MODE
11275 000036C3 E808070000        <1>       CALL  SETUP_END           ; VARIOUS CLEANUPS
11276 000036C8 6689F3            <1>       MOV   BX,SI               ; GET SAVED AL TO BL
11277 000036CB 88D8              <1>       MOV   AL,BL               ; PUT BACK FOR RETURN
11278 000036CD C3                <1>       RETn
11279                            <1> DC_NON:
11280 000036CE 800D[504E0100]80  <1>       OR    byte [DSKETTE_STATUS], TIME_OUT ; SET TIMEOUT, NO DRIVE
11281 000036D5 EBE7              <1>       JMP   SHORT FINIS
11282                            <1>
11283                            <1> ;--------------------------------------------------------------------------
11284                            <1> ; FORMAT_SET(AH = 17H)
11285                            <1> ;     THIS ROUTINE IS USED TO ESTABLISH THE TYPE OF MEDIA TO BE USED
11286                            <1> ;     FOR THE FOLLOWING FORMAT OPERATION.
11287                            <1> ;
11288                            <1> ; ON ENTRY:  SI LOW = DASD TYPE FOR FORMAT
11289                            <1> ;            DI     = DRIVE #
11290                            <1> ;
11291                            <1> ; ON EXIT:   @DSKETTE_STATUS REFLECTS STATUS
11292                            <1> ;            AH = @DSKETTE_STATUS
11293                            <1> ;            CY = 1 IF ERROR
11294                            <1> ;--------------------------------------------------------------------------
11295                            <1> FORMAT_SET:
11296 000036D7 E884010000        <1>       CALL  XLAT_NEW            ; TRANSLATE STATE TO PRESENT ARCH.
11297 000036DC 6656              <1>       PUSH  SI                  ; SAVE DASD TYPE
11298 000036DE 6689F0            <1>       MOV   AX,SI               ; AH = ? , AL , DASD TYPE
11299 000036E1 30E4              <1>       XOR   AH,AH               ; AH , 0 , AL , DASD TYPE
11300 000036E3 6689C6            <1>       MOV   SI,AX               ; SI = DASD TYPE
11301 000036E6 80A7[5D4E0100]0F  <1>       AND   byte [DSK_STATE+eDI], ~(MED_DET+DBL_STEP+RATE_MSK) ; CLEAR STATE
11302 000036ED 664E              <1>       DEC   SI                  ; CHECK FOR 320/360K MEDIA & DRIVE
11303 000036EF 7509              <1>       JNZ   short NOT_320       ; BYPASS IF NOT
11304 000036F1 808F[5D4E0100]90  <1>       OR    byte [DSK_STATE+eDI], MED_DET+RATE_250 ; SET TO 320/360
11305 000036F8 EB48              <1>       JMP   SHORT S0
11306                            <1>
11307                            <1> NOT_320:
11308 000036FA E8B6030000        <1>       CALL  MED_CHANGE          ; CHECK FOR TIME_OUT
11309 000036FF 803D[504E0100]80  <1>       CMP   byte [DSKETTE_STATUS], TIME_OUT
11310 00003706 743A              <1>       JZ    short S0            ; IF TIME OUT TELL CALLER
11311                            <1> S3:
11312 00003708 664E              <1>       DEC   SI                  ; CHECK FOR 320/360K IN 1.2M DRIVE
11313 0000370A 7509              <1>       JNZ   short NOT_320_12    ; BYPASS IF NOT
11314 0000370C 808F[5D4E0100]70  <1>       OR    byte [DSK_STATE+eDI], MED_DET+DBL_STEP+RATE_300 ; SET STATE
11315 00003713 EB2D              <1>       JMP   SHORT S0
11316                            <1>
11317                            <1> NOT_320_12:
11318 00003715 664E              <1>       DEC   SI                  ; CHECK FOR 1.2M MEDIA IN 1.2M DRIVE
11319 00003717 7509              <1>       JNZ   short NOT_12        ; BYPASS IF NOT
11320 00003719 808F[5D4E0100]10  <1>       OR    byte [DSK_STATE+eDI], MED_DET+RATE_500 ; SET STATE VARIABLE
```

```
11321 00003720 EB20                <1>       JMP    SHORT S0            ; RETURN TO CALLER
11322                              <1>
11323                              <1> NOT_12:
11324 00003722 664E                <1>       DEC    SI                  ; CHECK FOR SET DASD TYPE 04
11325 00003724 752B                <1>       JNZ    short FS_ERR        ; BAD COMMAND EXIT IF NOT VALID TYPE
11326                              <1>
11327 00003726 F687[5D4E0100]04    <1>       TEST   byte [DSK_STATE+eDI], DRV_DET ; DRIVE DETERMINED ?
11328 0000372D 740B                <1>       JZ     short ASSUME        ; IF STILL NOT DETERMINED ASSUME
11329 0000372F B050                <1>       MOV    AL,MED_DET+RATE_300
11330 00003731 F687[5D4E0100]02    <1>         TEST    byte [DSK_STATE+eDI], FMT_CAPA ; MULTIPLE FORMAT CAPABILITY ?
11331 00003738 7502                <1>       JNZ    short OR_IT_IN             ; IF 1.2 M THEN DATA RATE 300
11332                              <1>
11333                              <1> ASSUME:
11334 0000373A B090                <1>       MOV    AL,MED_DET+RATE_250 ; SET UP
11335                              <1>
11336                              <1> OR_IT_IN:
11337 0000373C 0887[5D4E0100]      <1>       OR     [DSK_STATE+eDI], AL ; OR IN THE CORRECT STATE
11338                              <1> S0:
11339 00003742 E84A010000          <1>       CALL   XLAT_OLD            ; TRANSLATE STATE TO COMPATIBLE MODE
11340 00003747 E884060000          <1>       CALL   SETUP_END           ; VARIOUS CLEANUPS
11341 0000374C 665B                <1>       POP    BX                  ; GET SAVED AL TO BL
11342 0000374E 88D8                <1>       MOV    AL,BL               ; PUT BACK FOR RETURN
11343 00003750 C3                  <1>       RETn
11344                              <1>
11345                              <1> FS_ERR:
11346 00003751 C605[504E0100]01    <1>       MOV    byte [DSKETTE_STATUS], BAD_CMD ; UNKNOWN STATE,BAD COMMAND
11347 00003758 EBE8                <1>       JMP    SHORT S0
11348                              <1>
11349                              <1> ;--------------------------------------------------------------------------
11350                              <1> ; SET_MEDIA  (AH = 18H)
11351                              <1> ;     THIS ROUTINE SETS THE TYPE OF MEDIA AND DATA RATE
11352                              <1> ;     TO BE USED FOR THE FOLLOWING FORMAT OPERATION.
11353                              <1> ;
11354                              <1> ; ON ENTRY:
11355                              <1> ;     [BP]   = SECTOR PER TRACK
11356                              <1> ;     [BP+1] = TRACK #
11357                              <1> ;     DI     = DRIVE #
11358                              <1> ;
11359                              <1> ; ON EXIT:
11360                              <1> ;     @DSKETTE_STATUS REFLECTS STATUS
11361                              <1> ;     IF NO ERROR:
11362                              <1> ;           AH = 0
11363                              <1> ;           CY = 0
11364                              <1> ;           ES = SEGMENT OF MEDIA/DRIVE PARAMETER TABLE
11365                              <1> ;           DI/[BP+6] = OFFSET OF MEDIA/DRIVE PARAMETER TABLE
11366                              <1> ;     IF ERROR:
11367                              <1> ;           AH = @DSKETTE_STATUS
11368                              <1> ;           CY = 1
11369                              <1> ;--------------------------------------------------------------------------
11370                              <1> SET_MEDIA:
11371 0000375A E801010000          <1>       CALL   XLAT_NEW            ; TRANSLATE STATE TO PRESENT ARCH.
11372 0000375F F687[5D4E0100]01    <1>         TEST    byte [DSK_STATE+eDI], TRK_CAPA ; CHECK FOR CHANGE LINE AVAILABLE
11373 00003766 7415                <1>       JZ     short SM_CMOS       ; JUMP IF 40 TRACK DRIVE
11374 00003768 E848030000          <1>       CALL   MED_CHANGE          ; RESET CHANGE LINE
11375 0000376D 803D[504E0100]80    <1>       CMP    byte [DSKETTE_STATUS], TIME_OUT ; IF TIME OUT TELL CALLER
11376 00003774 746B                <1>       JE     short SM_RTN
11377 00003776 C605[504E0100]00    <1>       MOV    byte [DSKETTE_STATUS], 0 ; CLEAR STATUS
11378                              <1> SM_CMOS:
11379 0000377D E819070000          <1>       CALL   CMOS_TYPE           ; RETURN DRIVE TYPE IN (AL)
11380                              <1>       ;;20/02/2015
11381                              <1>       ;;JC   short MD_NOT_FND    ; ERROR IN CMOS
11382                              <1>       ;;OR   AL,AL               ; TEST FOR NO DRIVE
11383 00003782 745D                <1>       JZ     short SM_RTN        ; RETURN IF SO
11384 00003784 E863000000          <1>       CALL   DR_TYPE_CHECK       ; RTN CS:BX = MEDIA/DRIVE PARAM TBL
11385 00003789 7231                <1>       JC     short MD_NOT_FND    ; TYPE NOT IN TABLE (BAD CMOS)
11386 0000378B 57                  <1>       PUSH   eDI                 ; SAVE REG.
11387 0000378C 31DB                <1>       XOR    eBX,eBX                  ; BX = INDEX TO DR. TYPE TABLE
11388 0000378E B906000000          <1>       MOV    eCX,DR_CNT          ; CX = LOOP COUNT
11389                              <1> DR_SEARCH:
11390 00003793 8AA3[705C0000]      <1>       MOV    AH, [DR_TYPE+eBX]   ; GET DRIVE TYPE
11391 00003799 80E47F              <1>       AND    AH,BIT7OFF          ; MASK OUT MSB
11392 0000379C 38E0                <1>       CMP    AL,AH               ; DRIVE TYPE MATCH ?
11393 0000379E 7516                <1>       JNE    short NXT_MD        ; NO, CHECK NEXT DRIVE TYPE
11394                              <1> DR_FND:
11395 000037A0 8BBB[715C0000]      <1>       MOV    eDI, [DR_TYPE+eBX+1]      ; DI = MEDIA/DRIVE PARAM TABLE
11396                              <1> MD_SEARCH:
11397 000037A6 8A6704              <1>         MOV    AH, [eDI+MD.SEC_TRK]    ; GET SECTOR/TRACK
11398 000037A9 386500              <1>       CMP    [eBP],AH            ; MATCH?
11399 000037AC 7508                <1>       JNE    short NXT_MD        ; NO, CHECK NEXT MEDIA
11400 000037AE 8A670B              <1>         MOV    AH, [eDI+MD.MAX_TRK]    ; GET MAX. TRACK #
11401 000037B1 386501              <1>       CMP    [eBP+1],AH          ; MATCH?
11402 000037B4 740F                <1>       JE     short MD_FND        ; YES, GO GET RATE
11403                              <1> NXT_MD:
11404                              <1>       ;ADD   BX,3                ; CHECK NEXT DRIVE TYPE
11405 000037B6 83C305              <1>         add  ebx, 5 ; 18/02/2015
11406 000037B9 E2D8                <1>       LOOP    DR_SEARCH
11407 000037BB 5F                  <1>       POP    eDI                 ; RESTORE REG.
11408                              <1> MD_NOT_FND:
11409 000037BC C605[504E0100]0C    <1>       MOV    byte [DSKETTE_STATUS], MED_NOT_FND ; ERROR, MEDIA TYPE NOT FOUND
11410 000037C3 EB1C                <1>       JMP    SHORT SM_RTN        ; RETURN
11411                              <1> MD_FND:
11412 000037C5 8A470C              <1>         MOV    AL, [eDI+MD.RATE]       ; GET RATE
11413 000037C8 3C40                <1>       CMP    AL,RATE_300         ; DOUBLE STEP REQUIRED FOR RATE 300
11414 000037CA 7502                <1>       JNE    short MD_SET
11415 000037CC 0C20                <1>       OR     AL,DBL_STEP
11416                              <1> MD_SET:
11417                              <1>       ;MOV   [BP+6],DI           ; SAVE TABLE POINTER IN STACK
11418 000037CE 897D0C              <1>       mov    [ebp+12], edi ; 18/02/2015
11419 000037D1 0C10                <1>       OR     AL,MED_DET          ; SET MEDIA ESTABLISHED
11420 000037D3 5F                  <1>       POP    eDI
11421 000037D4 80A7[5D4E0100]0F    <1>       AND    byte [DSK_STATE+eDI], ~(MED_DET+DBL_STEP+RATE_MSK) ; CLEAR STATE
11422 000037DB 0887[5D4E0100]      <1>       OR     [DSK_STATE+eDI], AL
```

```
11423                                  <1>       ;MOV   AX, CS              ; SEGMENT OF MEDIA/DRIVE PARAMETER TABLE
11424                                  <1>       ;MOV   ES, AX              ; ES IS SEGMENT OF TABLE
11425                                  <1> SM_RTN:
11426 000037E1 E8AB000000             <1>       CALL   XLAT_OLD            ; TRANSLATE STATE TO COMPATIBLE MODE
11427 000037E6 E8E5050000             <1>       CALL   SETUP_END           ; VARIOUS CLEANUPS
11428 000037EB C3                     <1>       RETn
11429                                  <1>
11430                                  <1> ;--------------------------------------------------------------
11431                                  <1> ; DR_TYPE_CHECK                                          :
11432                                  <1> ;     CHECK IF THE GIVEN DRIVE TYPE IN REGISTER (AL)           :
11433                                  <1> ;     IS SUPPORTED IN BIOS DRIVE TYPE TABLE                :
11434                                  <1> ; ON ENTRY:                                        :
11435                                  <1> ;     AL = DRIVE TYPE                                      :
11436                                  <1> ; ON EXIT:                                         :
11437                                  <1> ;     CS = SEGMENT MEDIA/DRIVE PARAMETER TABLE (CODE)          :
11438                                  <1> ;     CY = 0       DRIVE TYPE SUPPORTED                        :
11439                                  <1> ;          BX = OFFSET TO MEDIA/DRIVE PARAMETER TABLE          :
11440                                  <1> ;     CY = 1 DRIVE TYPE NOT SUPPORTED                :
11441                                  <1> ; REGISTERS ALTERED: eBX                                 :
11442                                  <1> ;--------------------------------------------------------------
11443                                  <1> DR_TYPE_CHECK:
11444 000037EC 6650                   <1>       PUSH   AX
11445 000037EE 51                     <1>       PUSH   eCX
11446 000037EF 31DB                   <1>       XOR    eBX,eBX                   ; BX = INDEX TO DR_TYPE TABLE
11447 000037F1 B906000000             <1>       MOV    eCX,DR_CNT          ; CX = LOOP COUNT
11448                                  <1> TYPE_CHK:
11449 000037F6 8AA3[705C0000]         <1>       MOV    AH,[DR_TYPE+eBX]    ; GET DRIVE TYPE
11450 000037FC 38E0                   <1>       CMP    AL,AH               ; DRIVE TYPE MATCH?
11451 000037FE 740D                   <1>       JE     short DR_TYPE_VALID ; YES, RETURN WITH CARRY RESET
11452                                  <1>       ;ADD   BX,3                ; CHECK NEXT DRIVE TYPE
11453 00003800 83C305                 <1>         add  ebx, 5 ; 16/02/2015 (32 bit address modification)
11454 00003803 E2F1                   <1>       LOOP    TYPE_CHK
11455                                  <1>       ;
11456 00003805 BB[CF5C0000]           <1>       mov    ebx, MD_TBL6        ; 1.44MB fd parameter table
11457                                  <1>                                  ; Default for GET_PARM (11/12/2014)
11458                                  <1>       ;
11459 0000380A F9                     <1>       STC                        ; DRIVE TYPE NOT FOUND IN TABLE
11460 0000380B EB06                   <1>       JMP    SHORT TYPE_RTN
11461                                  <1> DR_TYPE_VALID:
11462 0000380D 8B9B[715C0000]         <1>       MOV    eBX,[DR_TYPE+eBX+1]        ; BX = MEDIA TABLE
11463                                  <1> TYPE_RTN:
11464 00003813 59                     <1>       POP    eCX
11465 00003814 6658                   <1>       POP    AX
11466 00003816 C3                     <1>       RETn
11467                                  <1>
11468                                  <1> ;--------------------------------------------------------------
11469                                  <1> ; SEND_SPEC                                          :
11470                                  <1> ;     SEND THE SPECIFY COMMAND TO CONTROLLER USING DATA FROM     :
11471                                  <1> ;     THE DRIVE PARAMETER TABLE POINTED BY @DISK_POINTER   :
11472                                  <1> ; ON ENTRY: @DISK_POINTER = DRIVE PARAMETER TABLE         :
11473                                  <1> ; ON EXIT:  NONE                                   :
11474                                  <1> ; REGISTERS ALTERED: CX, DX                              :
11475                                  <1> ;--------------------------------------------------------------
11476                                  <1> SEND_SPEC:
11477 00003817 50                     <1>       PUSH   eAX                 ; SAVE AX
11478 00003818 B8[3E380000]           <1>       MOV    eAX, SPECBAC        ; LOAD ERROR ADDRESS
11479 0000381D 50                     <1>       PUSH   eAX                 ; PUSH NEC_OUT ERROR RETURN
11480 0000381E B403                   <1>       MOV    AH,03H              ; SPECIFY COMMAND
11481 00003820 E885070000             <1>       CALL   NEC_OUTPUT          ; OUTPUT THE COMMAND
11482 00003825 28D2                   <1>       SUB    DL,DL               ; FIRST SPECIFY BYTE
11483 00003827 E878060000             <1>       CALL   GET_PARM            ; GET PARAMETER TO AH
11484 0000382C E879070000             <1>       CALL   NEC_OUTPUT          ; OUTPUT THE COMMAND
11485 00003831 B201                   <1>       MOV    DL,1                ; SECOND SPECIFY BYTE
11486 00003833 E86C060000             <1>       CALL   GET_PARM            ; GET PARAMETER TO AH
11487 00003838 E86D070000             <1>       CALL   NEC_OUTPUT          ; OUTPUT THE COMMAND
11488 0000383D 58                     <1>       POP    eAX                 ; POP ERROR RETURN
11489                                  <1> SPECBAC:
11490 0000383E 58                     <1>       POP    eAX                 ; RESTORE ORIGINAL AX VALUE
11491 0000383F C3                     <1>       RETn
11492                                  <1>
11493                                  <1> ;--------------------------------------------------------------
11494                                  <1> ; SEND_SPEC_MD                                           :
11495                                  <1> ;     SEND THE SPECIFY COMMAND TO CONTROLLER USING DATA FROM     :
11496                                  <1> ;     THE MEDIA/DRIVE PARAMETER TABLE POINTED BY (CS:BX)   :
11497                                  <1> ; ON ENTRY:  CS:BX = MEDIA/DRIVE PARAMETER TABLE          :
11498                                  <1> ; ON EXIT:   NONE                                  :
11499                                  <1> ; REGISTERS ALTERED: AX                                  :
11500                                  <1> ;--------------------------------------------------------------
11501                                  <1> SEND_SPEC_MD:
11502 00003840 50                     <1>       PUSH   eAX                 ; SAVE RATE DATA
11503 00003841 B8[5E380000]           <1>       MOV    eAX, SPEC_ESBAC           ; LOAD ERROR ADDRESS
11504 00003846 50                     <1>       PUSH   eAX                 ; PUSH NEC_OUT ERROR RETURN
11505 00003847 B403                   <1>       MOV    AH,03H              ; SPECIFY COMMAND
11506 00003849 E85C070000             <1>       CALL   NEC_OUTPUT          ; OUTPUT THE COMMAND
11507 0000384E 8A23                   <1>         MOV     AH, [eBX+MD.SPEC1]      ; GET 1ST SPECIFY BYTE
11508 00003850 E855070000             <1>       CALL   NEC_OUTPUT          ; OUTPUT THE COMMAND
11509 00003855 8A6301                 <1>         MOV     AH, [eBX+MD.SPEC2]      ; GET SECOND SPECIFY BYTE
11510 00003858 E84D070000             <1>       CALL   NEC_OUTPUT          ; OUTPUT THE COMMAND
11511 0000385D 58                     <1>       POP    eAX                 ; POP ERROR RETURN
11512                                  <1> SPEC_ESBAC:
11513 0000385E 58                     <1>       POP    eAX                 ; RESTORE ORIGINAL AX VALUE
11514 0000385F C3                     <1>       RETn
11515                                  <1>
11516                                  <1> ;--------------------------------------------------------------------------------
11517                                  <1> ; XLAT_NEW
11518                                  <1> ;     TRANSLATES DISKETTE STATE LOCATIONS FROM COMPATIBLE
11519                                  <1> ;     MODE TO NEW ARCHITECTURE.
11520                                  <1> ;
11521                                  <1> ; ON ENTRY:  DI = DRIVE #
11522                                  <1> ;--------------------------------------------------------------------------------
11523                                  <1> XLAT_NEW:
11524 00003860 83FF01                 <1>       CMP    eDI,1                     ; VALID DRIVE
```

```
11525 00003863 7725                <1>       JA     short XN_OUT               ; IF INVALID BACK
11526 00003865 80BF[5D4E0100]00    <1>       CMP    byte [DSK_STATE+eDI], 0          ; NO DRIVE ?
11527 0000386C 741D                <1>       JZ     short DO_DET               ; IF NO DRIVE ATTEMPT DETERMINE
11528 0000386E 6689F9              <1>       MOV    CX,DI                      ; CX = DRIVE NUMBER
11529 00003871 C0E102              <1>       SHL    CL,2                       ; CL = SHIFT COUNT, A=0, B=4
11530 00003874 A0[5C4E0100]        <1>       MOV    AL, [HF_CNTRL]                ; DRIVE INFORMATION
11531 00003879 D2C8                <1>       ROR    AL,CL                      ; TO LOW NIBBLE
11532 0000387B 2407                <1>       AND    AL,DRV_DET+FMT_CAPA+TRK_CAPA     ; KEEP DRIVE BITS
11533 0000387D 80A7[5D4E0100]F8    <1>         AND     byte [DSK_STATE+eDI], ~(DRV_DET+FMT_CAPA+TRK_CAPA)
11534 00003884 0887[5D4E0100]      <1>       OR     [DSK_STATE+eDI], AL        ; UPDATE DRIVE STATE
11535                              <1> XN_OUT:
11536 0000388A C3                  <1>       RETn
11537                              <1> DO_DET:
11538 0000388B E8BF080000          <1>       CALL   DRIVE_DET                  ; TRY TO DETERMINE
11539 00003890 C3                  <1>       RETn
11540                              <1>
11541                              <1> ;--------------------------------------------------------------------------
11542                              <1> ; XLAT_OLD
11543                              <1> ;     TRANSLATES DISKETTE STATE LOCATIONS FROM NEW
11544                              <1> ;     ARCHITECTURE TO COMPATIBLE MODE.
11545                              <1> ;
11546                              <1> ; ON ENTRY:  DI = DRIVE
11547                              <1> ;--------------------------------------------------------------------------
11548                              <1> XLAT_OLD:
11549 00003891 83FF01              <1>       CMP    eDI,1              ; VALID DRIVE ?
11550                              <1>         ;JA     short XO_OUT             ; IF INVALID BACK
11551 00003894 0F8786000000        <1>         ja      XO_OUT
11552 0000389A 80BF[5D4E0100]00    <1>         CMP  byte [DSK_STATE+eDI],0    ; NO DRIVE ?
11553 000038A1 747D                <1>       JZ     short XO_OUT        ; IF NO DRIVE TRANSLATE DONE
11554                              <1>
11555                              <1> ;-----       TEST FOR SAVED DRIVE INFORMATION ALREADY SET
11556                              <1>
11557 000038A3 6689F9              <1>       MOV    CX,DI              ; CX = DRIVE NUMBER
11558 000038A6 C0E102              <1>       SHL    CL,2               ; CL = SHIFT COUNT, A=0, B=4
11559 000038A9 B402                <1>       MOV    AH,FMT_CAPA        ; LOAD MULTIPLE DATA RATE BIT MASK
11560 000038AB D2CC                <1>       ROR    AH,CL              ; ROTATE BY MASK
11561 000038AD 8425[5C4E0100]      <1>       TEST   [HF_CNTRL], AH            ; MULTIPLE-DATA RATE DETERMINED ?
11562 000038B3 751C                <1>       JNZ    short SAVE_SET            ; IF SO, NO NEED TO RE-SAVE
11563                              <1>
11564                              <1> ;-----       ERASE DRIVE BITS IN @HF_CNTRL FOR THIS DRIVE
11565                              <1>
11566 000038B5 B407                <1>       MOV    AH,DRV_DET+FMT_CAPA+TRK_CAPA ; MASK TO KEEP
11567 000038B7 D2CC                <1>       ROR    AH,CL              ; FIX MASK TO KEEP
11568 000038B9 F6D4                <1>       NOT    AH                 ; TRANSLATE MASK
11569 000038BB 2025[5C4E0100]      <1>       AND    [HF_CNTRL], AH            ; KEEP BITS FROM OTHER DRIVE INTACT
11570                              <1>
11571                              <1> ;-----       ACCESS CURRENT DRIVE BITS AND STORE IN @HF_CNTRL
11572                              <1>
11573 000038C1 8A87[5D4E0100]      <1>       MOV    AL, [DSK_STATE+eDI] ; ACCESS STATE
11574 000038C7 2407                <1>       AND    AL,DRV_DET+FMT_CAPA+TRK_CAPA ; KEEP DRIVE BITS
11575 000038C9 D2C8                <1>       ROR    AL,CL              ; FIX FOR THIS DRIVE
11576 000038CB 0805[5C4E0100]      <1>       OR     [HF_CNTRL], AL            ; UPDATE SAVED DRIVE STATE
11577                              <1>
11578                              <1> ;-----       TRANSLATE TO COMPATIBILITY MODE
11579                              <1>
11580                              <1> SAVE_SET:
11581 000038D1 8AA7[5D4E0100]      <1>       MOV    AH, [DSK_STATE+eDI] ; ACCESS STATE
11582 000038D7 88E7                <1>       MOV    BH,AH              ; TO BH FOR LATER
11583 000038D9 80E4C0              <1>       AND    AH,RATE_MSK        ; KEEP ONLY RATE
11584 000038DC 80FC00              <1>       CMP    AH,RATE_500        ; RATE 500 ?
11585 000038DF 7410                <1>       JZ     short CHK_144      ; YES 1.2/1.2 OR 1.44/1.44
11586 000038E1 B001                <1>       MOV    AL,M3D1U           ; AL = 360 IN 1.2 UNESTABLISHED
11587 000038E3 80FC40              <1>       CMP    AH,RATE_300        ; RATE 300 ?
11588 000038E6 7518                <1>       JNZ    short CHK_250      ; NO, 360/360, 720/720 OR 720/1.44
11589 000038E8 F6C720              <1>       TEST   BH,DBL_STEP        ; CHECK FOR DOUBLE STEP
11590 000038EB 751F                <1>       JNZ    short TST_DET      ; MUST BE 360 IN 1.2
11591                              <1> UNKNO:
11592 000038ED B007                <1>       MOV    AL,MED_UNK         ; NONE OF THE ABOVE
11593 000038EF EB22                <1>       JMP    SHORT AL_SET       ; PROCESS COMPLETE
11594                              <1> CHK_144:
11595 000038F1 E8A5050000          <1>       CALL   CMOS_TYPE          ; RETURN DRIVE TYPE IN (AL)
11596                              <1>       ;;20/02/2015
11597                              <1>       ;;JC   short UNKNO        ; ERROR, SET 'NONE OF ABOVE'
11598 000038F6 74F5                <1>       jz     short UNKNO ;; 20/02/2015
11599 000038F8 3C02                <1>       CMP    AL,2               ; 1.2MB DRIVE ?
11600 000038FA 75F1                <1>       JNE    short UNKNO        ; NO, GO SET 'NONE OF ABOVE'
11601 000038FC B002                <1>       MOV    AL,M1D1U           ; AL = 1.2 IN 1.2 UNESTABLISHED
11602 000038FE EB0C                <1>       JMP    SHORT TST_DET
11603                              <1> CHK_250:
11604 00003900 B000                <1>       MOV    AL,M3D3U           ; AL = 360 IN 360 UNESTABLISHED
11605 00003902 80FC80              <1>       CMP    AH,RATE_250        ; RATE 250 ?
11606 00003905 75E6                <1>       JNZ    short UNKNO        ; IF SO FALL IHRU
11607 00003907 F6C701              <1>       TEST   BH,TRK_CAPA        ; 80 TRACK CAPABILITY ?
11608 0000390A 75E1                <1>       JNZ    short UNKNO        ; IF SO JUMP, FALL THRU TEST DET
11609                              <1> TST_DET:
11610 0000390C F6C710              <1>       TEST   BH,MED_DET         ; DETERMINED ?
11611 0000390F 7402                <1>       JZ     short AL_SET       ; IF NOT THEN SET
11612 00003911 0403                <1>       ADD    AL,3               ; MAKE DETERMINED/ESTABLISHED
11613                              <1> AL_SET:
11614 00003913 80A7[5D4E0100]F8    <1>       AND    byte [DSK_STATE+eDI], ~(DRV_DET+FMT_CAPA+TRK_CAPA) ; CLEAR DRIVE
11615 0000391A 0887[5D4E0100]      <1>       OR     [DSK_STATE+eDI], AL ; REPLACE WITH COMPATIBLE MODE
11616                              <1> XO_OUT:
11617 00003920 C3                  <1>       RETn
11618                              <1>
11619                              <1> ;--------------------------------------------------------------------------
11620                              <1> ; RD_WR_VF
11621                              <1> ;     COMMON READ, WRITE AND VERIFY:
11622                              <1> ;     MAIN LOOP FOR STATE RETRIES.
11623                              <1> ;
11624                              <1> ; ON ENTRY:  AH = READ/WRITE/VERIFY NEC PARAMETER
11625                              <1> ;            AL = READ/WRITE/VERIFY DMA PARAMETER
11626                              <1> ;
```

```
11627                           <1> ; ON EXIT:  @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION
11628                           <1> ;--------------------------------------------------------------------------------
11629                           <1> RD_WR_VF:
11630 00003921 6650               <1>       PUSH   AX                  ; SAVE DMA, NEC PARAMETERS
11631 00003923 E838FFFFFF         <1>       CALL   XLAT_NEW            ; TRANSLATE STATE TO PRESENT ARCH.
11632 00003928 E8F3000000         <1>       CALL   SETUP_STATE         ; INITIALIZE START AND END RATE
11633 0000392D 6658               <1>       POP    AX                  ; RESTORE READ/WRITE/VERIFY
11634                           <1> DO_AGAIN:
11635 0000392F 6650               <1>       PUSH   AX                  ; SAVE READ/WRITE/VERIFY PARAMETER
11636 00003931 E87F010000         <1>       CALL   MED_CHANGE          ; MEDIA CHANGE AND RESET IF CHANGED
11637 00003936 6658               <1>       POP    AX                  ; RESTORE READ/WRITE/VERIFY
11638 00003938 0F82C9000000       <1>         JC     RWV_END                ; MEDIA CHANGE ERROR OR TIME-OUT
11639                           <1> RWV:
11640 0000393E 6650               <1>       PUSH   AX                  ; SAVE READ/WRITE/VERIFY PARAMETER
11641 00003940 8AB7[5D4E0100]     <1>       MOV    DH, [DSK_STATE+eDI] ; GET RATE STATE OF THIS DRIVE
11642 00003946 80E6C0             <1>       AND    DH,RATE_MSK         ; KEEP ONLY RATE
11643 00003949 E84D050000         <1>       CALL   CMOS_TYPE           ; RETURN DRIVE TYPE IN AL (AL)
11644                           <1>       ;;20/02/2015
11645                           <1>       ;;JC   short RWV_ASSUME    ; ERROR IN CMOS
11646 0000394E 7451               <1>       jz     short RWV_ASSUME ; 20/02/2015
11647 00003950 3C01               <1>       CMP    AL,1                ; 40 TRACK DRIVE?
11648 00003952 750D               <1>       JNE    short RWV_1         ; NO, BYPASS CMOS VALIDITY CHECK
11649 00003954 F687[5D4E0100]01   <1>       TEST   byte [DSK_STATE+eDI], TRK_CAPA ; CHECK FOR 40 TRACK DRIVE
11650 0000395B 7413               <1>       JZ     short RWV_2         ; YES, CMOS IS CORRECT
11651 0000395D B002               <1>       MOV    AL,2                ; CHANGE TO 1.2M
11652 0000395F EB0F               <1>       JMP    SHORT RWV_2
11653                           <1> RWV_1:
11654 00003961 720D               <1>       JB     short RWV_2         ; NO DRIVE SPECIFIED, CONTINUE
11655 00003963 F687[5D4E0100]01   <1>       TEST    byte [DSK_STATE+eDI], TRK_CAPA ; IS IT REALLY 40 TRACK?
11656 0000396A 7504               <1>       JNZ    short RWV_2         ; NO, 80 TRACK
11657 0000396C B001               <1>       MOV    AL,1                ; IT IS 40 TRACK, FIX CMOS VALUE
11658 0000396E EB04               <1>       jmp    short rwv_3
11659                           <1> RWV_2:
11660 00003970 08C0               <1>       OR     AL,AL               ; TEST FOR NO DRIVE
11661 00003972 742D               <1>       JZ     short RWV_ASSUME    ; ASSUME TYPE, USE MAX TRACK
11662                           <1> rwv_3:
11663 00003974 E873FEFFFF         <1>       CALL   DR_TYPE_CHECK       ; RTN CS:BX = MEDIA/DRIVE PARAM TBL.
11664 00003979 7226               <1>       JC     short RWV_ASSUME    ; TYPE NOT IN TABLE (BAD CMOS)
11665                           <1>
11666                           <1> ;-----      SEARCH FOR MEDIA/DRIVE PARAMETER TABLE
11667                           <1>
11668 0000397B 57                 <1>       PUSH   eDI                 ; SAVE DRIVE #
11669 0000397C 31DB               <1>       XOR    eBX,eBX                  ; BX = INDEX TO DR_TYPE TABLE
11670 0000397E B906000000         <1>       MOV    eCX,DR_CNT          ; CX = LOOP COUNT
11671                           <1> RWV_DR_SEARCH:
11672 00003983 8AA3[705C0000]     <1>       MOV    AH, [DR_TYPE+eBX]   ; GET DRIVE TYPE
11673 00003989 80E47F             <1>       AND    AH,BIT7OFF          ; MASK OUT MSB
11674 0000398C 38E0               <1>       CMP    AL,AH               ; DRIVE TYPE MATCH?
11675 0000398E 750B               <1>       JNE    short RWV_NXT_MD    ; NO, CHECK NEXT DRIVE TYPE
11676                           <1> RWV_DR_FND:
11677 00003990 8BBB[715C0000]     <1>       MOV    eDI, [DR_TYPE+eBX+1]      ; DI = MEDIA/DRIVE PARAMETER TABLE
11678                           <1> RWV_MD_SEARH:
11679 00003996 3A770C             <1>         CMP     DH, [eDI+MD.RATE]       ; MATCH?
11680 00003999 741B               <1>       JE     short RWV_MD_FND    ; YES, GO GET 1ST SPECIFY BYTE
11681                           <1> RWV_NXT_MD:
11682                           <1>       ;ADD   BX,3                ; CHECK NEXT DRIVE TYPE
11683 0000399B 83C305             <1>       add    eBX, 5
11684 0000399E E2E3               <1>       LOOP   RWV_DR_SEARCH
11685 000039A0 5F                 <1>       POP    eDI                 ; RESTORE DRIVE #
11686                           <1>
11687                           <1> ;-----      ASSUME PRIMARY DRIVE IS INSTALLED AS SHIPPED
11688                           <1>
11689                           <1> RWV_ASSUME:
11690 000039A1 BB[8E5C0000]       <1>       MOV    eBX, MD_TBL1        ; POINT TO 40 TRACK 250 KBS
11691 000039A6 F687[5D4E0100]01   <1>       TEST   byte [DSK_STATE+eDI], TRK_CAPA ; TEST FOR 80 TRACK
11692 000039AD 740A               <1>       JZ     short RWV_MD_FND1   ; MUST BE 40 TRACK
11693 000039AF BB[A85C0000]       <1>       MOV    eBX, MD_TBL3        ; POINT TO 80 TRACK 500 KBS
11694 000039B4 EB03               <1>       JMP    short RWV_MD_FND1   ; GO SPECIFY PARAMTERS
11695                           <1>
11696                           <1> ;-----      CS:BX POINTS TO MEDIA/DRIVE PARAMETER TABLE
11697                           <1>
11698                           <1> RWV_MD_FND:
11699 000039B6 89FB               <1>       MOV    eBX,eDI                  ; BX = MEDIA/DRIVE PARAMETER TABLE
11700 000039B8 5F                 <1>       POP    eDI                 ; RESTORE DRIVE #
11701                           <1>
11702                           <1> ;-----      SEND THE SPECIFY COMMAND TO THE CONTROLLER
11703                           <1>
11704                           <1> RWV_MD_FND1:
11705 000039B9 E882FEFFFF         <1>       CALL   SEND_SPEC_MD
11706 000039BE E864010000         <1>       CALL   CHK_LASTRATE        ; ZF=1 ATTEMP RATE IS SAME AS LAST RATE
11707 000039C3 7405               <1>       JZ     short RWV_DBL       ; YES,SKIP SEND RATE COMMAND
11708 000039C5 E83B010000         <1>       CALL   SEND_RATE           ; SEND DATA RATE TO NEC
11709                           <1> RWV_DBL:
11710 000039CA 53                 <1>       PUSH   eBX                 ; SAVE MEDIA/DRIVE PARAM TBL ADDRESS
11711 000039CB E822040000         <1>       CALL   SETUP_DBL           ; CHECK FOR DOUBLE STEP
11712 000039D0 5B                 <1>       POP    eBX                 ; RESTORE ADDRESS
11713 000039D1 7226               <1>       JC     short CHK_RET       ; ERROR FROM READ ID, POSSIBLE RETRY
11714 000039D3 6658               <1>       POP    AX                  ; RESTORE NEC, DMA COMMAND
11715 000039D5 6650               <1>       PUSH   AX                  ; SAVE NEC COMMAND
11716 000039D7 53                 <1>       PUSH   eBX                 ; SAVE MEDIA/DRIVE PARAM TBL ADDRESS
11717 000039D8 E861010000         <1>       CALL   DMA_SETUP           ; SET UP THE DMA
11718 000039DD 5B                 <1>       POP    eBX
11719 000039DE 6658               <1>       POP    AX                  ; RESTORE NEC COMMAND
11720 000039E0 722F               <1>       JC     short RWV_BAC       ; CHECK FOR DMA BOUNDARY ERROR
11721 000039E2 6650               <1>       PUSH   AX                  ; SAVE NEC COMMAND
11722 000039E4 53                 <1>       PUSH   eBX                 ; SAVE MEDIA/DRIVE PARAM TBL ADDRESS
11723 000039E5 E83C020000         <1>       CALL   NEC_INIT            ; INITIALIZE NEC
11724 000039EA 5B                 <1>       POP    eBX                 ; RESTORE ADDRESS
11725 000039EB 720C               <1>       JC     short CHK_RET       ; ERROR - EXIT
11726 000039ED E866020000         <1>       CALL   RWV_COM                  ; OP CODE COMMON TO READ/WRITE/VERIFY
11727 000039F2 7205               <1>       JC     short CHK_RET       ; ERROR - EXIT
11728 000039F4 E8AB020000         <1>       CALL   NEC_TERM            ; TERMINATE, GET STATUS, ETC.
```

```
11729                                  <1> CHK_RET:
11730 000039F9 E84A030000              <1>       CALL   RETRY              ; CHECK FOR, SETUP RETRY
11731 000039FE 6658                    <1>       POP    AX                 ; RESTORE READ/WRITE/VERIFY PARAMETER
11732 00003A00 7305                    <1>       JNC    short RWV_END      ; CY = 0 NO RETRY
11733 00003A02 E928FFFFFF              <1>         JMP     DO_AGAIN               ; CY = 1 MEANS RETRY
11734                                  <1> RWV_END:
11735 00003A07 E8F4020000              <1>       CALL   DSTATE             ; ESTABLISH STATE IF SUCCESSFUL
11736 00003A0C E887030000              <1>       CALL   NUM_TRANS          ; AL = NUMBER TRANSFERRED
11737                                  <1> RWV_BAC:                      ; BAD DMA ERROR ENTRY
11738 00003A11 6650                    <1>       PUSH   AX                 ; SAVE NUMBER TRANSFERRED
11739 00003A13 E879FEFFFF              <1>       CALL   XLAT_OLD           ; TRANSLATE STATE TO COMPATIBLE MODE
11740 00003A18 6658                    <1>       POP    AX                 ; RESTORE NUMBER TRANSFERRED
11741 00003A1A E8B1030000              <1>       CALL   SETUP_END          ; VARIOUS CLEANUPS
11742 00003A1F C3                      <1>       RETn
11743                                  <1>
11744                                  <1> ;--------------------------------------------------------------------------
11745                                  <1> ; SETUP_STATE:     INITIALIZES START AND END RATES.
11746                                  <1> ;--------------------------------------------------------------------------
11747                                  <1> SETUP_STATE:
11748 00003A20 F687[5D4E0100]10        <1>       TEST   byte [DSK_STATE+eDI], MED_DET ; MEDIA DETERMINED ?
11749 00003A27 7537                    <1>       JNZ    short J1C          ; NO STATES IF DETERMINED
11750 00003A29 66B84000                <1>         MOV     AX,(RATE_500*256)+RATE_300  ; AH = START RATE, AL = END RATE
11751 00003A2D F687[5D4E0100]04        <1>       TEST   byte [DSK_STATE+eDI],DRV_DET ; DRIVE ?
11752 00003A34 740D                    <1>       JZ     short AX_SET        ; DO NOT KNOW DRIVE
11753 00003A36 F687[5D4E0100]02        <1>       TEST   byte [DSK_STATE+eDI], FMT_CAPA ; MULTI-RATE?
11754 00003A3D 7504                    <1>       JNZ    short AX_SET        ; JUMP IF YES
11755 00003A3F 66B88080                <1>         MOV     AX,RATE_250*257         ; START A END RATE 250 FOR 360 DRIVE
11756                                  <1> AX_SET:
11757 00003A43 80A7[5D4E0100]1F        <1>       AND    byte [DSK_STATE+eDI], ~(RATE_MSK+DBL_STEP) ; TURN OFF THE RATE
11758 00003A4A 08A7[5D4E0100]          <1>       OR     [DSK_STATE+eDI], AH ; RATE FIRST TO TRY
11759 00003A50 8025[584E0100]F3        <1>       AND    byte [LASTRATE], ~STRT_MSK ; ERASE LAST TO TRY RATE BITS
11760 00003A57 C0C804                  <1>       ROR    AL,4                ; TO OPERATION LAST RATE LOCATION
11761 00003A5A 0805[584E0100]          <1>       OR     [LASTRATE], AL            ; LAST RATE
11762                                  <1> J1C:
11763 00003A60 C3                      <1>       RETn
11764                                  <1>
11765                                  <1> ;--------------------------------------------------------------------------
11766                                  <1> ;  FMT_INIT: ESTABLISH STATE IF UNESTABLISHED AT FORMAT TIME.
11767                                  <1> ;--------------------------------------------------------------------------
11768                                  <1> FMT_INIT:
11769 00003A61 F687[5D4E0100]10        <1>       TEST   byte [DSK_STATE+eDI], MED_DET ; IS MEDIA ESTABLISHED
11770 00003A68 7546                    <1>       JNZ    short F1_OUT        ; IF SO RETURN
11771 00003A6A E82C040000              <1>       CALL   CMOS_TYPE           ; RETURN DRIVE TYPE IN AL
11772                                  <1>       ;; 20/02/2015
11773                                  <1>       ;;JC   short CL_DRV        ; ERROR IN CMOS ASSUME NO DRIVE
11774 00003A6F 7440                    <1>       jz     short CL_DRV ;; 20/02/2015
11775 00003A71 FEC8                    <1>       DEC    AL                  ; MAKE ZERO ORIGIN
11776                                  <1>       ;;JS   short CL_DRV        ; NO DRIVE IF AL 0
11777 00003A73 8AA7[5D4E0100]          <1>       MOV    AH, [DSK_STATE+eDI] ; AH = CURRENT STATE
11778 00003A79 80E40F                  <1>       AND    AH, ~(MED_DET+DBL_STEP+RATE_MSK) ; CLEAR
11779 00003A7C 08C0                    <1>       OR     AL,AL               ; CHECK FOR 360
11780 00003A7E 7505                    <1>       JNZ    short N_360         ; IF 360 WILL BE 0
11781 00003A80 80CC90                  <1>       OR     AH,MED_DET+RATE_250 ; ESTABLISH MEDIA
11782 00003A83 EB25                    <1>       JMP    SHORT SKP_STATE           ; SKIP OTHER STATE PROCESSING
11783                                  <1> N_360:
11784 00003A85 FEC8                    <1>       DEC    AL                  ; 1.2 M DRIVE
11785 00003A87 7505                    <1>       JNZ    short N_12          ; JUMP IF NOT
11786                                  <1> F1_RATE:
11787 00003A89 80CC10                  <1>       OR     AH,MED_DET+RATE_500 ; SET FORMAT RATE
11788 00003A8C EB1C                    <1>       JMP    SHORT SKP_STATE           ; SKIP OTHER STATE PROCESSING
11789                                  <1> N_12:
11790 00003A8E FEC8                    <1>       DEC    AL                  ; CHECK FOR TYPE 3
11791 00003A90 750F                    <1>       JNZ    short N_720         ; JUMP IF NOT
11792 00003A92 F6C404                  <1>       TEST   AH,DRV_DET          ; IS DRIVE DETERMINED
11793 00003A95 7410                    <1>       JZ     short ISNT_12       ; TREAT AS NON 1.2 DRIVE
11794 00003A97 F6C402                  <1>       TEST   AH,FMT_CAPA         ; IS 1.2M
11795 00003A9A 740B                    <1>       JZ     short ISNT_12       ; JUMP IF NOT
11796 00003A9C 80CC50                  <1>       OR     AH,MED_DET+RATE_300 ; RATE 300
11797 00003A9F EB09                    <1>       JMP    SHORT SKP_STATE           ; CONTINUE
11798                                  <1> N_720:
11799 00003AA1 FEC8                    <1>       DEC    AL                  ; CHECK FOR TYPE 4
11800 00003AA3 750C                    <1>       JNZ    short CL_DRV        ; NO DRIVE, CMOS BAD
11801 00003AA5 EBE2                    <1>       JMP    SHORT F1_RATE
11802                                  <1> ISNT_12:
11803 00003AA7 80CC90                  <1>       OR     AH,MED_DET+RATE_250 ; MUST BE RATE 250
11804                                  <1>
11805                                  <1> SKP_STATE:
11806 00003AAA 88A7[5D4E0100]          <1>       MOV    [DSK_STATE+eDI], AH ; STORE AWAY
11807                                  <1> F1_OUT:
11808 00003AB0 C3                      <1>       RETn
11809                                  <1> CL_DRV:
11810 00003AB1 30E4                    <1>       XOR    AH,AH               ; CLEAR STATE
11811 00003AB3 EBF5                    <1>       JMP    SHORT SKP_STATE           ; SAVE IT
11812                                  <1>
11813                                  <1> ;--------------------------------------------------------------------------
11814                                  <1> ; MED_CHANGE
11815                                  <1> ;     CHECKS FOR MEDIA CHANGE, RESETS MEDIA CHANGE,
11816                                  <1> ;     CHECKS MEDIA CHANGE AGAIN.
11817                                  <1> ;
11818                                  <1> ; ON EXIT:   CY = 1 MEANS MEDIA CHANGE OR TIMEOUT
11819                                  <1> ;            @DSKETTE_STATUS = ERROR CODE
11820                                  <1> ;--------------------------------------------------------------------------
11821                                  <1> MED_CHANGE:
11822 00003AB5 E888060000              <1>       CALL   READ_DSKCHNG        ; READ DISK CHANCE LINE STATE
11823 00003ABA 7447                    <1>       JZ     short MC_OUT        ; BYPASS HANDLING DISK CHANGE LINE
11824 00003ABC 80A7[5D4E0100]EF        <1>       AND    byte [DSK_STATE+eDI], ~MED_DET ; CLEAR STATE FOR THIS DRIVE
11825                                  <1>
11826                                  <1> ;     THIS SEQUENCE ENSURES WHENEVER A DISKETTE IS CHANGED THAT
11827                                  <1> ;     ON THE NEXT OPERATION THE REQUIRED MOTOR START UP TIME WILL
11828                                  <1> ;     BE WAITED. (DRIVE MOTOR MAY GO OFF UPON DOOR OPENING).
11829                                  <1>
11830 00003AC3 6689F9                  <1>       MOV    CX,DI               ; CL = DRIVE 0
```

```
11831 00003AC6 B001                <1>       MOV   AL,1                ; MOTOR ON BIT MASK
11832 00003AC8 D2E0                <1>       SHL   AL,CL               ; TO APPROPRIATE POSITION
11833 00003ACA F6D0                <1>       NOT   AL                  ; KEEP ALL BUT MOTOR ON
11834 00003ACC FA                  <1>       CLI                       ; NO INTERRUPTS
11835 00003ACD 2005[4E4E0100]      <1>       AND   [MOTOR_STATUS], AL  ; TURN MOTOR OFF INDICATOR
11836 00003AD3 FB                  <1>       STI                       ; INTERRUPTS ENABLED
11837 00003AD4 E810040000          <1>       CALL  MOTOR_ON            ; TURN MOTOR ON
11838                              <1>
11839                              <1> ;-----      THIS SEQUENCE OF SEEKS IS USED TO RESET DISKETTE CHANGE SIGNAL
11840                              <1>
11841 00003AD9 E86DF9FFFF          <1>       CALL  DSK_RESET           ; RESET NEC
11842 00003ADE B501                <1>       MOV   CH,01H              ; MOVE TO CYLINDER 1
11843 00003AE0 E8FF040000          <1>       CALL  SEEK                ; ISSUE SEEK
11844 00003AE5 30ED                <1>       XOR   CH,CH               ; MOVE TO CYLINDER 0
11845 00003AE7 E8F8040000          <1>       CALL  SEEK                ; ISSUE SEEK
11846 00003AEC C605[504E0100]06    <1>       MOV   byte [DSKETTE_STATUS], MEDIA_CHANGE ; STORE IN STATUS
11847                              <1> OK1:
11848 00003AF3 E84A060000          <1>       CALL  READ_DSKCHNG        ; CHECK MEDIA CHANGED AGAIN
11849 00003AF8 7407                <1>       JZ    short OK2           ; IF ACTIVE, NO DISKETTE, TIMEOUT
11850                              <1> OK4:
11851 00003AFA C605[504E0100]80    <1>       MOV   byte [DSKETTE_STATUS], TIME_OUT ; TIMEOUT IF DRIVE EMPTY
11852                              <1> OK2:
11853 00003B01 F9                  <1>       STC                       ; MEDIA CHANGED, SET CY
11854 00003B02 C3                  <1>       RETn
11855                              <1> MC_OUT:
11856 00003B03 F8                  <1>       CLC                       ; NO MEDIA CHANGED, CLEAR CY
11857 00003B04 C3                  <1>       RETn
11858                              <1>
11859                              <1> ;--------------------------------------------------------------------------
11860                              <1> ; SEND_RATE
11861                              <1> ;     SENDS DATA RATE COMMAND TO NEC
11862                              <1> ; ON ENTRY:  DI = DRIVE #
11863                              <1> ; ON EXIT:   NONE
11864                              <1> ; REGISTERS ALTERED: DX
11865                              <1> ;--------------------------------------------------------------------------
11866                              <1> SEND_RATE:
11867 00003B05 6650                <1>       PUSH  AX                  ; SAVE REG.
11868 00003B07 8025[584E0100]3F    <1>       AND   byte [LASTRATE], ~SEND_MSK ; ELSE CLEAR LAST RATE ATTEMPTED
11869 00003B0E 8A87[5D4E0100]      <1>       MOV   AL, [DSK_STATE+eDI] ; GET RATE STATE OF THIS DRIVE
11870 00003B14 24C0                <1>       AND   AL,SEND_MSK         ; KEEP ONLY RATE BITS
11871 00003B16 0805[584E0100]      <1>       OR    [LASTRATE], AL            ; SAVE NEW RATE FOR NEXT CHECK
11872 00003B1C C0C002              <1>       ROL   AL,2                ; MOVE TO BIT OUTPUT POSITIONS
11873 00003B1F 66BAF703            <1>       MOV   DX,03F7H            ; OUTPUT NEW DATA RATE
11874 00003B23 EE                  <1>       OUT   DX,AL
11875 00003B24 6658                <1>       POP   AX                  ; RESTORE REG.
11876 00003B26 C3                  <1>       RETn
11877                              <1>
11878                              <1> ;--------------------------------------------------------------------------
11879                              <1> ; CHK_LASTRATE
11880                              <1> ;     CHECK PREVIOUS DATE RATE SNT TO THE CONTROLLER.
11881                              <1> ; ON ENTRY:
11882                              <1> ;     DI = DRIVE #
11883                              <1> ; ON EXIT:
11884                              <1> ;     ZF =  1 DATA RATE IS THE SAME AS THE LAST RATE SENT TO NEC
11885                              <1> ;     ZF =  0 DATA RATE IS DIFFERENT FROM LAST RATE
11886                              <1> ; REGISTERS ALTERED: DX
11887                              <1> ;--------------------------------------------------------------------------
11888                              <1> CHK_LASTRATE:
11889 00003B27 6650                <1>       PUSH  AX                  ; SAVE REG
11890 00003B29 2225[584E0100]      <1>       AND   AH, [LASTRATE]            ; GET LAST DATA RATE SELECTED
11891 00003B2F 8A87[5D4E0100]      <1>       MOV   AL, [DSK_STATE+eDI] ; GET RATE STATE OF THIS DRIVE
11892 00003B35 6625C0C0            <1>         AND     AX, SEND_MSK*257        ; KEEP ONLY RATE BITS OF BOTH
11893 00003B39 38E0                <1>       CMP   AL, AH              ; COMPARE TO PREVIOUSLY TRIED
11894                              <1>                                 ; ZF = 1 RATE IS THE SAME
11895 00003B3B 6658                <1>       POP   AX                  ; RESTORE REG.
11896 00003B3D C3                  <1>       RETn
11897                              <1>
11898                              <1> ;--------------------------------------------------------------------------
11899                              <1> ; DMA_SETUP
11900                              <1> ;     THIS ROUTINE SETS UP THE DMA FOR READ/WRITE/VERIFY OPERATIONS.
11901                              <1> ;
11902                              <1> ; ON ENTRY:  AL = DMA COMMAND
11903                              <1> ;
11904                              <1> ; ON EXIT:   @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION
11905                              <1> ;--------------------------------------------------------------------------
11906                              <1>
11907                              <1> ; SI = Head #, # of Sectors or DASD Type
11908                              <1>
11909                              <1> ; 22/08/2015
11910                              <1> ; 08/02/2015 - Protected Mode Modification
11911                              <1> ; 06/02/2015 - 07/02/2015
11912                              <1> ; NOTE: Buffer address must be in 1st 16MB of Physical Memory (24 bit limit).
11913                              <1> ; (DMA Addres = Physical Address)
11914                              <1> ; (Retro UNIX 386 v1 Kernel/System Mode Virtual Address = Physical Address)
11915                              <1> ;
11916                              <1>
11917                              <1>
11918                              <1> ; 04/02/2016 (clc)
11919                              <1> ; 20/02/2015 modification (source: AWARD BIOS 1999, DMA_SETUP)
11920                              <1> ; 16/12/2014 (IODELAY)
11921                              <1>
11922                              <1> DMA_SETUP:
11923                              <1>
11924                              <1> ;; 20/02/2015
11925 00003B3E 8B5504              <1>       mov   edx, [ebp+4]        ; Buffer address
11926 00003B41 F7C2000000FF        <1>       test  edx, 0FF000000h           ; 16 MB limit (22/08/2015, bugfix)
11927 00003B47 756E                <1>       jnz   short dma_bnd_err_stc
11928                              <1>       ;
11929 00003B49 6650                <1>       push  ax                  ; DMA command
11930 00003B4B 52                  <1>       push  edx                 ; *
11931 00003B4C B203                <1>       mov   dl, 3               ; GET BYTES/SECTOR PARAMETER
11932 00003B4E E851030000          <1>       call  GET_PARM            ;
```

```
11933 00003B53 88E1              <1>       mov    cl, ah                  ; SHIFT COUNT (0=128, 1=256, 2=512 ETC)
11934 00003B55 6689F0            <1>       mov    ax, si             ; Sector count
11935 00003B58 88C4              <1>       mov    ah, al             ; AH =  # OF SECTORS
11936 00003B5A 28C0              <1>       sub    al, al             ; AL = 0, AX = # SECTORS * 256
11937 00003B5C 66D1E8            <1>       shr    ax, 1              ; AX = # SECTORS * 128
11938 00003B5F 66D3E0            <1>       shl    ax, cl             ; SHIFT BY PARAMETER VALUE
11939 00003B62 6648              <1>       dec    ax                 ; -1 FOR DMA VALUE
11940 00003B64 6689C1            <1>       mov    cx, ax
11941 00003B67 5A                <1>       pop    edx                ; *
11942 00003B68 6658              <1>       pop    ax
11943 00003B6A 3C42              <1>       cmp    al, 42h
11944 00003B6C 7507              <1>         jne     short NOT_VERF
11945 00003B6E BA0000FF00        <1>       mov    edx, 0FF0000h
11946 00003B73 EB08              <1>       jmp    short J33
11947                            <1> NOT_VERF:
11948 00003B75 6601CA            <1>       add    dx, cx             ; check for overflow
11949 00003B78 723E              <1>       jc     short dma_bnd_err
11950                            <1>       ;
11951 00003B7A 6629CA            <1>       sub    dx, cx             ; Restore start address
11952                            <1> J33:
11953 00003B7D FA                <1>       CLI                       ; DISABLE INTERRUPTS DURING DMA SET-UP
11954 00003B7E E60C              <1>       OUT    DMA+12,AL          ; SET THE FIRST/LA5T F/F
11955                            <1>       IODELAY                        ; WAIT FOR I/O
11956 00003B80 EB00              <2>  jmp short $+2
11957 00003B82 EB00              <2>  jmp short $+2
11958 00003B84 E60B              <1>       OUT    DMA+11,AL          ; OUTPUT THE MODE BYTE
11959 00003B86 89D0              <1>       mov    eax, edx           ; Buffer address
11960 00003B88 E604              <1>       OUT    DMA+4,AL           ; OUTPUT LOW ADDRESS
11961                            <1>       IODELAY                        ; WAIT FOR I/O
11962 00003B8A EB00              <2>  jmp short $+2
11963 00003B8C EB00              <2>  jmp short $+2
11964 00003B8E 88E0              <1>       MOV    AL,AH
11965 00003B90 E604              <1>       OUT    DMA+4,AL           ; OUTPUT HIGH ADDRESS
11966 00003B92 C1E810            <1>       shr    eax, 16
11967                            <1>       IODELAY                        ; I/O WAIT STATE
11968 00003B95 EB00              <2>  jmp short $+2
11969 00003B97 EB00              <2>  jmp short $+2
11970 00003B99 E681              <1>       OUT    081H,AL                 ; OUTPUT highest BITS TO PAGE REGISTER
11971                            <1>       IODELAY
11972 00003B9B EB00              <2>  jmp short $+2
11973 00003B9D EB00              <2>  jmp short $+2
11974 00003B9F 6689C8            <1>       mov    ax, cx             ; Byte count - 1
11975 00003BA2 E605              <1>       OUT    DMA+5,AL           ; LOW BYTE OF COUNT
11976                            <1>       IODELAY                        ; WAIT FOR I/O
11977 00003BA4 EB00              <2>  jmp short $+2
11978 00003BA6 EB00              <2>  jmp short $+2
11979 00003BA8 88E0              <1>       MOV    AL, AH
11980 00003BAA E605              <1>       OUT    DMA+5,AL           ; HIGH BYTE OF COUNT
11981                            <1>       IODELAY
11982 00003BAC EB00              <2>  jmp short $+2
11983 00003BAE EB00              <2>  jmp short $+2
11984 00003BB0 FB                <1>       STI                       ; RE-ENABLE INTERRUPTS
11985 00003BB1 B002              <1>       MOV    AL, 2              ; MODE FOR 8237
11986 00003BB3 E60A              <1>       OUT    DMA+10, AL         ; INITIALIZE THE DISKETTE CHANNEL
11987                            <1>
11988 00003BB5 F8                <1>       clc    ; 04/02/2016
11989 00003BB6 C3                <1>       retn
11990                            <1>
11991                            <1> dma_bnd_err_stc:
11992 00003BB7 F9                <1>       stc
11993                            <1> dma_bnd_err:
11994 00003BB8 C605[504E0100]09  <1>       MOV    byte [DSKETTE_STATUS], DMA_BOUNDARY ; SET ERROR
11995 00003BBF C3                <1>       RETn                      ; CY SET BY ABOVE IF ERROR
11996                            <1>
11997                            <1> ;; 16/12/2014
11998                            <1> ;;    CLI                       ; DISABLE INTERRUPTS DURING DMA SET-UP
11999                            <1> ;;    OUT    DMA+12,AL          ; SET THE FIRST/LA5T F/F
12000                            <1> ;;    ;JMP   $+2                ; WAIT FOR I/O
12001                            <1> ;;    IODELAY
12002                            <1> ;;    OUT    DMA+11,AL          ; OUTPUT THE MODE BYTE
12003                            <1> ;;    ;SIODELAY
12004                            <1> ;;      ;CMP AL, 42H                  ; DMA VERIFY COMMAND
12005                            <1> ;;      ;JNE short NOT_VERF           ; NO
12006                            <1> ;;      ;XOR AX, AX             ; START ADDRESS
12007                            <1> ;;      ;JMP SHORT J33
12008                            <1> ;;;NOT_VERF:
12009                            <1> ;;    ;MOV   AX,ES              ; GET THE ES VALUE
12010                            <1> ;;    ;ROL   AX,4               ; ROTATE LEFT
12011                            <1> ;;    ;MOV   CH,AL              ; GET HIGHEST NIBBLE OF ES TO CH
12012                            <1> ;;    ;AND   AL,11110000B       ; ZERO THE LOW NIBBLE FROM SEGMENT
12013                            <1> ;;    ;ADD   AX,[BP+2]          ; TEST FOR CARRY FROM ADDITION
12014                            <1> ;;    mov    eax, [ebp+4] ; 06/02/2015
12015                            <1> ;;    ;JNC   short J33
12016                            <1> ;;    ;INC   CH                 ; CARRY MEANS HIGH 4 BITS MUST BE INC
12017                            <1> ;;;J33:
12018                            <1> ;;    PUSH   eAX                ; SAVE START ADDRESS
12019                            <1> ;;    OUT    DMA+4,AL           ; OUTPUT LOW ADDRESS
12020                            <1> ;;    ;JMP   $+2                ; WAIT FOR I/O
12021                            <1> ;;    IODELAY
12022                            <1> ;;    MOV    AL,AH
12023                            <1> ;;    OUT    DMA+4,AL           ; OUTPUT HIGH ADDRESS
12024                            <1> ;;    shr    eax, 16           ; 07/02/2015
12025                            <1> ;;    ;MOV   AL,CH              ; GET HIGH 4 BITS
12026                            <1> ;;    ;JMP   $+2                ; I/O WAIT STATE
12027                            <1> ;;    IODELAY
12028                            <1> ;;    ;AND   AL,00001111B
12029                            <1> ;;    OUT    081H,AL                 ; OUTPUT HIGH 4 BITS TO PAGE REGISTER
12030                            <1> ;;    ;SIODELAY
12031                            <1> ;;
12032                            <1> ;;;----- DETERMINE COUNT
12033                            <1> ;;    sub    eax, eax ; 08/02/2015
12034                            <1> ;;    MOV    AX, SI             ; AL =  # OF SECTORS
```

```
12035                                  <1> ;;    XCHG   AL, AH             ; AH =  # OF SECTORS
12036                                  <1> ;;    SUB    AL, AL             ; AL = 0, AX = # SECTORS * 256
12037                                  <1> ;;    SHR    AX, 1              ; AX = # SECTORS * 128
12038                                  <1> ;;    PUSH   AX                 ; SAVE # OF SECTORS * 128
12039                                  <1> ;;    MOV    DL, 3              ; GET BYTES/SECTOR PARAMETER
12040                                  <1> ;;    CALL   GET_PARM           ; "
12041                                  <1> ;;    MOV    CL,AH              ; SHIFT COUNT (0=128, 1=256, 2=512 ETC)
12042                                  <1> ;;    POP    AX                 ; AX = # SECTORS * 128
12043                                  <1> ;;    SHL    AX,CL              ; SHIFT BY PARAMETER VALUE
12044                                  <1> ;;    DEC    AX                 ; -1 FOR DMA VALUE
12045                                  <1> ;;    PUSH   eAX  ; 08/02/2015  ; SAVE COUNT VALUE
12046                                  <1> ;;    OUT    DMA+5,AL           ; LOW BYTE OF COUNT
12047                                  <1> ;;    ;JMP   $+2                ; WAIT FOR I/O
12048                                  <1> ;;    IODELAY
12049                                  <1> ;;    MOV    AL, AH
12050                                  <1> ;;    OUT    DMA+5,AL           ; HIGH BYTE OF COUNT
12051                                  <1> ;;    ;IODELAY
12052                                  <1> ;;    STI                       ; RE-ENABLE INTERRUPTS
12053                                  <1> ;;    POP    eCX  ; 08/02/2015  ; RECOVER COUNT VALUE
12054                                  <1> ;;    POP    eAX  ; 08/02/2015  ; RECOVER ADDRESS VALUE
12055                                  <1> ;;    ;ADD   AX, CX             ; ADD, TEST FOR 64K OVERFLOW
12056                                  <1> ;;    add    ecx, eax ; 08/02/2015
12057                                  <1> ;;    MOV    AL, 2              ; MODE FOR 8237
12058                                  <1> ;;    ;JMP   $+2                ; WAIT FOR I/O
12059                                  <1> ;;    SIODELAY
12060                                  <1> ;;    OUT    DMA+10, AL         ; INITIALIZE THE DISKETTE CHANNEL
12061                                  <1> ;;    ;JNC   short NO_BAD       ; CHECK FOR ERROR
12062                                  <1> ;;    jc     short dma_bnd_err ; 08/02/2015
12063                                  <1> ;;    and    ecx, 0FFF00000h ; 16 MB limit
12064                                  <1> ;;    jz     short NO_BAD
12065                                  <1> ;;dma_bnd_err:
12066                                  <1> ;;    MOV    byte [DSKETTE_STATUS], DMA_BOUNDARY ; SET ERROR
12067                                  <1> ;;NO_BAD:
12068                                  <1> ;;    RETn                      ; CY SET BY ABOVE IF ERROR
12069                                  <1>
12070                                  <1> ;--------------------------------------------------------------------------------
12071                                  <1> ; FMTDMA_SET
12072                                  <1> ;     THIS ROUTINE SETS UP THE DMA CONTROLLER FOR A FORMAT OPERATION.
12073                                  <1> ;
12074                                  <1> ; ON ENTRY:  NOTHING REQUIRED
12075                                  <1> ;
12076                                  <1> ; ON EXIT:  @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION
12077                                  <1> ;--------------------------------------------------------------------------------
12078                                  <1>
12079                                  <1> FMTDMA_SET:
12080                                  <1> ;; 20/02/2015 modification
12081 00003BC0 8B5504                  <1>       mov    edx, [ebp+4]       ; Buffer address
12082 00003BC3 F7C20000F0FF            <1>       test   edx, 0FFF00000h           ; 16 MB limit
12083 00003BC9 75EC                    <1>       jnz    short dma_bnd_err_stc
12084                                  <1>       ;
12085 00003BCB 6652                    <1>       push   dx                 ; *
12086 00003BCD B204                    <1>       mov    DL, 4              ; SECTORS/TRACK VALUE IN PARM TABLE
12087 00003BCF E8D0020000              <1>       call   GET_PARM           ; "
12088 00003BD4 88E0                    <1>       mov    al, ah             ; AL = SECTORS/TRACK VALUE
12089 00003BD6 28E4                    <1>       sub    ah, ah             ; AX = SECTORS/TRACK VALUE
12090 00003BD8 66C1E002                <1>       shl    ax, 2              ; AX = SEC/TRK * 4 (OFFSET C,H,R,N)
12091 00003BDC 6648                    <1>       dec    ax                 ; -1 FOR DMA VALUE
12092 00003BDE 6689C1                  <1>       mov    cx, ax
12093 00003BE1 665A                    <1>       pop    dx                 ; *
12094 00003BE3 6601CA                  <1>       add    dx, cx             ; check for overflow
12095 00003BE6 72D0                    <1>       jc     short dma_bnd_err
12096                                  <1>       ;
12097 00003BE8 6629CA                  <1>       sub    dx, cx             ; Restore start address
12098                                  <1>       ;
12099 00003BEB B04A                    <1>       MOV    AL, 04AH           ; WILL WRITE TO THE DISKETTE
12100 00003BED FA                      <1>       CLI                       ; DISABLE INTERRUPTS DURING DMA SET-UP
12101 00003BEE E60C                    <1>       OUT    DMA+12,AL          ; SET THE FIRST/LA5T F/F
12102                                  <1>       IODELAY                         ; WAIT FOR I/O
12103 00003BF0 EB00                    <2>  jmp short $+2
12104 00003BF2 EB00                    <2>  jmp short $+2
12105 00003BF4 E60B                    <1>       OUT    DMA+11,AL          ; OUTPUT THE MODE BYTE
12106 00003BF6 89D0                    <1>       mov    eax, edx           ; Buffer address
12107 00003BF8 E604                    <1>       OUT    DMA+4,AL           ; OUTPUT LOW ADDRESS
12108                                  <1>       IODELAY                         ; WAIT FOR I/O
12109 00003BFA EB00                    <2>  jmp short $+2
12110 00003BFC EB00                    <2>  jmp short $+2
12111 00003BFE 88E0                    <1>       MOV    AL,AH
12112 00003C00 E604                    <1>       OUT    DMA+4,AL           ; OUTPUT HIGH ADDRESS
12113 00003C02 C1E810                  <1>       shr    eax, 16
12114                                  <1>       IODELAY                         ; I/O WAIT STATE
12115 00003C05 EB00                    <2>  jmp short $+2
12116 00003C07 EB00                    <2>  jmp short $+2
12117 00003C09 E681                    <1>       OUT    081H,AL                  ; OUTPUT highest BITS TO PAGE REGISTER
12118                                  <1>       IODELAY
12119 00003C0B EB00                    <2>  jmp short $+2
12120 00003C0D EB00                    <2>  jmp short $+2
12121 00003C0F 6689C8                  <1>       mov    ax, cx             ; Byte count - 1
12122 00003C12 E605                    <1>       OUT    DMA+5,AL           ; LOW BYTE OF COUNT
12123                                  <1>       IODELAY                         ; WAIT FOR I/O
12124 00003C14 EB00                    <2>  jmp short $+2
12125 00003C16 EB00                    <2>  jmp short $+2
12126 00003C18 88E0                    <1>       MOV    AL, AH
12127 00003C1A E605                    <1>       OUT    DMA+5,AL           ; HIGH BYTE OF COUNT
12128                                  <1>       IODELAY
12129 00003C1C EB00                    <2>  jmp short $+2
12130 00003C1E EB00                    <2>  jmp short $+2
12131 00003C20 FB                      <1>       STI                       ; RE-ENABLE INTERRUPTS
12132 00003C21 B002                    <1>       MOV    AL, 2              ; MODE FOR 8237
12133 00003C23 E60A                    <1>       OUT    DMA+10, AL         ; INITIALIZE THE DISKETTE CHANNEL
12134 00003C25 C3                      <1>       retn
12135                                  <1>
12136                                  <1> ;; 08/02/2015 - Protected Mode Modification
```

```
12137                                  <1> ;;    MOV   AL, 04AH            ; WILL WRITE TO THE DISKETTE
12138                                  <1> ;;    CLI                       ; DISABLE INTERRUPTS DURING DMA SET-UP
12139                                  <1> ;;    OUT   DMA+12,AL           ; SET THE FIRST/LA5T F/F
12140                                  <1> ;;    ;JMP  $+2                 ; WAIT FOR I/O
12141                                  <1> ;;    IODELAY
12142                                  <1> ;;    OUT   DMA+11,AL           ; OUTPUT THE MODE BYTE
12143                                  <1> ;;    ;MOV  AX,ES               ; GET THE ES VALUE
12144                                  <1> ;;    ;ROL  AX,4                ; ROTATE LEFT
12145                                  <1> ;;    ;MOV  CH,AL               ; GET HIGHEST NIBBLE OF ES TO CH
12146                                  <1> ;;    ;AND  AL,11110000B        ; ZERO THE LOW NIBBLE FROM SEGMENT
12147                                  <1> ;;    ;ADD  AX,[BP+2]           ; TEST FOR CARRY FROM ADDITION
12148                                  <1> ;;    ;JNC  short J33A
12149                                  <1> ;;    ;INC  CH                  ; CARRY MEANS HIGH 4 BITS MUST BE INC
12150                                  <1> ;;    mov   eax, [ebp+4] ; 08/02/2015
12151                                  <1> ;;;J33A:
12152                                  <1> ;;    PUSH  eAX ; 08/02/2015    ; SAVE START ADDRESS
12153                                  <1> ;;    OUT   DMA+4,AL            ; OUTPUT LOW ADDRESS
12154                                  <1> ;;    ;JMP  $+2                 ; WAIT FOR I/O
12155                                  <1> ;;    IODELAY
12156                                  <1> ;;    MOV   AL,AH
12157                                  <1> ;;    OUT   DMA+4,AL            ; OUTPUT HIGH ADDRESS
12158                                  <1> ;;    shr   eax, 16 ; 08/02/2015
12159                                  <1> ;;    ;MOV  AL,CH               ; GET HIGH 4 BITS
12160                                  <1> ;;    ;JMP  $+2                 ; I/O WAIT STATE
12161                                  <1> ;;    IODELAY
12162                                  <1> ;;    ;AND  AL,00001111B
12163                                  <1> ;;    OUT   081H,AL                   ; OUTPUT HIGH 4 BITS TO PAGE REGISTER
12164                                  <1> ;;
12165                                  <1> ;;;----- DETERMINE COUNT
12166                                  <1> ;;    sub   eax, eax ; 08/02/2015
12167                                  <1> ;;    MOV   DL, 4               ; SECTORS/TRACK VALUE IN PARM TABLE
12168                                  <1> ;;    CALL  GET_PARM            ; "
12169                                  <1> ;;    XCHG  AL, AH              ; AL = SECTORS/TRACK VALUE
12170                                  <1> ;;    SUB   AH, AH              ; AX = SECTORS/TRACK VALUE
12171                                  <1> ;;    SHL   AX, 2               ; AX = SEC/TRK * 4 (OFFSET C,H,R,N)
12172                                  <1> ;;    DEC   AX                  ; -1 FOR DMA VALUE
12173                                  <1> ;;    PUSH  eAX   ; 08/02/2015  ; SAVE # OF BYTES TO BE TRANSFERED
12174                                  <1> ;;    OUT   DMA+5,AL            ; LOW BYTE OF COUNT
12175                                  <1> ;;    ;JMP  $+2                 ; WAIT FOR I/O
12176                                  <1> ;;    IODELAY
12177                                  <1> ;;    MOV   AL, AH
12178                                  <1> ;;    OUT   DMA+5,AL            ; HIGH BYTE OF COUNT
12179                                  <1> ;;    STI                       ; RE-ENABLE INTERRUPTS
12180                                  <1> ;;    POP   eCX   ; 08/02/2015  ; RECOVER COUNT VALUE
12181                                  <1> ;;    POP   eAX   ; 08/02/2015  ; RECOVER ADDRESS VALUE
12182                                  <1> ;;    ;ADD  AX, CX              ; ADD, TEST FOR 64K OVERFLOW
12183                                  <1> ;;    add   ecx, eax ; 08/02/2015
12184                                  <1> ;;    MOV   AL, 2               ; MODE FOR 8237
12185                                  <1> ;;    ;JMP  $+2                 ; WAIT FOR I/O
12186                                  <1> ;;    SIODELAY
12187                                  <1> ;;    OUT   DMA+10, AL          ; INITIALIZE THE DISKETTE CHANNEL
12188                                  <1> ;;    ;JNC  short FMTDMA_OK           ; CHECK FOR ERROR
12189                                  <1> ;;    jc    short fmtdma_bnd_err ; 08/02/2015
12190                                  <1> ;;    and   ecx, 0FFF00000h  ; 16 MB limit
12191                                  <1> ;;    jz    short FMTDMA_OK
12192                                  <1> ;;    stc   ; 20/02/2015
12193                                  <1> ;;fmtdma_bnd_err:
12194                                  <1> ;;    MOV   byte [DSKETTE_STATUS], DMA_BOUNDARY ; SET ERROR
12195                                  <1> ;;FMTDMA_OK:
12196                                  <1> ;;    RETn                      ; CY SET BY ABOVE IF ERROR
12197                                  <1>
12198                                  <1> ;--------------------------------------------------------------------------
12199                                  <1> ; NEC_INIT
12200                                  <1> ;     THIS ROUTINE SEEKS TO THE REQUESTED TRACK AND INITIALIZES
12201                                  <1> ;     THE NEC FOR THE READ/WRITE/VERIFY/FORMAT OPERATION.
12202                                  <1> ;
12203                                  <1> ; ON ENTRY:  AH = NEC COMMAND TO BE PERFORMED
12204                                  <1> ;
12205                                  <1> ; ON EXIT:   @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION
12206                                  <1> ;--------------------------------------------------------------------------
12207                                  <1> NEC_INIT:
12208 00003C26 6650                   <1>       PUSH  AX                  ; SAVE NEC COMMAND
12209 00003C28 E8BC020000             <1>       CALL  MOTOR_ON            ; TURN MOTOR ON FOR SPECIFIC DRIVE
12210                                  <1>
12211                                  <1> ;-----      DO THE SEEK OPERATION
12212                                  <1>
12213 00003C2D 8A6D01                 <1>       MOV   CH,[eBP+1]          ; CH = TRACK #
12214 00003C30 E8AF030000             <1>       CALL  SEEK                ; MOVE TO CORRECT TRACK
12215 00003C35 6658                   <1>       POP   AX                  ; RECOVER COMMAND
12216 00003C37 721E                   <1>       JC    short ER_1          ; ERROR ON SEEK
12217 00003C39 BB[573C0000]           <1>       MOV   eBX, ER_1           ; LOAD ERROR ADDRESS
12218 00003C3E 53                     <1>       PUSH  eBX                 ; PUSH NEC_OUT ERROR RETURN
12219                                  <1>
12220                                  <1> ;-----      SEND OUT THE PARAMETERS TO THE CONTROLLER
12221                                  <1>
12222 00003C3F E866030000             <1>       CALL  NEC_OUTPUT          ; OUTPUT THE OPERATION COMMAND
12223 00003C44 6689F0                 <1>       MOV   AX,SI               ; AH = HEAD #
12224 00003C47 89FB                   <1>       MOV   eBX,eDI                   ; BL = DRIVE #
12225 00003C49 C0E402                 <1>       SAL   AH,2                ; MOVE IT TO BIT 2
12226 00003C4C 80E404                 <1>       AND   AH,00000100B        ; ISOLATE THAT BIT
12227 00003C4F 08DC                   <1>       OR    AH,BL               ; OR IN THE DRIVE NUMBER
12228 00003C51 E854030000             <1>       CALL  NEC_OUTPUT          ; FALL THRU CY SET IF ERROR
12229 00003C56 5B                     <1>       POP   eBX                 ; THROW AWAY ERROR RETURN
12230                                  <1> ER_1:
12231 00003C57 C3                     <1>       RETn
12232                                  <1>
12233                                  <1> ;--------------------------------------------------------------------------
12234                                  <1> ; RWV_COM
12235                                  <1> ;     THIS ROUTINE SENDS PARAMETERS TO THE NEC SPECIFIC TO THE
12236                                  <1> ;     READ/WRITE/VERIFY OPERATIONS.
12237                                  <1> ;
12238                                  <1> ; ON ENTRY:  CS:BX = ADDRESS OF MEDIA/DRIVE PARAMETER TABLE
```

```
12239                                  <1> ; ON EXIT:  @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION
12240                                  <1> ;--------------------------------------------------------------------------------
12241                                  <1> RWV_COM:
12242 00003C58 B8[A33C0000]            <1>       MOV   eAX, ER_2           ; LOAD ERROR ADDRESS
12243 00003C5D 50                      <1>       PUSH  eAX                 ; PUSH NEC_OUT ERROR RETURN
12244 00003C5E 8A6501                  <1>       MOV   AH,[eBP+1]          ; OUTPUT TRACK #
12245 00003C61 E844030000              <1>       CALL  NEC_OUTPUT
12246 00003C66 6689F0                  <1>       MOV   AX,SI               ; OUTPUT HEAD #
12247 00003C69 E83C030000              <1>       CALL  NEC_OUTPUT
12248 00003C6E 8A6500                  <1>         MOV     AH,[eBP]              ; OUTPUT SECTOR #
12249 00003C71 E834030000              <1>       CALL  NEC_OUTPUT
12250 00003C76 B203                    <1>       MOV   DL,3                ; BYTES/SECTOR PARAMETER FROM BLOCK
12251 00003C78 E827020000              <1>       CALL  GET_PARM            ; ... TO THE NEC
12252 00003C7D E828030000              <1>       CALL  NEC_OUTPUT          ; OUTPUT TO CONTROLLER
12253 00003C82 B204                    <1>       MOV   DL,4                ; EOT PARAMETER FROM BLOCK
12254 00003C84 E81B020000              <1>       CALL  GET_PARM            ; ... TO THE NEC
12255 00003C89 E81C030000              <1>       CALL  NEC_OUTPUT          ; OUTPUT TO CONTROLLER
12256 00003C8E 8A6305                  <1>         MOV     AH, [eBX+MD.GAP]        ; GET GAP LENGTH
12257                                  <1> _R15:
12258 00003C91 E814030000              <1>       CALL  NEC_OUTPUT
12259 00003C96 B206                    <1>       MOV   DL,6                ; DTL PARAMETER PROM BLOCK
12260 00003C98 E807020000              <1>       CALL  GET_PARM            ;  TO THE NEC
12261 00003C9D E808030000              <1>       CALL  NEC_OUTPUT          ; OUTPUT TO CONTROLLER
12262 00003CA2 58                      <1>       POP   eAX                 ; THROW AWAY ERROR EXIT
12263                                  <1> ER_2:
12264 00003CA3 C3                      <1>       RETn
12265                                  <1>
12266                                  <1> ;--------------------------------------------------------------------------------
12267                                  <1> ; NEC_TERM
12268                                  <1> ;     THIS ROUTINE WAITS FOR THE OPERATION THEN ACCEPTS THE STATUS
12269                                  <1> ;     FROM THE NEC FOR THE READ/WRITE/VERIFY/FORWAT OPERATION.
12270                                  <1> ;
12271                                  <1> ; ON EXIT:  @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION
12272                                  <1> ;--------------------------------------------------------------------------------
12273                                  <1> NEC_TERM:
12274                                  <1>
12275                                  <1> ;-----      LET THE OPERATION HAPPEN
12276                                  <1>
12277 00003CA4 56                      <1>       PUSH  eSI                 ; SAVE HEAD #, # OF SECTORS
12278 00003CA5 E80D040000              <1>       CALL  WAIT_INT            ; WAIT FOR THE INTERRUPT
12279 00003CAA 9C                      <1>       PUSHF
12280 00003CAB E837040000              <1>       CALL  RESULTS                   ; GET THE NEC STATUS
12281 00003CB0 724B                    <1>       JC    short SET_END_POP
12282 00003CB2 9D                      <1>       POPF
12283 00003CB3 723E                    <1>       JC    short SET_END       ; LOOK FOR ERROR
12284                                  <1>
12285                                  <1> ;-----      CHECK THE RESULTS RETURNED BY THE CONTROLLER
12286                                  <1>
12287 00003CB5 FC                      <1>       CLD                       ; SET THE CORRECT DIRECTION
12288 00003CB6 BE[514E0100]            <1>       MOV   eSI, NEC_STATUS           ; POINT TO STATUS FIELD
12289 00003CBB AC                      <1>       lodsb                     ; GET ST0
12290 00003CBC 24C0                    <1>       AND   AL,11000000B        ; TEST FOR NORMAL TERMINATION
12291 00003CBE 7433                    <1>       JZ    short SET_END
12292 00003CC0 3C40                    <1>       CMP   AL,01000000B        ; TEST FOR ABNORMAL TERMINATION
12293 00003CC2 7527                    <1>       JNZ   short J18           ; NOT ABNORMAL, BAD NEC
12294                                  <1>
12295                                  <1> ;-----      ABNORMAL TERMINATION, FIND OUT WHY
12296                                  <1>
12297 00003CC4 AC                      <1>       lodsb                     ; GET ST1
12298 00003CC5 D0E0                    <1>       SAL   AL,1                ; TEST FOR EDT FOUND
12299 00003CC7 B404                    <1>       MOV   AH,RECORD_NOT_FND
12300 00003CC9 7222                    <1>       JC    short J19
12301 00003CCB C0E002                  <1>       SAL   AL,2
12302 00003CCE B410                    <1>       MOV   AH,BAD_CRC
12303 00003CD0 721B                    <1>       JC    short J19
12304 00003CD2 D0E0                    <1>       SAL   AL,1                ; TEST FOR DMA OVERRUN
12305 00003CD4 B408                    <1>       MOV   AH,BAD_DMA
12306 00003CD6 7215                    <1>       JC    short J19
12307 00003CD8 C0E002                  <1>       SAL   AL,2                ; TEST FOR RECORD NOT FOUND
12308 00003CDB B404                    <1>       MOV   AH,RECORD_NOT_FND
12309 00003CDD 720E                    <1>       JC    short J19
12310 00003CDF D0E0                    <1>       SAL   AL,1
12311 00003CE1 B403                    <1>       MOV   AH,WRITE_PROTECT    ; TEST FOR WRITE_PROTECT
12312 00003CE3 7208                    <1>       JC    short J19
12313 00003CE5 D0E0                    <1>       SAL   AL,1                ; TEST MISSING ADDRESS MARK
12314 00003CE7 B402                    <1>       MOV   AH,BAD_ADDR_MARK
12315 00003CE9 7202                    <1>       JC    short J19
12316                                  <1>
12317                                  <1> ;-----      NEC MUST HAVE FAILED
12318                                  <1> J18:
12319 00003CEB B420                    <1>       MOV   AH,BAD_NEC
12320                                  <1> J19:
12321 00003CED 0825[504E0100]          <1>       OR    [DSKETTE_STATUS], AH
12322                                  <1> SET_END:
12323 00003CF3 803D[504E0100]01        <1>       CMP   byte [DSKETTE_STATUS], 1 ; SET ERROR CONDITION
12324 00003CFA F5                      <1>       CMC
12325 00003CFB 5E                      <1>       POP   eSI
12326 00003CFC C3                      <1>       RETn                      ; RESTORE HEAD #, # OF SECTORS
12327                                  <1>
12328                                  <1> SET_END_POP:
12329 00003CFD 9D                      <1>       POPF
12330 00003CFE EBF3                    <1>       JMP   SHORT SET_END
12331                                  <1>
12332                                  <1> ;--------------------------------------------------------------------------------
12333                                  <1> ; DSTATE:   ESTABLISH STATE UPON SUCCESSFUL OPERATION.
12334                                  <1> ;--------------------------------------------------------------------------------
12335                                  <1> DSTATE:
12336 00003D00 803D[504E0100]00        <1>       CMP   byte [DSKETTE_STATUS],0   ; CHECK FOR ERROR
12337 00003D07 753E                    <1>       JNZ   short SETBAC            ; IF ERROR JUMP
12338 00003D09 808F[5D4E0100]10        <1>       OR    byte [DSK_STATE+eDI],MED_DET ; NO ERROR, MARK MEDIA AS DETERMINED
12339 00003D10 F687[5D4E0100]04        <1>       TEST  byte [DSK_STATE+eDI],DRV_DET ; DRIVE DETERMINED ?
12340 00003D17 752E                    <1>       JNZ   short SETBAC        ; IF DETERMINED NO TRY TO DETERMINE
```

```
12341 00003D19 8A87[5D4E0100]      <1>       MOV    AL,[DSK_STATE+eDI]  ; LOAD STATE
12342 00003D1F 24C0                <1>       AND    AL,RATE_MSK         ; KEEP ONLY RATE
12343 00003D21 3C80                <1>       CMP    AL,RATE_250         ; RATE 250 ?
12344 00003D23 751B                <1>       JNE    short M_12          ; NO, MUST BE 1.2M OR 1.44M DRIVE
12345                              <1>
12346                              <1> ;-----      CHECK IF IT IS 1.44M
12347                              <1>
12348 00003D25 E871010000          <1>       CALL   CMOS_TYPE           ; RETURN DRIVE TYPE IN (AL)
12349                              <1>       ;;20/02/2015
12350                              <1>       ;;JC   short M_12          ; CMOS BAD
12351 00003D2A 7414                <1>       jz     short M_12 ;; 20/02/2015
12352 00003D2C 3C04                <1>       CMP    AL, 4               ; 1.44MB DRIVE ?
12353 00003D2E 7410                <1>       JE     short M_12          ; YES
12354                              <1> M_720:
12355 00003D30 80A7[5D4E0100]FD    <1>       AND    byte [DSK_STATE+eDI], ~FMT_CAPA ; TURN OFF FORMAT CAPABILITY
12356 00003D37 808F[5D4E0100]04    <1>       OR     byte [DSK_STATE+eDI],DRV_DET  ; MARK DRIVE DETERMINED
12357 00003D3E EB07                <1>       JMP    SHORT SETBAC        ; BACK
12358                              <1> M_12:
12359 00003D40 808F[5D4E0100]06    <1>       OR     byte [DSK_STATE+eDI],DRV_DET+FMT_CAPA
12360                              <1>                                  ; TURN ON DETERMINED & FMT CAPA
12361                              <1> SETBAC:
12362 00003D47 C3                  <1>       RETn
12363                              <1>
12364                              <1> ;--------------------------------------------------------------------------
12365                              <1> ; RETRY
12366                              <1> ;     DETERMINES WHETHER A RETRY IS NECESSARY.
12367                              <1> ;     IF RETRY IS REQUIRED THEN STATE INFORMATION IS UPDATED FOR RETRY.
12368                              <1> ;
12369                              <1> ; ON EXIT:   CY = 1 FOR RETRY, CY = 0 FOR NO RETRY
12370                              <1> ;--------------------------------------------------------------------------
12371                              <1> RETRY:
12372 00003D48 803D[504E0100]00    <1>       CMP    byte [DSKETTE_STATUS],0   ; GET STATUS OF OPERATION
12373 00003D4F 7445                <1>       JZ     short NO_RETRY            ; SUCCESSFUL OPERATION
12374 00003D51 803D[504E0100]80    <1>       CMP    byte [DSKETTE_STATUS],TIME_OUT ; IF TIME OUT NO RETRY
12375 00003D58 743C                <1>       JZ     short NO_RETRY
12376 00003D5A 8AA7[5D4E0100]      <1>       MOV    AH,[DSK_STATE+eDI]  ; GET MEDIA STATE OF DRIVE
12377 00003D60 F6C410              <1>       TEST   AH,MED_DET          ; ESTABLISHED/DETERMINED ?
12378 00003D63 7531                <1>       JNZ    short NO_RETRY            ; IF ESTABLISHED STATE THEN TRUE ERROR
12379 00003D65 80E4C0              <1>       AND    AH,RATE_MSK         ; ISOLATE RATE
12380 00003D68 8A2D[584E0100]      <1>       MOV    CH,[LASTRATE]       ; GET START OPERATION STATE
12381 00003D6E C0C504              <1>       ROL    CH,4                ; TO CORRESPONDING BITS
12382 00003D71 80E5C0              <1>       AND    CH,RATE_MSK         ; ISOLATE RATE BITS
12383 00003D74 38E5                <1>       CMP    CH,AH               ; ALL RATES TRIED
12384 00003D76 741E                <1>       JE     short NO_RETRY            ; IF YES, THEN TRUE ERROR
12385                              <1>
12386                              <1> ;     SETUP STATE INDICATOR FOR RETRY ATTEMPT TO NEXT RATE
12387                              <1> ;      00000000B (500) -> 10000000B    (250)
12388                              <1> ;      10000000B (250) -> 01000000B    (300)
12389                              <1> ;      01000000B (300) -> 00000000B    (500)
12390                              <1>
12391 00003D78 80FC01              <1>       CMP    AH,RATE_500+1       ; SET CY FOR RATE 500
12392 00003D7B D0DC                <1>       RCR    AH,1                ; TO NEXT STATE
12393 00003D7D 80E4C0              <1>       AND    AH,RATE_MSK         ; KEEP ONLY RATE BITS
12394 00003D80 80A7[5D4E0100]1F    <1>       AND    byte [DSK_STATE+eDI], ~(RATE_MSK+DBL_STEP)
12395                              <1>                                  ; RATE, DBL STEP OFF
12396 00003D87 08A7[5D4E0100]      <1>       OR     [DSK_STATE+eDI],AH  ; TURN ON NEW RATE
12397 00003D8D C605[504E0100]00    <1>       MOV    byte [DSKETTE_STATUS],0   ; RESET STATUS FOR RETRY
12398 00003D94 F9                  <1>       STC                        ; SET CARRY FOR RETRY
12399 00003D95 C3                  <1>       RETn                       ; RETRY RETURN
12400                              <1>
12401                              <1> NO_RETRY:
12402 00003D96 F8                  <1>       CLC                        ; CLEAR CARRY NO RETRY
12403 00003D97 C3                  <1>       RETn                       ; NO RETRY RETURN
12404                              <1>
12405                              <1> ;--------------------------------------------------------------------------
12406                              <1> ; NUM_TRANS
12407                              <1> ;     THIS ROUTINE CALCULATES THE NUMBER OF SECTORS THAT WERE
12408                              <1> ;     ACTUALLY TRANSFERRED TO/FROM THE DISKETTE.
12409                              <1> ;
12410                              <1> ; ON ENTRY:  [BP+1] = TRACK
12411                              <1> ;            SI-HI  = HEAD
12412                              <1> ;            [BP]   = START SECTOR
12413                              <1> ;
12414                              <1> ; ON EXIT:   AL = NUMBER ACTUALLY TRANSFERRED
12415                              <1> ;--------------------------------------------------------------------------
12416                              <1> NUM_TRANS:
12417 00003D98 30C0                <1>       XOR    AL,AL               ; CLEAR FOR ERROR
12418 00003D9A 803D[504E0100]00    <1>       CMP    byte [DSKETTE_STATUS],0   ; CHECK FOR ERROR
12419 00003DA1 752C                <1>       JNZ    NT_OUT              ; IF ERROR 0 TRANSFERRED
12420 00003DA3 B204                <1>       MOV    DL,4                ; SECTORS/TRACK OFFSET TO DL
12421 00003DA5 E8FA000000          <1>       CALL   GET_PARM            ; AH = SECTORS/TRACK
12422 00003DAA 8A1D[564E0100]      <1>       MOV    BL, [NEC_STATUS+5]  ; GET ENDING SECTOR
12423 00003DB0 6689F1              <1>       MOV    CX,SI               ; CH = HEAD # STARTED
12424 00003DB3 3A2D[554E0100]      <1>       CMP    CH, [NEC_STATUS+4]  ; GET HEAD ENDED UP ON
12425 00003DB9 750D                <1>       JNZ    DIF_HD              ; IF ON SAME HEAD, THEN NO ADJUST
12426 00003DBB 8A2D[544E0100]      <1>       MOV    CH, [NEC_STATUS+3]  ; GET TRACK ENDED UP ON
12427 00003DC1 3A6D01              <1>       CMP    CH,[eBP+1]          ; IS IT ASKED FOR TRACK
12428 00003DC4 7404                <1>       JZ     short SAME_TRK           ; IF SAME TRACK NO INCREASE
12429 00003DC6 00E3                <1>       ADD    BL,AH               ; ADD SECTORS/TRACK
12430                              <1> DIF_HD:
12431 00003DC8 00E3                <1>       ADD    BL,AH               ; ADD SECTORS/TRACK
12432                              <1> SAME_TRK:
12433 00003DCA 2A5D00              <1>       SUB    BL,[eBP]            ; SUBTRACT START FROM END
12434 00003DCD 88D8                <1>       MOV    AL,BL               ; TO AL
12435                              <1> NT_OUT:
12436 00003DCF C3                  <1>       RETn
12437                              <1>
12438                              <1> ;--------------------------------------------------------------------------
12439                              <1> ; SETUP_END
12440                              <1> ;     RESTORES @MOTOR_COUNT TO PARAMETER PROVIDED IN TABLE
12441                              <1> ;     AND LOADS @DSKETTE_STATUS TO AH, AND SETS CY.
12442                              <1> ;
```

```
12443                                  <1> ; ON EXIT:
12444                                  <1> ;     AH, @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION
12445                                  <1> ;--------------------------------------------------------------------------------
12446                                  <1> SETUP_END:
12447 00003DD0 B202                    <1>       MOV    DL,2                ; GET THE MOTOR WAIT PARAMETER
12448 00003DD2 6650                    <1>       PUSH   AX                  ; SAVE NUMBER TRANSFERRED
12449 00003DD4 E8CB000000              <1>       CALL   GET_PARM
12450 00003DD9 8825[4F4E0100]          <1>       MOV    [MOTOR_COUNT],AH    ; STORE UPON RETURN
12451 00003DDF 6658                    <1>       POP    AX                  ; RESTORE NUMBER TRANSFERRED
12452 00003DE1 8A25[504E0100]          <1>       MOV    AH, [DSKETTE_STATUS]      ; GET STATUS OF OPERATION
12453 00003DE7 08E4                    <1>       OR     AH,AH               ; CHECK FOR ERROR
12454 00003DE9 7402                    <1>       JZ     short NUN_ERR       ; NO ERROR
12455 00003DEB 30C0                    <1>       XOR    AL,AL               ; CLEAR NUMBER RETURNED
12456                                  <1> NUN_ERR:
12457 00003DED 80FC01                  <1>       CMP    AH,1                ; SET THE CARRY FLAG TO INDICATE
12458 00003DF0 F5                      <1>       CMC                        ; SUCCESS OR FAILURE
12459 00003DF1 C3                      <1>       RETn
12460                                  <1>
12461                                  <1> ;--------------------------------------------------------------------------------
12462                                  <1> ; SETUP_DBL
12463                                  <1> ;     CHECK DOUBLE STEP.
12464                                  <1> ;
12465                                  <1> ; ON ENTRY : DI = DRIVE
12466                                  <1> ;
12467                                  <1> ; ON EXIT :  CY = 1 MEANS ERROR
12468                                  <1> ;--------------------------------------------------------------------------------
12469                                  <1> SETUP_DBL:
12470 00003DF2 8AA7[5D4E0100]          <1>       MOV    AH, [DSK_STATE+eDI] ; ACCESS STATE
12471 00003DF8 F6C410                  <1>       TEST   AH,MED_DET          ; ESTABLISHED STATE ?
12472 00003DFB 757E                    <1>       JNZ    short NO_DBL              ; IF ESTABLISHED THEN DOUBLE DONE
12473                                  <1>
12474                                  <1> ;-----      CHECK FOR TRACK 0 TO SPEED UP ACKNOWLEDGE OF UNFORMATTED DISKETTE
12475                                  <1>
12476 00003DFD C605[4D4E0100]00        <1>       MOV    byte [SEEK_STATUS],0      ; SET RECALIBRATE REQUIRED ON ALL DRIVES
12477 00003E04 E8E0000000              <1>       CALL   MOTOR_ON            ; ENSURE MOTOR STAY ON
12478 00003E09 B500                    <1>       MOV    CH,0                ; LOAD TRACK 0
12479 00003E0B E8D4010000              <1>       CALL   SEEK                ; SEEK TO TRACK 0
12480 00003E10 E868000000              <1>       CALL   READ_ID                  ; READ ID FUNCTION
12481 00003E15 7249                    <1>       JC     short SD_ERR        ; IF ERROR NO TRACK 0
12482                                  <1>
12483                                  <1> ;-----      INITIALIZE START AND MAX TRACKS (TIMES 2 FOR BOTH HEADS)
12484                                  <1>
12485 00003E17 66B95004                <1>       MOV    CX,0450H            ; START, MAX TRACKS
12486 00003E1B F687[5D4E0100]01        <1>       TEST   byte [DSK_STATE+eDI],TRK_CAPA ; TEST FOR 80 TRACK CAPABILITY
12487 00003E22 7402                    <1>       JZ     short CNT_OK        ; IF NOT COUNT IS SETUP
12488 00003E24 B1A0                    <1>       MOV    CL,0A0H                  ; MAXIMUM TRACK 1.2 MB
12489                                  <1>
12490                                  <1> ;     ATTEMPT READ ID OF ALL TRACKS, ALL HEADS UNTIL SUCCESS; UPON SUCCESS,
12491                                  <1> ;     MUST SEE IF ASKED FOR TRACK IN SINGLE STEP MODE = TRACK ID READ; IF NOT
12492                                  <1> ;     THEN SET DOUBLE STEP ON.
12493                                  <1>
12494                                  <1> CNT_OK:
12495 00003E26 C605[4F4E0100]FF        <1>         MOV     byte [MOTOR_COUNT], 0FFH ; ENSURE MOTOR STAYS ON FOR OPERATION
12496 00003E2D 6651                    <1>       PUSH   CX                  ; SAVE TRACK, COUNT
12497 00003E2F C605[504E0100]00        <1>       MOV    byte [DSKETTE_STATUS],0   ; CLEAR STATUS, EXPECT ERRORS
12498 00003E36 6631C0                  <1>       XOR    AX,AX               ; CLEAR AX
12499 00003E39 D0ED                    <1>       SHR    CH,1                ; HALVE TRACK, CY = HEAD
12500 00003E3B C0D003                  <1>       RCL    AL,3                ; AX = HEAD IN CORRECT BIT
12501 00003E3E 6650                    <1>       PUSH   AX                  ; SAVE HEAD
12502 00003E40 E89F010000              <1>       CALL   SEEK                ; SEEK TO TRACK
12503 00003E45 6658                    <1>       POP    AX                  ; RESTORE HEAD
12504 00003E47 6609C7                  <1>       OR     DI,AX               ; DI = HEAD OR'ED DRIVE
12505 00003E4A E82E000000              <1>       CALL   READ_ID                  ; READ ID HEAD 0
12506 00003E4F 9C                      <1>       PUSHF                      ; SAVE RETURN FROM READ_ID
12507 00003E50 6681E7FB00              <1>       AND    DI,11111011B        ; TURN OFF HEAD 1 BIT
12508 00003E55 9D                      <1>       POPF                       ; RESTORE ERROR RETURN
12509 00003E56 6659                    <1>       POP    CX                  ; RESTORE COUNT
12510 00003E58 7308                    <1>       JNC    short DO_CHK        ; IF OK, ASKED = RETURNED TRACK ?
12511 00003E5A FEC5                    <1>       INC    CH                  ; INC FOR NEXT TRACK
12512 00003E5C 38CD                    <1>       CMP    CH,CL               ; REACHED MAXIMUM YET
12513 00003E5E 75C6                    <1>       JNZ    short CNT_OK        ; CONTINUE TILL ALL TRIED
12514                                  <1>
12515                                  <1> ;-----      FALL THRU, READ ID FAILED FOR ALL TRACKS
12516                                  <1>
12517                                  <1> SD_ERR:
12518 00003E60 F9                      <1>       STC                        ; SET CARRY FOR ERROR
12519 00003E61 C3                      <1>       RETn                       ; SETUP_DBL ERROR EXIT
12520                                  <1>
12521                                  <1> DO_CHK:
12522 00003E62 8A0D[544E0100]          <1>       MOV    CL, [NEC_STATUS+3]  ; LOAD RETURNED TRACK
12523 00003E68 888F[614E0100]          <1>       MOV    [DSK_TRK+eDI], CL   ; STORE TRACK NUMBER
12524 00003E6E D0ED                    <1>       SHR    CH,1                ; HALVE TRACK
12525 00003E70 38CD                    <1>       CMP    CH,CL               ; IS IT THE SAME AS ASKED FOR TRACK
12526 00003E72 7407                    <1>       JZ     short NO_DBL        ; IF SAME THEN NO DOUBLE STEP
12527 00003E74 808F[5D4E0100]20        <1>       OR     byte [DSK_STATE+eDI],DBL_STEP ; TURN ON DOUBLE STEP REQUIRED
12528                                  <1> NO_DBL:
12529 00003E7B F8                      <1>       CLC                        ; CLEAR ERROR FLAG
12530 00003E7C C3                      <1>       RETn
12531                                  <1>
12532                                  <1> ;--------------------------------------------------------------------------------
12533                                  <1> ; READ_ID
12534                                  <1> ;     READ ID FUNCTION.
12535                                  <1> ;
12536                                  <1> ; ON ENTRY:  DI : BIT 2 = HEAD; BITS 1,0 = DRIVE
12537                                  <1> ;
12538                                  <1> ; ON EXIT:   DI : BIT 2 IS RESET, BITS 1,0 = DRIVE
12539                                  <1> ;            @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION
12540                                  <1> ;--------------------------------------------------------------------------------
12541                                  <1> READ_ID:
12542 00003E7D B8[9A3E0000]            <1>       MOV    eAX, ER_3           ; MOVE NEC OUTPUT ERROR ADDRESS
12543 00003E82 50                      <1>       PUSH   eAX
12544 00003E83 B44A                    <1>       MOV    AH,4AH              ; READ ID COMMAND
```

```
12545 00003E85 E820010000          <1>       CALL   NEC_OUTPUT          ; TO CONTROLLER
12546 00003E8A 6689F8              <1>       MOV    AX,DI               ; DRIVE # TO AH, HEAD 0
12547 00003E8D 88C4                <1>       MOV    AH,AL
12548 00003E8F E816010000          <1>       CALL   NEC_OUTPUT          ; TO CONTROLLER
12549 00003E94 E80BFEFFFF          <1>       CALL   NEC_TERM            ; WAIT FOR OPERATION, GET STATUS
12550 00003E99 58                  <1>       POP    eAX                 ; THROW AWAY ERROR ADDRESS
12551                              <1> ER_3:
12552 00003E9A C3                  <1>       RETn
12553                              <1>
12554                              <1> ;--------------------------------------------------------------------------
12555                              <1> ; CMOS_TYPE
12556                              <1> ;     RETURNS DISKETTE TYPE FROM CMOS
12557                              <1> ;
12558                              <1> ; ON ENTRY:  DI = DRIVE #
12559                              <1> ;
12560                              <1> ; ON EXIT:   AL = TYPE; CY REFLECTS STATUS
12561                              <1> ;--------------------------------------------------------------------------
12562                              <1>
12563                              <1> CMOS_TYPE: ; 11/12/2014
12564 00003E9B 8A87[F65C0000]      <1> mov   al, [eDI+fd0_type]
12565 00003EA1 20C0                <1> and   al, al ; 18/12/2014
12566 00003EA3 C3                  <1> retn
12567                              <1>
12568                              <1> ;CMOS_TYPE:
12569                              <1> ;     MOV    AL, CMOS_DIAG       ; CMOS DIAGNOSTIC STATUS BYTE ADDRESS
12570                              <1> ;     CALL   CMOS_READ           ; GET CMOS STATUS
12571                              <1> ;     TEST   AL,BAD_BAT+BAD_CKSUM      ; BATTERY GOOD AND CHECKSUM VALID
12572                              <1> ;     STC                        ; SET CY = 1 INDICATING ERROR FOR RETURN
12573                              <1> ;     JNZ    short BAD_CM        ; ERROR IF EITHER BIT ON
12574                              <1> ;     MOV    AL,CMOS_DISKETTE    ; ADDRESS OF DISKETTE BYTE IN CMOS
12575                              <1> ;     CALL   CMOS_READ           ; GET DISKETTE BYTE
12576                              <1> ;     OR     DI,DI               ; SEE WHICH DRIVE IN QUESTION
12577                              <1> ;     JNZ    short TB            ; IF DRIVE 1, DATA IN LOW NIBBLE
12578                              <1> ;     ROR    AL,4                ; EXCHANGE NIBBLES IF SECOND DRIVE
12579                              <1> ;TB:
12580                              <1> ;     AND    AL,0FH              ; KEEP ONLY DRIVE DATA, RESET CY, 0
12581                              <1> ;BAD_CM:
12582                              <1> ;     RETn                       ; CY, STATUS OF READ
12583                              <1>
12584                              <1> ;--------------------------------------------------------------------------
12585                              <1> ; GET_PARM
12586                              <1> ;     THIS ROUTINE FETCHES THE INDEXED POINTER FROM THE DISK_BASE
12587                              <1> ;     BLOCK POINTED TO BY THE DATA VARIABLE @DISK_POINTER. A BYTE FROM
12588                              <1> ;     THAT TABLE IS THEN MOVED INTO AH, THE INDEX OF THAT BYTE BEING
12589                              <1> ;     THE PARAMETER IN DL.
12590                              <1> ;
12591                              <1> ; ON ENTRY:  DL = INDEX OF BYTE TO BE FETCHED
12592                              <1> ;
12593                              <1> ; ON EXIT:   AH = THAT BYTE FROM BLOCK
12594                              <1> ;            AL,DH DESTROYED
12595                              <1> ;--------------------------------------------------------------------------
12596                              <1> GET_PARM:
12597                              <1>       ;PUSH  DS
12598 00003EA4 56                  <1>       PUSH   eSI
12599                              <1>       ;SUB   AX,AX               ; DS = 0, BIOS DATA AREA
12600                              <1>       ;MOV   DS,AX
12601                              <1>       ;;mov  ax, cs
12602                              <1>       ;;mov  ds, ax
12603                              <1>       ; 08/02/2015 (protected mode modifications, bx -> ebx)
12604 00003EA5 87D3                <1>       XCHG   eDX,eBX                   ; BL = INDEX
12605                              <1>       ;SUB   BH,BH               ; BX = INDEX
12606 00003EA7 81E3FF000000        <1>       and    ebx, 0FFh
12607                              <1>       ;LDS   SI, [DISK_POINTER]  ; POINT TO BLOCK
12608                              <1>       ;
12609                              <1>       ; 17/12/2014
12610 00003EAD 66A1[E55C0000]      <1>       mov    ax, [cfd] ; current (AL) and previous fd (AH)
12611 00003EB3 38E0                <1>       cmp    al, ah
12612 00003EB5 7425                <1>       je     short gpndc
12613 00003EB7 A2[E65C0000]        <1>       mov    [pfd], al ; current drive -> previous drive
12614 00003EBC 53                  <1>       push   ebx ; 08/02/2015
12615 00003EBD 88C3                <1>       mov    bl, al
12616                              <1>       ; 11/12/2014
12617 00003EBF 8A83[F65C0000]      <1>       mov    al, [eBX+fd0_type]  ; Drive type (0,1,2,3,4)
12618                              <1>       ; 18/12/2014
12619 00003EC5 20C0                <1>       and    al, al
12620 00003EC7 7507                <1>       jnz    short gpdtc
12621 00003EC9 BB[CF5C0000]        <1>       mov    ebx, MD_TBL6        ; 1.44 MB param. tbl. (default)
12622 00003ECE EB05                <1>         jmp     short gpdpu
12623                              <1> gpdtc:
12624 00003ED0 E817F9FFFF          <1>       call   DR_TYPE_CHECK
12625                              <1>       ; cf = 1 -> eBX points to 1.44MB fd parameter table (default)
12626                              <1> gpdpu:
12627 00003ED5 891D[6C5C0000]      <1>       mov    [DISK_POINTER], ebx
12628 00003EDB 5B                  <1>       pop    ebx
12629                              <1> gpndc:
12630 00003EDC 8B35[6C5C0000]      <1>       mov    esi, [DISK_POINTER] ; 08/02/2015, si -> esi
12631 00003EE2 8A241E              <1>       MOV    AH, [eSI+eBX]       ; GET THE WORD
12632 00003EE5 87D3                <1>       XCHG   eDX,eBX                   ; RESTORE BX
12633 00003EE7 5E                  <1>       POP    eSI
12634                              <1>       ;POP   DS
12635 00003EE8 C3                  <1>       RETn
12636                              <1>
12637                              <1> ;--------------------------------------------------------------------------
12638                              <1> ; MOTOR_ON
12639                              <1> ;     TURN MOTOR ON AND WAIT FOR MOTOR START UP TIME. THE @MOTOR_COUNT
12640                              <1> ;     IS REPLACED WITH A SUFFICIENTLY HIGH NUMBER (0FFH) TO ENSURE
12641                              <1> ;     THAT THE MOTOR DOES NOT GO OFF DURING THE OPERATION. IF THE
12642                              <1> ;     MOTOR NEEDED TO BE TURNED ON, THE MULTI-TASKING HOOK FUNCTION
12643                              <1> ;     (AX=90FDH, INT 15) IS CALLED TELLING THE OPERATING SYSTEM
12644                              <1> ;     THAT THE BIOS IS ABOUT TO WAIT FOR MOTOR START UP. IF THIS
12645                              <1> ;     FUNCTION RETURNS WITH CY = 1, IT MEANS THAT THE MINIMUM WAIT
12646                              <1> ;     HAS BEEN COMPLETED. AT THIS POINT A CHECK IS MADE TO ENSURE
```

```
12647                                  <1> ;     THAT THE MOTOR WASN'T TURNED OFF BY THE TIMER. IF THE HOOK DID
12648                                  <1> ;     NOT WAIT, THE WAIT FUNCTION (AH=086H) IS CALLED TO WAIT THE
12649                                  <1> ;     PRESCRIBED AMOUNT OF TIME. IF THE CARRY FLAG IS SET ON RETURN,
12650                                  <1> ;     IT MEANS THAT THE FUNCTION IS IN USE AND DID NOT PERFORM THE
12651                                  <1> ;     WAIT. A TIMER 1 WAIT LOOP WILL THEN DO THE WAIT.
12652                                  <1> ;
12653                                  <1> ; ON ENTRY:  DI = DRIVE #
12654                                  <1> ; ON EXIT:   AX,CX,DX DESTROYED
12655                                  <1> ;--------------------------------------------------------------------------
12656                                  <1> MOTOR_ON:
12657 00003EE9 53                      <1>       PUSH   eBX                  ; SAVE REG.
12658 00003EEA E82A000000              <1>       CALL   TURN_ON                    ; TURN ON MOTOR
12659 00003EEF 7226                    <1>       JC     short MOT_IS_ON            ; IF CY=1 NO WAIT
12660 00003EF1 E89BF9FFFF              <1>       CALL   XLAT_OLD            ; TRANSLATE STATE TO COMPATIBLE MODE
12661 00003EF6 E865F9FFFF              <1>       CALL   XLAT_NEW            ; TRANSLATE STATE TO PRESENT ARCH,
12662                                  <1>       ;CALL  TURN_ON             ; CHECK AGAIN IF MOTOR ON
12663                                  <1>       ;JC    MOT_IS_ON           ; IF NO WAIT MEANS IT IS ON
12664                                  <1> M_WAIT:
12665 00003EFB B20A                    <1>       MOV    DL,10               ; GET THE MOTOR WAIT PARAMETER
12666 00003EFD E8A2FFFFFF              <1>       CALL   GET_PARM
12667                                  <1>       ;MOV   AL,AH               ; AL = MOTOR WAIT PARAMETER
12668                                  <1>       ;XOR   AH,AH               ; AX = MOTOR WAIT PARAMETER
12669                                  <1>       ;CMP   AL,8                ; SEE IF AT LEAST A SECOND IS SPECIFIED
12670 00003F02 80FC08                  <1>       cmp    ah, 8
12671                                  <1>       ;JAE   short GP2           ; IF YES, CONTINUE
12672 00003F05 7702                    <1>       ja     short J13
12673                                  <1>       ;MOV   AL,8                ; ONE SECOND WAIT FOR MOTOR START UP
12674 00003F07 B408                    <1>       mov    ah, 8
12675                                  <1>
12676                                  <1> ;-----      AS CONTAINS NUMBER OF 1/8 SECONDS (125000 MICROSECONDS) TO WAIT
12677                                  <1> GP2:
12678                                  <1> ;-----      FOLLOWING LOOPS REQUIRED WHEN RTC WAIT FUNCTION IS ALREADY IN USE
12679                                  <1> J13:                           ; WAIT FOR 1/8 SECOND PER (AL)
12680 00003F09 B95E200000              <1>       MOV    eCX,8286            ; COUNT FOR 1/8 SECOND AT 15.085737 US
12681 00003F0E E8DADEFFFF              <1>       CALL   WAITF               ; GO TO FIXED WAIT ROUTINE
12682                                  <1>       ;DEC   AL                  ; DECREMENT TIME VALUE
12683 00003F13 FECC                    <1>       dec    ah
12684 00003F15 75F2                    <1>       JNZ    short J13           ; ARE WE DONE YET
12685                                  <1> MOT_IS_ON:
12686 00003F17 5B                      <1>       POP    eBX                 ; RESTORE REG.
12687 00003F18 C3                      <1>       RETn
12688                                  <1>
12689                                  <1> ;--------------------------------------------------------------------------
12690                                  <1> ; TURN_ON
12691                                  <1> ;     TURN MOTOR ON AND RETURN WAIT STATE.
12692                                  <1> ;
12693                                  <1> ; ON ENTRY:  DI = DRIVE #
12694                                  <1> ;
12695                                  <1> ; ON EXIT:   CY = 0 MEANS WAIT REQUIRED
12696                                  <1> ;            CY = 1 MEANS NO WAIT REQUIRED
12697                                  <1> ;            AX,BX,CX,DX DESTROYED
12698                                  <1> ;--------------------------------------------------------------------------
12699                                  <1> TURN_ON:
12700 00003F19 89FB                    <1>       MOV    eBX,eDI                   ; BX = DRIVE #
12701 00003F1B 88D9                    <1>       MOV    CL,BL               ; CL = DRIVE #
12702 00003F1D C0C304                  <1>       ROL    BL,4                ; BL = DRIVE SELECT
12703 00003F20 FA                      <1>       CLI                        ; NO INTERRUPTS WHILE DETERMINING STATUS
12704 00003F21 C605[4F4E0100]FF        <1>       MOV    byte [MOTOR_COUNT],0FFH    ; ENSURE MOTOR STAYS ON FOR OPERATION
12705 00003F28 A0[4E4E0100]            <1>       MOV    AL, [MOTOR_STATUS]  ; GET DIGITAL OUTPUT REGISTER REFLECTION
12706 00003F2D 2430                    <1>       AND    AL,00110000B        ; KEEP ONLY DRIVE SELECT BITS
12707 00003F2F B401                    <1>       MOV    AH,1                ; MASK FOR DETERMINING MOTOR BIT
12708 00003F31 D2E4                    <1>       SHL    AH,CL               ; AH = MOTOR ON, A=00000001, B=00000010
12709                                  <1>
12710                                  <1> ;  AL = DRIVE SELECT FROM @MOTOR_STATUS
12711                                  <1> ;  BL = DRIVE SELECT DESIRED
12712                                  <1> ;  AH = MOTOR ON MASK DESIRED
12713                                  <1>
12714 00003F33 38D8                    <1>       CMP    AL,BL               ; REQUESTED DRIVE ALREADY SELECTED ?
12715 00003F35 7508                    <1>       JNZ    short TURN_IT_ON    ; IF NOT SELECTED JUMP
12716 00003F37 8425[4E4E0100]          <1>       TEST   AH, [MOTOR_STATUS]  ; TEST MOTOR ON BIT
12717 00003F3D 7535                    <1>       JNZ    short NO_MOT_WAIT   ; JUMP IF MOTOR ON AND SELECTED
12718                                  <1>
12719                                  <1> TURN_IT_ON:
12720 00003F3F 08DC                    <1>       OR     AH,BL               ; AH = DRIVE SELECT AND MOTOR ON
12721 00003F41 8A3D[4E4E0100]          <1>       MOV    BH,[MOTOR_STATUS]   ; SAVE COPY OF @MOTOR_STATUS BEFORE
12722 00003F47 80E70F                  <1>       AND    BH,00001111B        ; KEEP ONLY MOTOR BITS
12723 00003F4A 8025[4E4E0100]CF        <1>       AND    byte [MOTOR_STATUS],11001111B ; CLEAR OUT DRIVE SELECT
12724 00003F51 0825[4E4E0100]          <1>       OR     [MOTOR_STATUS],AH   ; OR IN DRIVE SELECTED AND MOTOR ON
12725 00003F57 A0[4E4E0100]            <1>       MOV    AL,[MOTOR_STATUS]   ; GET DIGITAL OUTPUT REGISTER REFLECTION
12726 00003F5C 88C3                    <1>       MOV    BL,AL               ; BL=@MOTOR_STATUS AFTER, BH=BEFORE
12727 00003F5E 80E30F                  <1>       AND    BL,00001111B        ; KEEP ONLY MOTOR BITS
12728 00003F61 FB                      <1>       STI                        ; ENABLE INTERRUPTS AGAIN
12729 00003F62 243F                    <1>       AND    AL,00111111B        ; STRIP AWAY UNWANTED BITS
12730 00003F64 C0C004                  <1>       ROL    AL,4                ; PUT BITS IN DESIRED POSITIONS
12731 00003F67 0C0C                    <1>       OR     AL,00001100B        ; NO RESET, ENABLE DMA/INTERRUPT
12732 00003F69 66BAF203                <1>       MOV    DX,03F2H            ; SELECT DRIVE AND TURN ON MOTOR
12733 00003F6D EE                      <1>       OUT    DX,AL
12734 00003F6E 38FB                    <1>       CMP    BL,BH               ; NEW MOTOR TURNED ON ?
12735                                  <1>       ;JZ    short NO_MOT_WAIT   ; NO WAIT REQUIRED IF JUST SELECT
12736 00003F70 7403                    <1>       je     short no_mot_w1 ; 27/02/2015
12737 00003F72 F8                      <1>       CLC                        ; (re)SET CARRY MEANING WAIT
12738 00003F73 C3                      <1>       RETn
12739                                  <1>
12740                                  <1> NO_MOT_WAIT:
12741 00003F74 FB                      <1>       sti
12742                                  <1> no_mot_w1: ; 27/02/2015
12743 00003F75 F9                      <1>       STC                        ; SET NO WAIT REQUIRED
12744                                  <1>       ;STI                       ; INTERRUPTS BACK ON
12745 00003F76 C3                      <1>       RETn
12746                                  <1>
12747                                  <1> ;--------------------------------------------------------------------------
12748                                  <1> ; HD_WAIT
```

```
12749                                  <1> ;     WAIT FOR HEAD SETTLE TIME.
12750                                  <1> ;
12751                                  <1> ; ON ENTRY:  DI = DRIVE #
12752                                  <1> ;
12753                                  <1> ; ON EXIT:   AX,BX,CX,DX DESTROYED
12754                                  <1> ;--------------------------------------------------------------------------
12755                                  <1> HD_WAIT:
12756 00003F77 B209                   <1>       MOV    DL,9                ; GET HEAD SETTLE PARAMETER
12757 00003F79 E826FFFFFF             <1>       CALL   GET_PARM
12758 00003F7E 08E4                   <1>       or     ah, ah ; 17/12/2014
12759 00003F80 7519                   <1>       jnz    short DO_WAT
12760 00003F82 F605[4E4E0100]80       <1>         TEST    byte [MOTOR_STATUS],10000000B ; SEE IF A WRITE OPERATION
12761                                  <1>       ;JZ    short ISNT_WRITE    ; IF NOT, DO NOT ENFORCE ANY VALUES
12762                                  <1>       ;OR    AH,AH               ; CHECK FOR ANY WAIT?
12763                                  <1>       ;JNZ   short DO_WAT        ; IF THERE DO NOT ENFORCE
12764 00003F89 741E                   <1>       jz     short HW_DONE
12765 00003F8B B40F                   <1>       MOV    AH,HD12_SETTLE            ; LOAD 1.2M HEAD SETTLE MINIMUM
12766 00003F8D 8A87[5D4E0100]         <1>       MOV    AL,[DSK_STATE+eDI]  ; LOAD STATE
12767 00003F93 24C0                   <1>       AND    AL,RATE_MSK         ; KEEP ONLY RATE
12768 00003F95 3C80                   <1>       CMP    AL,RATE_250         ; 1.2 M DRIVE ?
12769 00003F97 7502                   <1>       JNZ    short DO_WAT        ; DEFAULT HEAD SETTLE LOADED
12770                                  <1> ;GP3:
12771 00003F99 B414                   <1>       MOV    AH,HD320_SETTLE           ; USE 320/360 HEAD SETTLE
12772                                  <1> ;     JMP    SHORT DO_WAT
12773                                  <1>
12774                                  <1> ;ISNT_WRITE:
12775                                  <1> ;     OR     AH,AH               ; CHECK FOR NO WAIT
12776                                  <1> ;     JZ     short HW_DONE       ; IF NOT WRITE AND 0 ITS OK
12777                                  <1>
12778                                  <1> ;-----       AH CONTAINS NUMBER OF MILLISECONDS TO WAIT
12779                                  <1> DO_WAT:
12780                                  <1> ;     MOV    AL,AH               ; AL = # MILLISECONDS
12781                                  <1> ;     ;XOR   AH,AH               ; AX = # MILLISECONDS
12782                                  <1> J29:                             ;      1 MILLISECOND LOOP
12783                                  <1>       ;mov   cx, WAIT_FDU_HEAD_SETTLE ; 33 ; 1 ms in 30 micro units.
12784 00003F9B B942000000             <1>       MOV    eCX,66              ; COUNT AT 15.085737 US PER COUNT
12785 00003FA0 E848DEFFFF             <1>       CALL   WAITF               ; DELAY FOR 1 MILLISECOND
12786                                  <1>       ;DEC   AL                  ; DECREMENT THE COUNT
12787 00003FA5 FECC                   <1>       dec    ah
12788 00003FA7 75F2                   <1>       JNZ    short J29           ; DO AL MILLISECOND # OF TIMES
12789                                  <1> HW_DONE:
12790 00003FA9 C3                     <1>       RETn
12791                                  <1>
12792                                  <1> ;--------------------------------------------------------------------------
12793                                  <1> ; NEC_OUTPUT
12794                                  <1> ;     THIS ROUTINE SENDS A BYTE TO THE NEC CONTROLLER AFTER TESTING
12795                                  <1> ;     FOR CORRECT DIRECTION AND CONTROLLER READY THIS ROUTINE WILL
12796                                  <1> ;     TIME OUT IF THE BYTE IS NOT ACCEPTED WITHIN A REASONABLE AMOUNT
12797                                  <1> ;     OF TIME, SETTING THE DISKETTE STATUS ON COMPLETION.
12798                                  <1> ;
12799                                  <1> ; ON ENTRY:  AH = BYTE TO BE OUTPUT
12800                                  <1> ;
12801                                  <1> ; ON EXIT:   CY = 0  SUCCESS
12802                                  <1> ;            CY = 1  FAILURE -- DISKETTE STATUS UPDATED
12803                                  <1> ;                    IF A FAILURE HAS OCCURRED, THE RETURN IS MADE ONE LEVEL
12804                                  <1> ;                    HIGHER THAN THE CALLER OF NEC OUTPUT. THIS REMOVES THE
12805                                  <1> ;                    REQUIREMENT OF TESTING AFTER EVERY CALL OF NEC_OUTPUT.
12806                                  <1> ;            AX,CX,DX DESTROYED
12807                                  <1> ;--------------------------------------------------------------------------
12808                                  <1>
12809                                  <1> ; 09/12/2014 [Erdogan Tan]
12810                                  <1> ;     (from 'PS2 Hardware Interface Tech. Ref. May 88', Page 09-05.)
12811                                  <1> ; Diskette Drive Controller Status Register (3F4h)
12812                                  <1> ;     This read only register facilitates the transfer of data between
12813                                  <1> ;     the system microprocessor and the controller.
12814                                  <1> ; Bit 7 - When set to 1, the Data register is ready to transfer data
12815                                  <1> ;       with the system micrprocessor.
12816                                  <1> ; Bit 6 - The direction of data transfer. If this bit is set to 0,
12817                                  <1> ;       the transfer is to the controller.
12818                                  <1> ; Bit 5 - When this bit is set to 1, the controller is in the non-DMA mode.
12819                                  <1> ; Bit 4 - When this bit is set to 1, a Read or Write command is being executed.
12820                                  <1> ; Bit 3 - Reserved.
12821                                  <1> ; Bit 2 - Reserved.
12822                                  <1> ; Bit 1 - When this bit is set to 1, dskette drive 1 is in the seek mode.
12823                                  <1> ; Bit 0 - When this bit is set to 1, dskette drive 1 is in the seek mode.
12824                                  <1>
12825                                  <1> ; Data Register (3F5h)
12826                                  <1> ; This read/write register passes data, commands and parameters, and provides
12827                                  <1> ; diskette status information.
12828                                  <1>
12829                                  <1> NEC_OUTPUT:
12830                                  <1>       ;PUSH  BX                  ; SAVE REG.
12831 00003FAA 66BAF403               <1>       MOV    DX,03F4H            ; STATUS PORT
12832                                  <1>       ;MOV   BL,2                ; HIGH ORDER COUNTER
12833                                  <1>       ;XOR   CX,CX               ; COUNT FOR TIME OUT
12834                                  <1>       ; 16/12/2014
12835                                  <1>       ; waiting for (max.) 0.5 seconds
12836                                  <1>         ;;mov     byte [wait_count], 0 ;; 27/02/2015
12837                                  <1>       ;
12838                                  <1>       ; 17/12/2014
12839                                  <1>       ; Modified from AWARD BIOS 1999 - ADISK.ASM - SEND_COMMAND
12840                                  <1>       ;
12841                                  <1>       ;WAIT_FOR_PORT:     Waits for a bit at a port pointed to by DX to
12842                                  <1>       ;            go on.
12843                                  <1>       ;INPUT:
12844                                  <1>       ;      AH=Mask for isolation bits.
12845                                  <1>       ;      AL=pattern to look for.
12846                                  <1>       ;      DX=Port to test for
12847                                  <1>       ;      BH:CX=Number of memory refresh periods to delay.
12848                                  <1>       ;           (normally 30 microseconds per period.)
12849                                  <1>       ;
12850                                  <1>       ;WFP_SHORT:
```

```
12851                           <1>       ;     Wait for port if refresh cycle is short (15-80 Us range).
12852                           <1>       ;
12853                           <1>
12854                           <1> ;     mov   bl, WAIT_FDU_SEND_HI+1    ; 0+1
12855                           <1> ;     mov   cx, WAIT_FDU_SEND_LO      ; 16667
12856 00003FAE B91B410000        <1>       mov   ecx, WAIT_FDU_SEND_LH   ; 16667 (27/02/2015)
12857                           <1> ;
12858                           <1> ;WFPS_OUTER_LP:
12859                           <1> ;     ;
12860                           <1> ;WFPS_CHECK_PORT:
12861                           <1> J23:
12862 00003FB3 EC                <1>       IN    AL,DX               ; GET STATUS
12863 00003FB4 24C0              <1>       AND   AL,11000000B        ; KEEP STATUS AND DIRECTION
12864 00003FB6 3C80              <1>       CMP   AL,10000000B        ; STATUS 1 AND DIRECTION 0 ?
12865 00003FB8 7418              <1>       JZ    short J27           ; STATUS AND DIRECTION OK
12866                           <1> WFPS_HI:
12867 00003FBA E461              <1>       IN    AL, PORT_B   ;061h  ; SYS1 ; wait for hi to lo
12868 00003FBC A810              <1>       TEST  AL,010H                   ; transition on memory
12869 00003FBE 75FA              <1>       JNZ   SHORT WFPS_HI       ; refresh.
12870                           <1> WFPS_LO:
12871 00003FC0 E461              <1>       IN    AL, PORT_B          ; SYS1
12872 00003FC2 A810              <1>       TEST  AL,010H
12873 00003FC4 74FA              <1>       JZ    SHORT WFPS_LO
12874                           <1>       ;LOOP SHORT WFPS_CHECK_PORT
12875 00003FC6 E2EB              <1>       loop  J23    ; 27/02/2015
12876                           <1> ;     ;
12877                           <1> ;     dec   bl
12878                           <1> ;     jnz   short WFPS_OUTER_LP
12879                           <1> ;     jmp   short WFPS_TIMEOUT  ; fail
12880                           <1> ;J23:
12881                           <1> ;     IN    AL,DX               ; GET STATUS
12882                           <1> ;     AND   AL,11000000B        ; KEEP STATUS AND DIRECTION
12883                           <1> ;     CMP   AL,10000000B        ; STATUS 1 AND DIRECTION 0 ?
12884                           <1> ;     JZ    short J27           ; STATUS AND DIRECTION OK
12885                           <1>       ;LOOP J23                 ; CONTINUE TILL CX EXHAUSTED
12886                           <1>       ;DEC  BL                  ; DECREMENT COUNTER
12887                           <1>       ;JNZ  short J23           ; REPEAT TILL DELAY FINISHED, CX = 0
12888                           <1>
12889                           <1>       ;;27/02/2015
12890                           <1>       ;16/12/2014
12891                           <1>         ;;cmp     byte [wait_count], 10   ; (10/18.2 seconds)
12892                           <1>       ;;jb  short J23
12893                           <1>
12894                           <1> ;WFPS_TIMEOUT:
12895                           <1>
12896                           <1> ;-----      FALL THRU TO ERROR RETURN
12897                           <1>
12898 00003FC8 800D[504E0100]80  <1>       OR    byte [DSKETTE_STATUS],TIME_OUT
12899                           <1>       ;POP  BX                  ; RESTORE REG.
12900 00003FCF 58                <1>       POP   eAX ; 08/02/2015    ; DISCARD THE RETURN ADDRESS
12901 00003FD0 F9                <1>       STC                       ; INDICATE ERROR TO CALLER
12902 00003FD1 C3                <1>       RETn
12903                           <1>
12904                           <1> ;-----      DIRECTION AND STATUS OK; OUTPUT BYTE
12905                           <1>
12906                           <1> J27:
12907 00003FD2 88E0              <1>       MOV   AL,AH               ; GET BYTE TO OUTPUT
12908 00003FD4 6642              <1>       INC   DX                  ; DATA PORT = STATUS PORT + 1
12909 00003FD6 EE                <1>       OUT   DX,AL               ; OUTPUT THE BYTE
12910                           <1>       ;;NEWIODELAY  ;; 27/02/2015
12911                           <1>       ; 27/02/2015
12912 00003FD7 9C                <1>       PUSHF                     ; SAVE FLAGS
12913 00003FD8 B903000000        <1>       MOV   eCX, 3              ; 30 TO 45 MICROSECONDS WAIT FOR
12914 00003FDD E80BDEFFFF        <1>       CALL  WAITF               ; NEC FLAGS UPDATE CYCLE
12915 00003FE2 9D                <1>       POPF                      ; RESTORE FLAGS FOR EXIT
12916                           <1>       ;POP  BX                  ; RESTORE REG
12917 00003FE3 C3                <1>       RETn                      ; CY = 0 FROM TEST INSTRUCTION
12918                           <1>
12919                           <1> ;--------------------------------------------------------------------------------
12920                           <1> ; SEEK
12921                           <1> ;     THIS ROUTINE WILL MOVE THE HEAD ON THE NAMED DRIVE TO THE NAMED
12922                           <1> ;     TRACK. IF THE DRIVE HAS NOT BEEN ACCESSED SINCE THE DRIVE
12923                           <1> ;     RESET COMMAND WAS ISSUED, THE DRIVE WILL BE RECALIBRATED.
12924                           <1> ;
12925                           <1> ; ON ENTRY:  DI = DRIVE #
12926                           <1> ;            CH = TRACK #
12927                           <1> ;
12928                           <1> ; ON EXIT:   @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION.
12929                           <1> ;            AX,BX,CX DX DESTROYED
12930                           <1> ;--------------------------------------------------------------------------------
12931                           <1> SEEK:
12932 00003FE4 89FB              <1>       MOV   eBX,eDI                   ; BX = DRIVE #
12933 00003FE6 B001              <1>       MOV   AL,1                ; ESTABLISH MASK FOR RECALIBRATE TEST
12934 00003FE8 86CB              <1>       XCHG  CL,BL               ; SET DRIVE VALULE INTO CL
12935 00003FEA D2C0              <1>       ROL   AL,CL               ; SHIFT MASK BY THE DRIVE VALUE
12936 00003FEC 86CB              <1>       XCHG  CL,BL               ; RECOVER TRACK VALUE
12937 00003FEE 8405[4D4E0100]    <1>       TEST  AL,[SEEK_STATUS]    ; TEST FOR RECALIBRATE REQUIRED
12938 00003FF4 7526              <1>       JNZ   short J28A          ; JUMP IF RECALIBRATE NOT REQUIRED
12939                           <1>
12940 00003FF6 0805[4D4E0100]    <1>       OR    [SEEK_STATUS],AL    ; TURN ON THE NO RECALIBRATE BIT IN FLAG
12941 00003FFC E862000000        <1>       CALL  RECAL               ; RECALIBRATE DRIVE
12942 00004001 730E              <1>       JNC   short AFT_RECAL           ; RECALIBRATE DONE
12943                           <1>
12944                           <1> ;-----      ISSUE RECALIBRATE FOR 80 TRACK DISKETTES
12945                           <1>
12946 00004003 C605[504E0100]00  <1>       MOV   byte [DSKETTE_STATUS],0   ; CLEAR OUT INVALID STATUS
12947 0000400A E854000000        <1>       CALL  RECAL               ; RECALIBRATE DRIVE
12948 0000400F 7251              <1>       JC    short RB            ; IF RECALIBRATE FAILS TWICE THEN ERROR
12949                           <1>
12950                           <1> AFT_RECAL:
12951 00004011 C687[614E0100]00  <1>         MOV     byte [DSK_TRK+eDI],0    ; SAVE NEW CYLINDER AS PRESENT POSITION
12952 00004018 08ED              <1>       OR    CH,CH               ; CHECK FOR SEEK TO TRACK 0
```

```
12953 0000401A 743F                <1>       JZ     short DO_WAIT       ; HEAD SETTLE, CY = 0 IF JUMP
12954                              <1>
12955                              <1> ;-----      DRIVE IS IN SYNCHRONIZATION WITH CONTROLLER, SEEK TO TRACK
12956                              <1>
12957 0000401C F687[5D4E0100]20    <1> J28A: TEST   byte [DSK_STATE+eDI],DBL_STEP ; CHECK FOR DOUBLE STEP REQUIRED
12958 00004023 7402                <1>       JZ     short _R7           ; SINGLE STEP REQUIRED BYPASS DOUBLE
12959 00004025 D0E5                <1>       SHL    CH,1                ; DOUBLE NUMBER OF STEP TO TAKE
12960                              <1>
12961 00004027 3AAF[614E0100]      <1> _R7:  CMP    CH, [DSK_TRK+eDI]   ; SEE IF ALREADY AT THE DESIRED TRACK
12962 0000402D 7433                <1>       JE     short RB            ; IF YES, DO NOT NEED TO SEEK
12963                              <1>
12964 0000402F BA[62400000]        <1>       MOV    eDX, NEC_ERR        ; LOAD RETURN ADDRESS
12965 00004034 52                  <1>       PUSH   eDX ; (*)           ; ON STACK FOR NEC OUTPUT ERROR
12966 00004035 88AF[614E0100]      <1>       MOV    [DSK_TRK+eDI],CH    ; SAVE NEW CYLINDER AS PRESENT POSITION
12967 0000403B B40F                <1>       MOV    AH,0FH              ; SEEK COMMAND TO NEC
12968 0000403D E868FFFFFF          <1>       CALL   NEC_OUTPUT
12969 00004042 89FB                <1>       MOV    eBX,eDI                   ; BX = DRIVE #
12970 00004044 88DC                <1>       MOV    AH,BL               ; OUTPUT DRIVE NUMBER
12971 00004046 E85FFFFFFF          <1>       CALL   NEC_OUTPUT
12972 0000404B 8AA7[614E0100]      <1>       MOV    AH, [DSK_TRK+eDI]   ; GET CYLINDER NUMBER
12973 00004051 E854FFFFFF          <1>       CALL   NEC_OUTPUT
12974 00004056 E829000000          <1>       CALL   CHK_STAT_2          ; ENDING INTERRUPT AND SENSE STATUS
12975                              <1>
12976                              <1> ;-----      WAIT FOR HEAD SETTLE
12977                              <1>
12978                              <1> DO_WAIT:
12979 0000405B 9C                  <1>       PUSHF                      ; SAVE STATUS
12980 0000405C E816FFFFFF          <1>       CALL   HD_WAIT                   ; WAIT FOR HEAD SETTLE TIME
12981 00004061 9D                  <1>       POPF                       ; RESTORE STATUS
12982                              <1> RB:
12983                              <1> NEC_ERR:
12984                              <1>       ; 08/02/2015 (code trick here from original IBM PC/AT DISKETTE.ASM)
12985                              <1>       ; (*) nec_err -> retn (push edx -> pop edx) -> nec_err -> retn
12986 00004062 C3                  <1>       RETn                       ; RETURN TO CALLER
12987                              <1>
12988                              <1> ;--------------------------------------------------------------------------------
12989                              <1> ; RECAL
12990                              <1> ;     RECALIBRATE DRIVE
12991                              <1> ;
12992                              <1> ; ON ENTRY:  DI = DRIVE #
12993                              <1> ;
12994                              <1> ; ON EXIT:   CY REFLECTS STATUS OF OPERATION.
12995                              <1> ;--------------------------------------------------------------------------------
12996                              <1> RECAL:
12997 00004063 6651                <1>       PUSH   CX
12998 00004065 B8[81400000]        <1>       MOV    eAX, RC_BACK        ; LOAD NEC_OUTPUT ERROR
12999 0000406A 50                  <1>       PUSH   eAX
13000 0000406B B407                <1>       MOV    AH,07H              ; RECALIBRATE COMMAND
13001 0000406D E838FFFFFF          <1>       CALL   NEC_OUTPUT
13002 00004072 89FB                <1>       MOV    eBX,eDI                   ; BX = DRIVE #
13003 00004074 88DC                <1>       MOV    AH,BL
13004 00004076 E82FFFFFFF          <1>       CALL   NEC_OUTPUT          ; OUTPUT THE DRIVE NUMBER
13005 0000407B E804000000          <1>       CALL   CHK_STAT_2          ; GET THE INTERRUPT AND SENSE INT STATUS
13006 00004080 58                  <1>       POP    eAX                 ; THROW AWAY ERROR
13007                              <1> RC_BACK:
13008 00004081 6659                <1>       POP    CX
13009 00004083 C3                  <1>       RETn
13010                              <1>
13011                              <1> ;--------------------------------------------------------------------------------
13012                              <1> ; CHK_STAT_2
13013                              <1> ;     THIS ROUTINE HANDLES THE INTERRUPT RECEIVED AFTER RECALIBRATE,
13014                              <1> ;     OR SEEK TO THE ADAPTER. THE INTERRUPT IS WAITED FOR, THE
13015                              <1> ;     INTERRUPT STATUS SENSED, AND THE RESULT RETURNED TO THE CALLER.
13016                              <1> ;
13017                              <1> ; ON EXIT:   @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION.
13018                              <1> ;--------------------------------------------------------------------------------
13019                              <1> CHK_STAT_2:
13020 00004084 B8[AC400000]        <1>         MOV     eAX, CS_BACK            ; LOAD NEC_OUTPUT ERROR ADDRESS
13021 00004089 50                  <1>       PUSH   eAX
13022 0000408A E828000000          <1>       CALL   WAIT_INT            ; WAIT FOR THE INTERRUPT
13023 0000408F 721A                <1>       JC     short J34           ; IF ERROR, RETURN IT
13024 00004091 B408                <1>       MOV    AH,08H              ; SENSE INTERRUPT STATUS COMMAND
13025 00004093 E812FFFFFF          <1>       CALL   NEC_OUTPUT
13026 00004098 E84A000000          <1>       CALL   RESULTS                   ; READ IN THE RESULTS
13027 0000409D 720C                <1>       JC     short J34
13028 0000409F A0[514E0100]        <1>       MOV    AL,[NEC_STATUS]           ; GET THE FIRST STATUS BYTE
13029 000040A4 2460                <1>       AND    AL,01100000B        ; ISOLATE THE BITS
13030 000040A6 3C60                <1>       CMP    AL,01100000B        ; TEST FOR CORRECT VALUE
13031 000040A8 7403                <1>       JZ     short J35           ; IF ERROR, GO MARK IT
13032 000040AA F8                  <1>       CLC                        ; GOOD RETURN
13033                              <1> J34:
13034 000040AB 58                  <1>       POP    eAX                 ; THROW AWAY ERROR RETURN
13035                              <1> CS_BACK:
13036 000040AC C3                  <1>       RETn
13037                              <1> J35:
13038 000040AD 800D[504E0100]40    <1>       OR     byte [DSKETTE_STATUS], BAD_SEEK
13039 000040B4 F9                  <1>       STC                        ; ERROR RETURN CODE
13040 000040B5 EBF4                <1>       JMP    SHORT J34
13041                              <1>
13042                              <1> ;--------------------------------------------------------------------------------
13043                              <1> ; WAIT_INT
13044                              <1> ;     THIS ROUTINE WAITS FOR AN INTERRUPT TO OCCUR A TIME OUT ROUTINE
13045                              <1> ;     TAKES PLACE DURING THE WAIT, SO THAT AN ERROR MAY BE RETURNED
13046                              <1> ;     IF THE DRIVE IS NOT READY.
13047                              <1> ;
13048                              <1> ; ON EXIT:   @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION.
13049                              <1> ;--------------------------------------------------------------------------------
13050                              <1>
13051                              <1> ; 17/12/2014
13052                              <1> ; 2.5 seconds waiting !
13053                              <1> ;(AWARD BIOS - 1999, WAIT_FDU_INT_LOW, WAIT_FDU_INT_HI)
13054                              <1> ; amount of time to wait for completion interrupt from NEC.
```

```
13055                                  <1>
13056                                  <1>
13057                                  <1> WAIT_INT:
13058 000040B7 FB                      <1>       STI                         ; TURN ON INTERRUPTS, JUST IN CASE
13059 000040B8 F8                      <1>       CLC                         ; CLEAR TIMEOUT INDICATOR
13060                                  <1>        ;MOV  BL,10                ; CLEAR THE COUNTERS
13061                                  <1>        ;XOR  CX,CX                ; FOR 2 SECOND WAIT
13062                                  <1>
13063                                  <1>       ; Modification from AWARD BIOS - 1999 (ATORGS.ASM, WAIT
13064                                  <1>       ;
13065                                  <1>       ;WAIT_FOR_MEM:
13066                                  <1>       ;     Waits for a bit at a specified memory location pointed
13067                                  <1>       ;     to by ES:[DI] to become set.
13068                                  <1>       ;INPUT:
13069                                  <1>       ;     AH=Mask to test with.
13070                                  <1>       ;     ES:[DI] = memory location to watch.
13071                                  <1>       ;     BH:CX=Number of memory refresh periods to delay.
13072                                  <1>       ;          (normally 30 microseconds per period.)
13073                                  <1>
13074                                  <1>       ; waiting for (max.) 2.5 secs in 30 micro units.
13075                                  <1> ;     mov    cx, WAIT_FDU_INT_LO       ; 017798
13076                                  <1> ;;    mov    bl, WAIT_FDU_INT_HI
13077                                  <1> ;     mov    bl, WAIT_FDU_INT_HI + 1
13078                                  <1>       ; 27/02/2015
13079 000040B9 B986450100              <1>       mov    ecx, WAIT_FDU_INT_LH      ; 83334 (2.5 seconds)
13080                                  <1> WFMS_CHECK_MEM:
13081 000040BE F605[4D4E0100]80        <1>       test   byte [SEEK_STATUS],INT_FLAG ; TEST FOR INTERRUPT OCCURRING
13082 000040C5 7516                    <1>         jnz     short J37
13083                                  <1> WFMS_HI:
13084 000040C7 E461                    <1>       IN     AL,PORT_B  ; 061h   ; SYS1, wait for lo to hi
13085 000040C9 A810                    <1>       TEST   AL,010H                   ; transition on memory
13086 000040CB 75FA                    <1>       JNZ    SHORT WFMS_HI       ; refresh.
13087                                  <1> WFMS_LO:
13088 000040CD E461                    <1>       IN     AL,PORT_B           ;SYS1
13089 000040CF A810                    <1>       TEST   AL,010H
13090 000040D1 74FA                    <1>       JZ     SHORT WFMS_LO
13091 000040D3 E2E9                    <1>         LOOP    WFMS_CHECK_MEM
13092                                  <1> ;WFMS_OUTER_LP:
13093                                  <1> ;;    or     bl, bl              ; check outer counter
13094                                  <1> ;;    jz     short J36A          ; WFMS_TIMEOUT
13095                                  <1> ;     dec    bl
13096                                  <1> ;     jz     short J36A
13097                                  <1> ;     jmp    short WFMS_CHECK_MEM
13098                                  <1>
13099                                  <1>       ;17/12/2014
13100                                  <1>       ;16/12/2014
13101                                  <1> ;        mov     byte [wait_count], 0    ; Reset (INT 08H) counter
13102                                  <1> ;J36:
13103                                  <1> ;     TEST   byte [SEEK_STATUS],INT_FLAG ; TEST FOR INTERRUPT OCCURRING
13104                                  <1> ;     JNZ    short J37
13105                                  <1>       ;16/12/2014
13106                                  <1>       ;LOOP  J36                 ; COUNT DOWN WHILE WAITING
13107                                  <1>       ;DEC   BL                  ; SECOND LEVEL COUNTER
13108                                  <1>       ;JNZ   short J36
13109                                  <1> ;       cmp     byte [wait_count], 46   ; (46/18.2 seconds)
13110                                  <1> ;     jb     short J36
13111                                  <1>
13112                                  <1> ;WFMS_TIMEOUT:
13113                                  <1> ;J36A:
13114 000040D5 800D[504E0100]80        <1>       OR     byte [DSKETTE_STATUS], TIME_OUT ; NOTHING HAPPENED
13115 000040DC F9                      <1>       STC                         ; ERROR RETURN
13116                                  <1> J37:
13117 000040DD 9C                      <1>       PUSHF                       ; SAVE CURRENT CARRY
13118 000040DE 8025[4D4E0100]7F        <1>       AND    byte [SEEK_STATUS], ~INT_FLAG ; TURN OFF INTERRUPT FLAG
13119 000040E5 9D                      <1>       POPF                        ; RECOVER CARRY
13120 000040E6 C3                      <1>       RETn                        ; GOOD RETURN CODE
13121                                  <1>
13122                                  <1> ;--------------------------------------------------------------------------
13123                                  <1> ; RESULTS
13124                                  <1> ;     THIS ROUTINE WILL READ ANYTHING THAT THE NEC CONTROLLER RETURNS
13125                                  <1> ;     FOLLOWING AN INTERRUPT.
13126                                  <1> ;
13127                                  <1> ; ON EXIT:   @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION.
13128                                  <1> ;            AX,BX,CX,DX DESTROYED
13129                                  <1> ;--------------------------------------------------------------------------
13130                                  <1> RESULTS:
13131 000040E7 57                      <1>       PUSH   eDI
13132 000040E8 BF[514E0100]            <1>       MOV    eDI, NEC_STATUS           ; POINTER TO DATA AREA
13133 000040ED B307                    <1>       MOV    BL,7                ; MAX STATUS BYTES
13134 000040EF 66BAF403                <1>       MOV    DX,03F4H            ; STATUS PORT
13135                                  <1>
13136                                  <1> ;-----      WAIT FOR REQUEST FOR MASTER
13137                                  <1>
13138                                  <1> _R10:
13139                                  <1>       ; 16/12/2014
13140                                  <1>       ; wait for (max) 0.5 seconds
13141                                  <1>       ;MOV   BH,2                ; HIGH ORDER COUNTER
13142                                  <1>       ;XOR   CX,CX               ; COUNTER
13143                                  <1>
13144                                  <1>       ;Time to wait while waiting for each byte of NEC results = .5
13145                                  <1>       ;seconds.  .5 seconds = 500,000 micros.  500,000/30 = 16,667.
13146                                  <1>       ; 27/02/2015
13147 000040F3 B91B410000              <1>       mov    ecx, WAIT_FDU_RESULTS_LH ; 16667
13148                                  <1>       ;mov   cx, WAIT_FDU_RESULTS_LO  ; 16667
13149                                  <1>       ;mov   bh, WAIT_FDU_RESULTS_HI+1 ; 0+1
13150                                  <1>
13151                                  <1> WFPSR_OUTER_LP:
13152                                  <1>       ;
13153                                  <1> WFPSR_CHECK_PORT:
13154                                  <1> J39:                              ; WAIT FOR MASTER
13155 000040F8 EC                      <1>       IN     AL,DX               ; GET STATUS
13156 000040F9 24C0                    <1>       AND    AL,11000000B        ; KEEP ONLY STATUS AND DIRECTION
```

```
13157 000040FB 3CC0                <1>       CMP    AL,11000000B        ; STATUS 1 AND DIRECTION 1 ?
13158 000040FD 7418                <1>       JZ     short J42           ; STATUS AND DIRECTION OK
13159                              <1> WFPSR_HI:
13160 000040FF E461                <1>       IN     AL, PORT_B   ;061h  ; SYS1 ; wait for hi to lo
13161 00004101 A810                <1>       TEST   AL,010H                  ; transition on memory
13162 00004103 75FA                <1>       JNZ    SHORT WFPSR_HI           ; refresh.
13163                              <1> WFPSR_LO:
13164 00004105 E461                <1>       IN     AL, PORT_B          ; SYS1
13165 00004107 A810                <1>       TEST   AL,010H
13166 00004109 74FA                <1>       JZ     SHORT WFPSR_LO
13167 0000410B E2EB                <1>         LOOP    WFPSR_CHECK_PORT
13168                              <1>       ;; 27/02/2015
13169                              <1>       ;;dec  bh
13170                              <1>       ;;jnz  short WFPSR_OUTER_LP
13171                              <1>       ;jmp   short WFPSR_TIMEOUT ; fail
13172                              <1>
13173                              <1>       ;;mov  byte [wait_count], 0
13174                              <1> ;J39:                             ; WAIT FOR MASTER
13175                              <1> ;     IN     AL,DX                ; GET STATUS
13176                              <1> ;     AND    AL,11000000B         ; KEEP ONLY STATUS AND DIRECTION
13177                              <1> ;     CMP    AL,11000000B         ; STATUS 1 AND DIRECTION 1 ?
13178                              <1> ;     JZ     short J42            ; STATUS AND DIRECTION OK
13179                              <1>       ;LOOP  J39                  ; LOOP TILL TIMEOUT
13180                              <1>       ;DEC   BH                   ; DECREMENT HIGH ORDER COUNTER
13181                              <1>       ;JNZ   short J39            ; REPEAT TILL DELAY DONE
13182                              <1>       ;
13183                              <1>       ;;cmp  byte [wait_count], 10  ; (10/18.2 seconds)
13184                              <1>       ;;jb   short J39
13185                              <1>
13186                              <1> ;WFPSR_TIMEOUT:
13187 0000410D 800D[504E0100]80    <1>       OR     byte [DSKETTE_STATUS],TIME_OUT
13188 00004114 F9                  <1>       STC                         ; SET ERROR RETURN
13189 00004115 EB29                <1>       JMP    SHORT POPRES         ; POP REGISTERS AND RETURN
13190                              <1>
13191                              <1> ;-----       READ IN THE STATUS
13192                              <1>
13193                              <1> J42:
13194 00004117 EB00                <1>       JMP    $+2                  ; I/O DELAY
13195 00004119 6642                <1>       INC    DX                   ; POINT AT DATA PORT
13196 0000411B EC                  <1>       IN     AL,DX                ; GET THE DATA
13197                              <1>       ; 16/12/2014
13198                              <1>       NEWIODELAY
13199 0000411C E6EB                <2>  out 0ebh,al
13200 0000411E 8807                <1>         MOV     [eDI],AL                ; STORE THE BYTE
13201 00004120 47                  <1>       INC    eDI                  ; INCREMENT THE POINTER
13202                              <1>       ; 16/12/2014
13203                              <1> ;     push   cx
13204                              <1> ;     mov    cx, 30
13205                              <1> ;wdw2:
13206                              <1> ;     NEWIODELAY
13207                              <1> ;     loop   wdw2
13208                              <1> ;     pop    cx
13209                              <1>
13210 00004121 B903000000          <1>       MOV    eCX,3                ; MINIMUM 24 MICROSECONDS FOR NEC
13211 00004126 E8C2DCFFFF          <1>       CALL   WAITF                ; WAIT 30 TO 45 MICROSECONDS
13212 0000412B 664A                <1>       DEC    DX                   ; POINT AT STATUS PORT
13213 0000412D EC                  <1>       IN     AL,DX                ; GET STATUS
13214                              <1>       ; 16/12/2014
13215                              <1>       NEWIODELAY
13216 0000412E E6EB                <2>  out 0ebh,al
13217                              <1>       ;
13218 00004130 A810                <1>       TEST   AL,00010000B         ; TEST FOR NEC STILL BUSY
13219 00004132 740C                <1>       JZ     short POPRES         ; RESULTS DONE ?
13220                              <1>
13221 00004134 FECB                <1>       DEC    BL                   ; DECREMENT THE STATUS COUNTER
13222 00004136 75BB                <1>         JNZ     short _R10              ; GO BACK FOR MORE
13223 00004138 800D[504E0100]20    <1>       OR     byte [DSKETTE_STATUS],BAD_NEC ; TOO MANY STATUS BYTES
13224 0000413F F9                  <1>       STC                         ; SET ERROR FLAG
13225                              <1>
13226                              <1> ;-----       RESULT OPERATION IS DONE
13227                              <1> POPRES:
13228 00004140 5F                  <1>       POP    eDI
13229 00004141 C3                  <1>       RETn                        ; RETURN WITH CARRY SET
13230                              <1>
13231                              <1> ;--------------------------------------------------------------------------
13232                              <1> ; READ_DSKCHNG
13233                              <1> ;     READS THE STATE OF THE DISK CHANGE LINE.
13234                              <1> ;
13235                              <1> ; ON ENTRY:  DI = DRIVE #
13236                              <1> ;
13237                              <1> ; ON EXIT:   DI = DRIVE #
13238                              <1> ;            ZF = 0 : DISK CHANGE LINE INACTIVE
13239                              <1> ;            ZF = 1 : DISK CHANGE LINE ACTIVE
13240                              <1> ;            AX,CX,DX DESTROYED
13241                              <1> ;--------------------------------------------------------------------------
13242                              <1> READ_DSKCHNG:
13243 00004142 E8A2FDFFFF          <1>       CALL   MOTOR_ON             ; TURN ON THE MOTOR IF OFF
13244 00004147 66BAF703            <1>       MOV    DX,03F7H             ; ADDRESS DIGITAL INPUT REGISTER
13245 0000414B EC                  <1>       IN     AL,DX                ; INPUT DIGITAL INPUT REGISTER
13246 0000414C A880                <1>       TEST   AL,DSK_CHG           ; CHECK FOR DISK CHANGE LINE ACTIVE
13247 0000414E C3                  <1>       RETn                        ; RETURN TO CALLER WITH ZERO FLAG SET
13248                              <1>
13249                              <1> ;--------------------------------------------------------------------------
13250                              <1> ; DRIVE_DET
13251                              <1> ;     DETERMINES WHETHER DRIVE IS 80 OR 40 TRACKS AND
13252                              <1> ;     UPDATES STATE INFORMATION ACCORDINGLY.
13253                              <1> ; ON ENTRY:  DI = DRIVE #
13254                              <1> ;--------------------------------------------------------------------------
13255                              <1> DRIVE_DET:
13256 0000414F E895FDFFFF          <1>       CALL   MOTOR_ON             ; TURN ON MOTOR IF NOT ALREADY ON
13257 00004154 E80AFFFFFF          <1>       CALL   RECAL                ; RECALIBRATE DRIVE
13258 00004159 7251                <1>       JC     short DD_BAC         ; ASSUME NO DRIVE PRESENT
```

```
13259 0000415B B530               <1>       MOV   CH,TRK_SLAP         ; SEEK TO TRACK 48
13260 0000415D E882FEFFFF         <1>       CALL  SEEK
13261 00004162 7248               <1>       JC    short DD_BAC        ; ERROR NO DRIVE
13262 00004164 B50B               <1>       MOV   CH,QUIET_SEEK+1           ; SEEK TO TRACK 10
13263                             <1> SK_GIN:
13264 00004166 FECD               <1>       DEC   CH                  ; DECREMENT TO NEXT TRACK
13265 00004168 6651               <1>       PUSH  CX                  ; SAVE TRACK
13266 0000416A E875FEFFFF         <1>       CALL  SEEK
13267 0000416F 723C               <1>       JC    short POP_BAC       ; POP AND RETURN
13268 00004171 B8[AD410000]       <1>       MOV   eAX, POP_BAC        ; LOAD NEC OUTPUT ERROR ADDRESS
13269 00004176 50                 <1>       PUSH  eAX
13270 00004177 B404               <1>       MOV   AH,SENSE_DRV_ST           ; SENSE DRIVE STATUS COMMAND BYTE
13271 00004179 E82CFEFFFF         <1>       CALL  NEC_OUTPUT          ; OUTPUT TO NEC
13272 0000417E 6689F8             <1>       MOV   AX,DI               ; AL = DRIVE
13273 00004181 88C4               <1>       MOV   AH,AL               ; AH = DRIVE
13274 00004183 E822FEFFFF         <1>       CALL  NEC_OUTPUT          ; OUTPUT TO NEC
13275 00004188 E85AFFFFFF         <1>       CALL  RESULTS                   ; GO GET STATUS
13276 0000418D 58                 <1>       POP   eAX                 ; THROW AWAY ERROR ADDRESS
13277 0000418E 6659               <1>       POP   CX                  ; RESTORE TRACK
13278 00004190 F605[514E0100]10   <1>       TEST  byte [NEC_STATUS], HOME   ; TRACK 0 ?
13279 00004197 74CD               <1>       JZ    short SK_GIN        ; GO TILL TRACK 0
13280 00004199 08ED               <1>       OR    CH,CH               ; IS HOME AT TRACK 0
13281 0000419B 7408               <1>       JZ    short IS_80         ; MUST BE 80 TRACK DRIVE
13282                             <1>
13283                             <1> ;     DRIVE IS A 360; SET DRIVE TO DETERMINED;
13284                             <1> ;     SET MEDIA TO DETERMINED AT RATE 250.
13285                             <1>
13286 0000419D 808F[5D4E0100]94   <1>       OR    byte [DSK_STATE+eDI], DRV_DET+MED_DET+RATE_250
13287 000041A4 C3                 <1>       RETn                      ; ALL INFORMATION SET
13288                             <1> IS_80:
13289 000041A5 808F[5D4E0100]01   <1>       OR    byte [DSK_STATE+eDI], TRK_CAPA ; SETUP 80 TRACK CAPABILITY
13290                             <1> DD_BAC:
13291 000041AC C3                 <1>       RETn
13292                             <1> POP_BAC:
13293 000041AD 6659               <1>       POP   CX                  ; THROW AWAY
13294 000041AF C3                 <1>       RETn
13295                             <1>
13296                             <1> fdc_int:
13297                             <1>         ; 30/07/2015
13298                             <1>         ; 16/02/2015
13299                             <1> ;int_0Eh: ; 11/12/2014
13300                             <1>
13301                             <1> ;--- HARDWARE INT 0EH -- ( IRQ LEVEL  6 ) -------------------------------------
13302                             <1> ; DISK_INT
13303                             <1> ;     THIS ROUTINE HANDLES THE DISKETTE INTERRUPT.
13304                             <1> ;
13305                             <1> ; ON EXIT:  THE INTERRUPT FLAG IS SET IN @SEEK_STATUS.
13306                             <1> ;--------------------------------------------------------------------------------
13307                             <1> DISK_INT_1:
13308                             <1>
13309 000041B0 6650               <1>       PUSH  AX                  ; SAVE WORK REGISTER
13310 000041B2 1E                 <1>       push  ds
13311 000041B3 66B81000           <1>       mov   ax, KDATA
13312 000041B7 8ED8               <1>       mov   ds, ax
13313 000041B9 800D[4D4E0100]80   <1>         OR      byte [SEEK_STATUS], INT_FLAG ; TURN ON INTERRUPT OCCURRED
13314 000041C0 B020               <1>       MOV     AL,EOI                  ; END OF INTERRUPT MARKER
13315 000041C2 E620               <1>       OUT   INTA00,AL           ; INTERRUPT CONTROL PORT
13316 000041C4 1F                 <1>       pop   ds
13317 000041C5 6658               <1>       POP   AX                  ; RECOVER REGISTER
13318 000041C7 CF                 <1>       IRETd                     ; RETURN FROM INTERRUPT
13319                             <1>
13320                             <1> ;--------------------------------------------------------------------------------
13321                             <1> ; DSKETTE_SETUP
13322                             <1> ;     THIS ROUTINE DOES A PRELIMINARY CHECK TO SEE WHAT TYPE OF
13323                             <1> ;     DISKETTE DRIVES ARE ATTACH TO THE SYSTEM.
13324                             <1> ;--------------------------------------------------------------------------------
13325                             <1>
13326                             <1> ; 29/05/2016 - TRDOS 386 (TRDOS v2.0)
13327                             <1>
13328                             <1> DSKETTE_SETUP:
13329                             <1>       ;PUSH  AX                 ; SAVE REGISTERS
13330                             <1>       ;PUSH  BX
13331                             <1>       ;PUSH  CX
13332 000041C8 52                <1>       PUSH   eDX
13333                             <1>       ;PUSH  DI
13334                             <1>       ;;PUSH DS
13335                             <1>       ; 14/12/2014
13336                             <1>       ;mov   word [DISK_POINTER], MD_TBL6
13337                             <1>       ;mov   [DISK_POINTER+2], cs
13338                             <1>       ;
13339                             <1>       ;OR    byte [RTC_WAIT_FLAG], 1   ; NO RTC WAIT, FORCE USE OF LOOP
13340 000041C9 31FF              <1>       XOR    eDI,eDI                   ; INITIALIZE DRIVE POINTER
13341 000041CB 66C705[5D4E0100]00- <1>     MOV    WORD [DSK_STATE],0  ; INITIALIZE STATES
13342 000041D3 00                <1>
13343 000041D4 8025[584E0100]33   <1>       AND    byte [LASTRATE],~(STRT_MSK+SEND_MSK) ; CLEAR START & SEND
13344 000041DB 800D[584E0100]C0   <1>       OR     byte [LASTRATE],SEND_MSK ; INITIALIZE SENT TO IMPOSSIBLE
13345 000041E2 C605[4D4E0100]00   <1>       MOV    byte [SEEK_STATUS],0      ; INDICATE RECALIBRATE NEEDED
13346 000041E9 C605[4F4E0100]00   <1>       MOV    byte [MOTOR_COUNT],0      ; INITIALIZE MOTOR COUNT
13347 000041F0 C605[4E4E0100]00   <1>       MOV    byte [MOTOR_STATUS],0     ; INITIALIZE DRIVES TO OFF STATE
13348 000041F7 C605[504E0100]00   <1>       MOV    byte [DSKETTE_STATUS],0   ; NO ERRORS
13349                             <1>       ;
13350                             <1>       ; 28/02/2015
13351                             <1>       ;mov   word [cfd], 100h
13352 000041FE E848F2FFFF         <1>       call   DSK_RESET
13353 00004203 5A                 <1>       pop    edx
13354 00004204 F8                 <1>       clc    ; 29/05/2016
13355 00004205 C3                 <1>       retn
13356                             <1>
13357                             <1> ;SUP0:
13358                             <1> ;     CALL   DRIVE_DET          ; DETERMINE DRIVE
13359                             <1> ;     CALL   XLAT_OLD           ; TRANSLATE STATE TO COMPATIBLE MODE
13360                             <1> ;     ; 02/01/2015
```

```
13361                           <1> ;     ;INC   DI                  ; POINT TO NEXT DRIVE
13362                           <1> ;     ;CMP   DI,MAX_DRV          ; SEE IF DONE
13363                           <1> ;     ;JNZ   short SUP0          ; REPEAT FOR EACH ORIVE
13364                           <1> ;       cmp     byte [fd1_type], 0
13365                           <1> ;     jna    short sup1
13366                           <1> ;     or     di, di
13367                           <1> ;     jnz    short sup1
13368                           <1> ;     inc    di
13369                           <1> ;       jmp     short SUP0
13370                           <1> ;sup1:
13371                           <1> ;     MOV    byte [SEEK_STATUS],0      ; FORCE RECALIBRATE
13372                           <1> ;     ;AND   byte [RTC_WAIT_FLAG],0FEH ; ALLOW FOR RTC WAIT
13373                           <1> ;     CALL   SETUP_END           ; VARIOUS CLEANUPS
13374                           <1> ;     ;;POP  DS                  ; RESTORE CALLERS REGISTERS
13375                           <1> ;     ;POP   DI
13376                           <1> ;     POP    eDX
13377                           <1> ;     ;POP   CX
13378                           <1> ;     ;POP   BX
13379                           <1> ;     ;POP   AX
13380                           <1> ;     RETn
13381                           <1>
13382                           <1> ;////////////////////////////////////////////////
13383                           <1> ;; END OF DISKETTE I/O ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
13384                           <1> ;
13385                           <1>
13386                           <1> int13h: ; 21/02/2015
13387 00004206 9C                <1>       pushfd
13388 00004207 0E                <1>       push   cs
13389 00004208 E843010000        <1>       call   DISK_IO
13390 0000420D C3                <1>       retn
13391                           <1>
13392                           <1> ;;;;;; DISK I/O ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;; 21/02/2015 ;;;
13393                           <1> ;////////////////////////////////////////////////////////////////////
13394                           <1>
13395                           <1> ; DISK I/O - Erdogan Tan (Retro UNIX 386 v1 project)
13396                           <1> ; 18/02/2016
13397                           <1> ; 17/02/2016
13398                           <1> ; 23/02/2015
13399                           <1> ; 21/02/2015 (unix386.s)
13400                           <1> ; 22/12/2014 - 14/02/2015 (dsectrm2.s)
13401                           <1> ;
13402                           <1> ; Original Source Code:
13403                           <1> ; DISK ----- 09/25/85 FIXED DISK BIOS
13404                           <1> ; (IBM PC XT Model 286 System BIOS Source Code, 04-21-86)
13405                           <1> ;
13406                           <1> ; Modifications: by reference of AWARD BIOS 1999 (D1A0622)
13407                           <1> ;            Source Code - ATORGS.ASM, AHDSK.ASM
13408                           <1> ;
13409                           <1>
13410                           <1>
13411                           <1> ;The wait for controller to be not busy is 10 seconds.
13412                           <1> ;10,000,000 / 30 = 333,333.  333,333 decimal = 051615h
13413                           <1> ;;WAIT_HDU_CTLR_BUSY_LO   equ    1615h
13414                           <1> ;;WAIT_HDU_CTLR_BUSY_HI   equ      05h
13415                           <1> WAIT_HDU_CTRL_BUSY_LH     equ    51615h  ;21/02/2015
13416                           <1>
13417                           <1> ;The wait for controller to issue completion interrupt is 10 seconds.
13418                           <1> ;10,000,000 / 30 = 333,333.  333,333 decimal = 051615h
13419                           <1> ;;WAIT_HDU_INT_LO  equ    1615h
13420                           <1> ;;WAIT_HDU_INT_HI  equ      05h
13421                           <1> WAIT_HDU_INT_LH           equ    51615h ; 21/02/2015
13422                           <1>
13423                           <1> ;The wait for Data request on read and write longs is
13424                           <1> ;2000 us. (?)
13425                           <1> ;;WAIT_HDU_DRQ_LO  equ    1000   ; 03E8h
13426                           <1> ;;WAIT_HDU_DRQ_HI  equ    0
13427                           <1> WAIT_HDU_DRQ_LH           equ    1000   ; 21/02/2015
13428                           <1>
13429                           <1> ; Port 61h (PORT_B)
13430                           <1> SYS1        equ    61h    ; PORT_B  (diskette.inc)
13431                           <1>
13432                           <1> ; 23/12/2014
13433                           <1> %define CMD_BLOCK       eBP-8  ; 21/02/2015
13434                           <1>
13435                           <1>
13436                           <1> ;--- INT 13H ----------------------------------------------------------------
13437                           <1> ;                                                                      :
13438                           <1> ; FIXED DISK I/O INTERFACE                                                :
13439                           <1> ;                                                                      :
13440                           <1> ;     THIS INTERFACE PROVIDES ACCESS TO 5 1/4" FIXED DISKS THROUGH           :
13441                           <1> ;     THE IBM FIXED DISK CONTROLLER.                                   :
13442                           <1> ;                                                                      :
13443                           <1> ;     THE  BIOS  ROUTINES  ARE  MEANT  TO  BE  ACCESSED  THROUGH       :
13444                           <1> ;     SOFTWARE  INTERRUPTS  ONLY.    ANY  ADDRESSES  PRESENT     IN          :
13445                           <1> ;     THESE  LISTINGS  ARE  INCLUDED    ONLY  FOR  COMPLETENESS,       :
13446                           <1> ;     NOT  FOR  REFERENCE.  APPLICATIONS   WHICH  REFERENCE  ANY       :
13447                           <1> ;     ABSOLUTE  ADDRESSES  WITHIN  THE  CODE SEGMENTS  OF  BIOS       :
13448                           <1> ;     VIOLATE  THE  STRUCTURE  AND  DESIGN  OF  BIOS.                  :
13449                           <1> ;                                                                      :
13450                           <1> ;------------------------------------------------------------------------:
13451                           <1> ;                                                                      :
13452                           <1> ; INPUT  (AH)= HEX COMMAND VALUE                                       :
13453                           <1> ;                                                                      :
13454                           <1> ;     (AH)= 00H  RESET DISK (DL = 80H,81H) / DISKETTE                  :
13455                           <1> ;     (AH)= 01H  READ THE STATUS OF THE LAST DISK OPERATION INTO (AL)       :
13456                           <1> ;                NOTE: DL < 80H - DISKETTE                             :
13457                           <1> ;                      DL > 80H - DISK                                 :
13458                           <1> ;     (AH)= 02H  READ THE DESIRED SECTORS INTO MEMORY                  :
13459                           <1> ;     (AH)= 03H  WRITE THE DESIRED SECTORS FROM MEMORY                 :
13460                           <1> ;     (AH)= 04H  VERIFY THE DESIRED SECTORS                            :
13461                           <1> ;     (AH)= 05H  FORMAT THE DESIRED TRACK                              :
13462                           <1> ;     (AH)= 06H  UNUSED                                                :
```

```
13463                                  <1> ;     (AH)= 07H  UNUSED                                                 :
13464                                  <1> ;     (AH)= 08H  RETURN THE CURRENT DRIVE PARAMETERS                          :
13465                                  <1> ;     (AH)= 09H  INITIALIZE DRIVE PAIR CHARACTERISTICS                  :
13466                                  <1> ;                INTERRUPT 41 POINTS TO DATA BLOCK FOR DRIVE 0          :
13467                                  <1> ;                INTERRUPT 46 POINTS TO DATA BLOCK FOR DRIVE 1          :
13468                                  <1> ;     (AH)= 0AH  READ LONG                                              :
13469                                  <1> ;     (AH)= 0BH  WRITE LONG  (READ & WRITE LONG ENCOMPASS 512 + 4 BYTES ECC) :
13470                                  <1> ;     (AH)= 0CH  SEEK                                                   :
13471                                  <1> ;     (AH)= 0DH  ALTERNATE DISK RESET (SEE DL)                          :
13472                                  <1> ;     (AH)= 0EH  UNUSED                                                 :
13473                                  <1> ;     (AH)= 0FH  UNUSED                                                 :
13474                                  <1> ;     (AH)= 10H  TEST DRIVE READY                                       :
13475                                  <1> ;     (AH)= 11H  RECALIBRATE                                            :
13476                                  <1> ;     (AH)= 12H  UNUSED                                                 :
13477                                  <1> ;     (AH)= 13H  UNUSED                                                 :
13478                                  <1> ;     (AH)= 14H  CONTROLLER INTERNAL DIAGNOSTIC                         :
13479                                  <1> ;     (AH)= 15H  READ DASD TYPE                                         :
13480                                  <1> ;                                                                       :
13481                                  <1> ;-----------------------------------------------------------------------------
13482                                  <1> ;                                                                       :
13483                                  <1> ;     REGISTERS USED FOR FIXED DISK OPERATIONS                          :
13484                                  <1> ;                                                                       :
13485                                  <1> ;           (DL)   -  DRIVE NUMBER     (80H-81H FOR DISK. VALUE CHECKED)  :
13486                                  <1> ;           (DH)   -  HEAD NUMBER         (0-15 ALLOWED, NOT VALUE CHECKED)  :
13487                                  <1> ;           (CH)   -  CYLINDER NUMBER  (0-1023, NOT VALUE CHECKED)(SEE CL):
13488                                  <1> ;           (CL)   -  SECTOR NUMBER    (1-17, NOT VALUE CHECKED)        :
13489                                  <1> ;                                                                       :
13490                                  <1> ;                     NOTE: HIGH 2 BITS OF CYLINDER NUMBER ARE PLACED     :
13491                                  <1> ;                          IN THE HIGH 2 BITS OF THE CL REGISTER        :
13492                                  <1> ;                          (10 BITS TOTAL)                              :
13493                                  <1> ;                                                                       :
13494                                  <1> ;           (AL)   -  NUMBER OF SECTORS (MAXIMUM POSSIBLE RANGE 1-80H,    :
13495                                  <1> ;                                       FOR READ/WRITE LONG 1-79H)      :
13496                                  <1> ;                                                                       :
13497                                  <1> ;           (ES:BX) -  ADDRESS OF BUFFER FOR READS AND WRITES,          :
13498                                  <1> ;                     (NOT REQUIRED FOR VERIFY)                         :
13499                                  <1> ;                                                                       :
13500                                  <1> ;           FORMAT (AH=5) ES:BX POINTS TO A 512 BYTE BUFFER. THE FIRST     :
13501                                  <1> ;                     2*(SECTORS/TRACK) BYTES CONTAIN F,N FOR EACH SECTOR.:
13502                                  <1> ;                     F = 00H FOR A GOOD SECTOR                         :
13503                                  <1> ;                         80H FOR A BAD SECTOR                          :
13504                                  <1> ;                     N = SECTOR NUMBER                                 :
13505                                  <1> ;                     FOR AN INTERLEAVE OF 2 AND 17 SECTORS/TRACK             :
13506                                  <1> ;                     THE TABLE SHOULD BE:                              :
13507                                  <1> ;                                                                       :
13508                                  <1> ;              DB     00H,01H,00H,0AH,00H,02H,00H,0BH,00H,03H,00H,0CH     :
13509                                  <1> ;              DB     00H,04H,00H,0DH,00H,05H,00H,0EH,00H,06H,00H,0FH     :
13510                                  <1> ;              DB     00H,07H,00H,10H,00H,08H,00H,11H,00H,09H           :
13511                                  <1> ;                                                                       :
13512                                  <1> ;-----------------------------------------------------------------------------
13513                                  <1>
13514                                  <1> ;-----------------------------------------------------------------------------
13515                                  <1> ; OUTPUT                                                                :
13516                                  <1> ;     AH = STATUS OF CURRENT OPERATION                                  :
13517                                  <1> ;          STATUS BITS ARE DEFINED IN THE EQUATES BELOW                 :
13518                                  <1> ;     CY = 0 SUCCESSFUL OPERATION (AH=0 ON RETURN)                      :
13519                                  <1> ;     CY = 1 FAILED OPERATION (AH HAS ERROR REASON)                     :
13520                                  <1> ;                                                                       :
13521                                  <1> ;     NOTE:  ERROR 11H  INDICATES THAT THE DATA READ HAD A RECOVERABLE      :
13522                                  <1> ;            ERROR WHICH WAS CORRECTED BY THE ECC ALGORITHM.  THE DATA      :
13523                                  <1> ;            IS PROBABLY GOOD,   HOWEVER THE BIOS ROUTINE INDICATES AN      :
13524                                  <1> ;            ERROR TO ALLOW THE CONTROLLING PROGRAM A CHANCE TO DECIDE      :
13525                                  <1> ;            FOR ITSELF.  THE  ERROR  MAY  NOT  RECUR  IF  THE DATA IS     :
13526                                  <1> ;            REWRITTEN.                                                     :
13527                                  <1> ;                                                                           :
13528                                  <1> ;     IF DRIVE PARAMETERS WERE REQUESTED (DL >= 80H),                   :
13529                                  <1> ;        INPUT:                                                         :
13530                                  <1> ;          (DL) = DRIVE NUMBER                                          :
13531                                  <1> ;          ; 27/05/2016 - TRDOS 386 (TRDOS v2.0)                                  :
13532                                  <1> ;          EBX = Buffer address for fixed disk parameters table (32 bytes)   :
13533                                  <1> ;        OUTPUT:                                                        :
13534                                  <1> ;          (DL) = NUMBER OF CONSECUTIVE ACKNOWLEDGING DRIVES ATTACHED (1-2)  :
13535                                  <1> ;               (CONTROLLER CARD ZERO TALLY ONLY)                       :
13536                                  <1> ;          (DH) = MAXIMUM USEABLE VALUE FOR HEAD NUMBER                 :
13537                                  <1> ;          (CH) = MAXIMUM USEABLE VALUE FOR CYLINDER NUMBER                   :
13538                                  <1> ;          (CL) = MAXIMUM USEABLE VALUE FOR SECTOR NUMBER               :
13539                                  <1> ;               AND CYLINDER NUMBER HIGH BITS                           :
13540                                  <1> ;                                                                       :
13541                                  <1> ;     IF READ DASD TYPE WAS REQUESTED,                                  :
13542                                  <1> ;                                                                       :
13543                                  <1> ;     AH = 0 - NOT PRESENT                                              :
13544                                  <1> ;          1 - DISKETTE - NO CHANGE LINE AVAILABLE                      :
13545                                  <1> ;          2 - DISKETTE - CHANGE LINE AVAILABLE                         :
13546                                  <1> ;          3 - FIXED DISK                                               :
13547                                  <1> ;                                                                       :
13548                                  <1> ;     CX,DX = NUMBER OF 512 BYTE BLOCKS WHEN AH = 3                     :
13549                                  <1> ;                                                                       :
13550                                  <1> ;     REGISTERS WILL BE PRESERVED EXCEPT WHEN THEY ARE USED TO RETURN        :
13551                                  <1> ;     INFORMATION.                                                      :
13552                                  <1> ;                                                                       :
13553                                  <1> ;     NOTE: IF AN ERROR IS REPORTED BY THE DISK CODE, THE APPROPRIATE        :
13554                                  <1> ;           ACTION IS TO RESET THE DISK, THEN RETRY THE OPERATION.            :
13555                                  <1> ;                                                                       :
13556                                  <1> ;-----------------------------------------------------------------------------
13557                                  <1>
13558                                  <1> SENSE_FAIL   EQU    0FFH          ; NOT IMPLEMENTED
13559                                  <1> NO_ERR       EQU    0E0H          ; STATUS ERROR/ERROR REGISTER=0
13560                                  <1> WRITE_FAULT  EQU    0CCH          ; WRITE FAULT ON SELECTED DRIVE
13561                                  <1> UNDEF_ERR    EQU    0BBH          ; UNDEFINED ERROR OCCURRED
13562                                  <1> NOT_RDY      EQU    0AAH          ; DRIVE NOT READY
13563                                  <1> TIME_OUT     EQU    80H           ; ATTACHMENT FAILED TO RESPOND
```

```
13564                                  <1> BAD_SEEK     EQU    40H          ; SEEK OPERATION FAILED
13565                                  <1> BAD_CNTLR    EQU    20H          ; CONTROLLER HAS FAILED
13566                                  <1> DATA_CORRECTED     EQU    11H          ; ECC CORRECTED DATA ERROR
13567                                  <1> BAD_ECC      EQU    10H          ; BAD ECC ON DISK READ
13568                                  <1> BAD_TRACK    EQU    0BH          ; NOT IMPLEMENTED
13569                                  <1> BAD_SECTOR   EQU    0AH          ; BAD SECTOR FLAG DETECTED
13570                                  <1> ;DMA_BOUNDARY      EQU    09H          ; DATA EXTENDS TOO FAR
13571                                  <1> INIT_FAIL    EQU    07H          ; DRIVE PARAMETER ACTIVITY FAILED
13572                                  <1> BAD_RESET    EQU    05H          ; RESET FAILED
13573                                  <1> ;RECORD_NOT_FND    EQU    04H          ; REQUESTED SECTOR NOT FOUND
13574                                  <1> ;BAD_ADDR_MARK     EQU    02H          ; ADDRESS MARK NOT FOUND
13575                                  <1> ;BAD_CMD     EQU    01H          ; BAD COMMAND PASSED TO DISK I/O
13576                                  <1>
13577                                  <1> ;--------------------------------------------------------
13578                                  <1> ;                                                      :
13579                                  <1> ; FIXED DISK PARAMETER TABLE                           :
13580                                  <1> ;  -  THE TABLE IS COMPOSED OF A BLOCK DEFINED AS:     :
13581                                  <1> ;                                                      :
13582                                  <1> ;  +0 (1 WORD) - MAXIMUM NUMBER OF CYLINDERS           :
13583                                  <1> ;  +2 (1 BYTE) - MAXIMUM NUMBER OF HEADS               :
13584                                  <1> ;  +3 (1 WORD) - NOT USED/SEE PC-XT                    :
13585                                  <1> ;  +5 (1 WORD) - STARTING WRITE PRECOMPENSATION CYL    :
13586                                  <1> ;  +7 (1 BYTE) - MAXIMUM ECC DATA BURST LENGTH         :
13587                                  <1> ;  +8 (1 BYTE) - CONTROL BYTE                          :
13588                                  <1> ;                BIT   7 DISABLE RETRIES -OR-          :
13589                                  <1> ;                BIT   6 DISABLE RETRIES               :
13590                                  <1> ;                BIT   3 MORE THAN 8 HEADS             :
13591                                  <1> ;  +9 (3 BYTES)- NOT USED/SEE PC-XT                    :
13592                                  <1> ; +12 (1 WORD) - LANDING ZONE                          :
13593                                  <1> ; +14 (1 BYTE) - NUMBER OF SECTORS/TRACK               :
13594                                  <1> ; +15 (1 BYTE) - RESERVED FOR FUTURE USE               :
13595                                  <1> ;                                                      :
13596                                  <1> ;      - TO DYNAMICALLY DEFINE A SET OF PARAMETERS     :
13597                                  <1> ;        BUILD A TABLE FOR UP TO 15 TYPES AND PLACE    :
13598                                  <1> ;        THE CORRESPONDING VECTOR INTO INTERRUPT 41     :
13599                                  <1> ;        FOR DRIVE 0 AND INTERRUPT 46 FOR DRIVE 1.     :
13600                                  <1> ;                                                      :
13601                                  <1> ;--------------------------------------------------------
13602                                  <1>
13603                                  <1> ;--------------------------------------------------------
13604                                  <1> ;                                                      :
13605                                  <1> ; HARDWARE SPECIFIC VALUES                             :
13606                                  <1> ;                                                      :
13607                                  <1> ;  -  CONTROLLER I/O PORT                              :
13608                                  <1> ;                                                      :
13609                                  <1> ;     > WHEN READ FROM:                                :
13610                                  <1> ;     HF_PORT+0 - READ DATA (FROM CONTROLLER TO CPU)         :
13611                                  <1> ;     HF_PORT+1 - GET ERROR REGISTER                   :
13612                                  <1> ;     HF_PORT+2 - GET SECTOR COUNT                     :
13613                                  <1> ;     HF_PORT+3 - GET SECTOR NUMBER                    :
13614                                  <1> ;     HF_PORT+4 - GET CYLINDER LOW                     :
13615                                  <1> ;     HF_PORT+5 - GET CYLINDER HIGH (2 BITS)           :
13616                                  <1> ;     HF_PORT+6 - GET SIZE/DRIVE/HEAD                  :
13617                                  <1> ;     HF_PORT+7 - GET STATUS REGISTER                  :
13618                                  <1> ;                                                      :
13619                                  <1> ;     > WHEN WRITTEN TO:                               :
13620                                  <1> ;     HF_PORT+0 - WRITE DATA (FROM CPU TO CONTROLLER)  :
13621                                  <1> ;     HF_PORT+1 - SET PRECOMPENSATION CYLINDER         :
13622                                  <1> ;     HF_PORT+2 - SET SECTOR COUNT                     :
13623                                  <1> ;     HF_PORT+3 - SET SECTOR NUMBER                    :
13624                                  <1> ;     HF_PORT+4 - SET CYLINDER LOW                     :
13625                                  <1> ;     HF_PORT+5 - SET CYLINDER HIGH (2 BITS)           :
13626                                  <1> ;     HF_PORT+6 - SET SIZE/DRIVE/HEAD                  :
13627                                  <1> ;     HF_PORT+7 - SET COMMAND REGISTER                 :
13628                                  <1> ;                                                      :
13629                                  <1> ;--------------------------------------------------------
13630                                  <1>
13631                                  <1> ;HF_PORT     EQU    01F0H  ; DISK PORT
13632                                  <1> ;HF1_PORT    equ    0170h
13633                                  <1> ;HF_REG_PORT EQU    03F6H
13634                                  <1> ;HF1_REG_PORT      equ    0376h
13635                                  <1>
13636                                  <1> HDC1_BASEPORT      equ    1F0h
13637                                  <1> HDC2_BASEPORT      equ    170h
13638                                  <1>
13639                                  <1> align 2
13640                                  <1>
13641                                  <1> ;-----             STATUS REGISTER
13642                                  <1>
13643                                  <1> ST_ERROR     EQU    00000001B    ;
13644                                  <1> ST_INDEX     EQU    00000010B    ;
13645                                  <1> ST_CORRCTD   EQU    00000100B    ; ECC CORRECTION SUCCESSFUL
13646                                  <1> ST_DRQ              EQU    00001000B    ;
13647                                  <1> ST_SEEK_COMPL       EQU    00010000B    ; SEEK COMPLETE
13648                                  <1> ST_WRT_FLT   EQU    00100000B    ; WRITE FAULT
13649                                  <1> ST_READY     EQU    01000000B    ;
13650                                  <1> ST_BUSY      EQU    10000000B    ;
13651                                  <1>
13652                                  <1> ;-----             ERROR REGISTER
13653                                  <1>
13654                                  <1> ERR_DAM      EQU    00000001B    ; DATA ADDRESS MARK NOT FOUND
13655                                  <1> ERR_TRK_0    EQU    00000010B    ; TRACK 0 NOT FOUND ON RECAL
13656                                  <1> ERR_ABORT    EQU    00000100B    ; ABORTED COMMAND
13657                                  <1> ;            EQU    00001000B    ; NOT USED
13658                                  <1> ERR_ID              EQU    00010000B    ; ID NOT FOUND
13659                                  <1> ;            EQU    00100000B    ; NOT USED
13660                                  <1> ERR_DATA_ECC EQU    01000000B
13661                                  <1> ERR_BAD_BLOCK       EQU    10000000B
13662                                  <1>
13663                                  <1>
13664                                  <1> RECAL_CMD    EQU    00010000B    ; DRIVE RECAL(10H)
13665                                  <1> READ_CMD     EQU    00100000B    ;      READ   (20H)
```

```
13666                                  <1> WRITE_CMD    EQU    00110000B    ;      WRITE  (30H)
13667                                  <1> VERIFY_CMD   EQU    01000000B    ;      VERIFY (40H)
13668                                  <1> FMTTRK_CMD   EQU    01010000B    ; FORMAT TRACK      (50H)
13669                                  <1> INIT_CMD     EQU    01100000B    ;   INITIALIZE      (60H)
13670                                  <1> SEEK_CMD     EQU    01110000B    ;      SEEK   (70H)
13671                                  <1> DIAG_CMD     EQU    10010000B    ; DIAGNOSTIC (90H)
13672                                  <1> SET_PARM_CMD EQU    10010001B    ; DRIVE PARMS(91H)
13673                                  <1> NO_RETRIES   EQU    00000001B    ; CHD MODIFIER      (01H)
13674                                  <1> ECC_MODE     EQU    00000010B    ; CMD MODIFIER      (02H)
13675                                  <1> BUFFER_MODE  EQU    00001000B    ; CMD MODIFIER      (08H)
13676                                  <1>
13677                                  <1> ;MAX_FILE    EQU    2
13678                                  <1> ;S_MAX_FILE  EQU    2
13679                                  <1> MAX_FILE     equ    4            ; 22/12/2014
13680                                  <1> S_MAX_FILE   equ    4            ; 22/12/2014
13681                                  <1>
13682                                  <1> DELAY_1      EQU    25H          ; DELAY FOR OPERATION COMPLETE
13683                                  <1> DELAY_2      EQU    0600H        ; DELAY FOR READY
13684                                  <1> DELAY_3      EQU    0100H        ; DELAY FOR DATA REQUEST
13685                                  <1>
13686                                  <1> HF_FAIL      EQU    08H          ; CMD MODIFIER      (08H)
13687                                  <1>
13688                                  <1> ;-----              COMMAND BLOCK REFERENCE
13689                                  <1>
13690                                  <1> ;CMD_BLOCK      EQU     BP-8            ; @CMD_BLOCK REFERENCES BLOCK HEAD IN SS
13691                                  <1>                                 ;  (BP) POINTS TO COMMAND BLOCK TAIL
13692                                  <1>                                 ;     AS DEFINED BY THE "ENTER" PARMS
13693                                  <1> ; 19/12/2014
13694                                  <1> ORG_VECTOR   equ    4*13h        ; INT 13h vector
13695                                  <1> DISK_VECTOR  equ    4*40h        ; INT 40h vector (for floppy disks)
13696                                  <1> ;HDISK_INT   equ    4*76h        ; Primary HDC - Hardware interrupt (IRQ14)
13697                                  <1> ;HDISK_INT1  equ    4*76h        ; Primary HDC - Hardware interrupt (IRQ14)
13698                                  <1> ;HDISK_INT2  equ    4*77h        ; Secondary HDC - Hardware interrupt (IRQ15)
13699                                  <1> ;HF_TBL_VEC  equ    4*41h        ; Pointer to 1st fixed disk parameter table
13700                                  <1> ;HF1_TBL_VEC equ    4*46h        ; Pointer to 2nd fixed disk parameter table
13701                                  <1>
13702                                  <1> align 2
13703                                  <1>
13704                                  <1> ;--------------------------------------------------------------
13705                                  <1> ; FIXED DISK I/O SETUP                                        :
13706                                  <1> ;                                                    :
13707                                  <1> ;  -  ESTABLISH TRANSFER VECTORS FOR THE FIXED DISK       :
13708                                  <1> ;  -  PERFORM POWER ON DIAGNOSTICS                        :
13709                                  <1> ;     SHOULD AN ERROR OCCUR A "1701" MESSAGE IS DISPLAYED       :
13710                                  <1> ;                                                    :
13711                                  <1> ;--------------------------------------------------------------
13712                                  <1>
13713                                  <1> ; 29/05/2016 - TRDOS 386 (TRDOS v2.0)
13714                                  <1>
13715                                  <1> DISK_SETUP:
13716                                  <1>       ;CLI
13717                                  <1>       ;;MOV  AX,ABS0                  ; GET ABSOLUTE SEGMENT
13718                                  <1>       ;xor   ax,ax
13719                                  <1>       ;MOV   DS,AX                    ; SET SEGMENT REGISTER
13720                                  <1>       ;MOV   AX, [ORG_VECTOR]         ; GET DISKETTE VECTOR
13721                                  <1>       ;MOV   [DISK_VECTOR],AX         ;  INTO INT 40H
13722                                  <1>       ;MOV   AX, [ORG_VECTOR+2]
13723                                  <1>       ;MOV   [DISK_VECTOR+2],AX
13724                                  <1>       ;MOV   word [ORG_VECTOR],DISK_IO ; FIXED DISK HANDLER
13725                                  <1>       ;MOV   [ORG_VECTOR+2],CS
13726                                  <1>       ; 1st controller (primary master, slave)   - IRQ 14
13727                                  <1>       ;;MOV  word [HDISK_INT],HD_INT          ; FIXED DISK INTERRUPT
13728                                  <1>       ;mov   word [HDISK_INT1],HD_INT  ;
13729                                  <1>       ;;MOV  [HDISK_INT+2],CS
13730                                  <1>       ;mov   [HDISK_INT1+2],CS
13731                                  <1>       ; 2nd controller (secondary master, slave) - IRQ 15
13732                                  <1>       ;mov   word [HDISK_INT2],HD1_INT ;
13733                                  <1>       ;mov   [HDISK_INT2+2],CS
13734                                  <1>       ;
13735                                  <1>       ;;MOV  word [HF_TBL_VEC],HD0_DPT ; PARM TABLE DRIVE 80
13736                                  <1>       ;;MOV  word [HF_TBL_VEC+2],DPT_SEGM
13737                                  <1>       ;;MOV  word [HF1_TBL_VEC],HD1_DPT; PARM TABLE DRIVE 81
13738                                  <1>       ;;MOV  word [HF1_TBL_VEC+2],DPT_SEGM
13739                                  <1>       ;push  cs
13740                                  <1>       ;pop   ds
13741                                  <1>       ;mov   word [HDPM_TBL_VEC],HD0_DPT      ; PARM TABLE DRIVE 80h
13742                                  <1>       ;mov   word [HDPM_TBL_VEC+2],DPT_SEGM
13743 0000420E C705[684E0100]0000-     <1>       mov    dword [HDPM_TBL_VEC], (DPT_SEGM*16)+HD0_DPT
13744 00004216 0900                    <1>
13745                                  <1>       ;mov   word [HDPS_TBL_VEC],HD1_DPT      ; PARM TABLE DRIVE 81h
13746                                  <1>       ;mov   word [HDPS_TBL_VEC+2],DPT_SEGM
13747 00004218 C705[6C4E0100]2000-     <1>       mov    dword [HDPS_TBL_VEC], (DPT_SEGM*16)+HD1_DPT
13748 00004220 0900                    <1>
13749                                  <1>       ;mov   word [HDSM_TBL_VEC],HD2_DPT      ; PARM TABLE DRIVE 82h
13750                                  <1>       ;mov   word [HDSM_TBL_VEC+2],DPT_SEGM
13751 00004222 C705[704E0100]4000-     <1>       mov    dword [HDSM_TBL_VEC], (DPT_SEGM*16)+HD2_DPT
13752 0000422A 0900                    <1>
13753                                  <1>       ;mov   word [HDSS_TBL_VEC],HD3_DPT      ; PARM TABLE DRIVE 83h
13754                                  <1>       ;mov   word [HDSS_TBL_VEC+2],DPT_SEGM
13755 0000422C C705[744E0100]6000-     <1>       mov    dword [HDSS_TBL_VEC], (DPT_SEGM*16)+HD3_DPT
13756 00004234 0900                    <1>
13757                                  <1>       ;
13758                                  <1>       ;;IN   AL,INTB01           ; TURN ON SECOND INTERRUPT CHIP
13759                                  <1>       ;;;AND AL,0BFH
13760                                  <1>       ;;and  al, 3Fh                   ; enable IRQ 14 and IRQ 15
13761                                  <1>       ;;;JMP $+2
13762                                  <1>       ;;IODELAY
13763                                  <1>       ;;OUT  INTB01,AL
13764                                  <1>       ;;IODELAY
13765                                  <1>       ;;IN   AL,INTA01           ; LET INTERRUPTS PASS THRU TO
13766                                  <1>       ;;AND  AL,0FBH             ;  SECOND CHIP
13767                                  <1>       ;;;JMP $+2
```

```
13768                                   <1>       ;;IODELAY
13769                                   <1>       ;;OUT  INTA01,AL
13770                                   <1>       ;
13771                                   <1>       ;STI
13772                                   <1>       ;;PUSH DS                  ; MOVE ABS0 POINTER TO
13773                                   <1>       ;;POP  ES                  ; EXTRA SEGMENT POINTER
13774                                   <1>       ;;;CALL      DDS                ; ESTABLISH DATA SEGMENT
13775                                   <1>       ;;MOV  byte [DISK_STATUS1],0     ; RESET THE STATUS INDICATOR
13776                                   <1>       ;;MOV  byte [HF_NUM],0           ; ZERO NUMBER OF FIXED DISKS
13777                                   <1>       ;;MOV  byte [CONTROL_BYTE],0
13778                                   <1>       ;;MOV  byte [PORT_OFF],0   ; ZERO CARD OFFSET
13779                                   <1>       ; 20/12/2014 - private code by Erdogan Tan
13780                                   <1>                  ; (out of original PC-AT, PC-XT BIOS code)
13781                                   <1>       ;mov   si, hd0_type
13782 00004236 BE[F85C0000]            <1>       mov    esi, hd0_type
13783                                   <1>       ;mov   cx, 4
13784 0000423B B904000000              <1>       mov    ecx, 4
13785                                   <1> hde_l:
13786 00004240 AC                      <1>       lodsb
13787 00004241 3C80                    <1>       cmp    al, 80h                   ; 8?h = existing
13788 00004243 7206                    <1>       jb     short _L4
13789 00004245 FE05[644E0100]          <1>       inc    byte [HF_NUM]       ; + 1 hard (fixed) disk drives
13790                                   <1> _L4: ; 26/02/2015
13791 0000424B E2F3                    <1>       loop   hde_l
13792                                   <1> ;_L4:                          ; 0 <= [HF_NUM] =< 4
13793                                   <1> ;L4:
13794                                   <1>       ;
13795                                   <1>       ;; 31/12/2014 - cancel controller diagnostics here
13796                                   <1>       ;;;mov       cx, 3  ; 26/12/2014 (Award BIOS 1999)
13797                                   <1>       ;;mov  cl, 3
13798                                   <1>       ;;
13799                                   <1>       ;;MOV  DL,80H              ; CHECK THE CONTROLLER
13800                                   <1> ;;hdc_dl:
13801                                   <1>       ;;MOV  AH,14H              ; USE CONTROLLER DIAGNOSTIC COMMAND
13802                                   <1>       ;;INT  13H                 ; CALL BIOS WITH DIAGNOSTIC COMMAND
13803                                   <1>       ;;;JC  short CTL_ERRX            ; DISPLAY ERROR MESSAGE IF BAD RETURN
13804                                   <1>       ;;;jc  short POD_DONE ;22/12/2014
13805                                   <1>       ;;jnc  short hdc_reset0
13806                                   <1>       ;;loop hdc_dl
13807                                   <1>       ;;; 27/12/2014
13808                                   <1>       ;;stc
13809                                   <1>       ;;retn
13810                                   <1>       ;
13811                                   <1> ;;hdc_reset0:
13812                                   <1>       ; 18/01/2015
13813 0000424D 8A0D[644E0100]          <1>       mov    cl, [HF_NUM]
13814 00004253 20C9                    <1>       and    cl, cl
13815 00004255 740E                    <1>       jz     short POD_DONE
13816                                   <1>       ;
13817 00004257 B27F                    <1>       mov    dl, 7Fh
13818                                   <1> hdc_reset1:
13819 00004259 FEC2                    <1>       inc    dl
13820                                   <1>       ;; 31/12/2015
13821                                   <1>       ;;push dx
13822                                   <1>       ;;push cx
13823                                   <1>       ;;push ds
13824                                   <1>       ;;sub  ax, ax
13825                                   <1>       ;;mov  ds, ax
13826                                   <1>       ;;MOV  AX, [TIMER_LOW]           ; GET START TIMER COUNTS
13827                                   <1>       ;;pop  ds
13828                                   <1>       ;;MOV  BX,AX
13829                                   <1>       ;;ADD  AX,6*182            ; 60 SECONDS* 18.2
13830                                   <1>       ;;MOV  CX,AX
13831                                   <1>       ;;mov  word [wait_count], 0      ; 22/12/2014 (reset wait counter)
13832                                   <1>       ;;
13833                                   <1>       ;; 31/12/2014 - cancel HD_RESET_1
13834                                   <1>       ;;CALL HD_RESET_1          ; SET UP DRIVE 0, (1,2,3)
13835                                   <1>       ;;pop  cx
13836                                   <1>       ;;pop  dx
13837                                   <1>       ;;
13838                                   <1>       ; 18/01/2015
13839 0000425B B40D                    <1>       mov    ah, 0Dh ; ALTERNATE RESET
13840                                   <1>       ;int   13h
13841 0000425D E8A4FFFFFF              <1>       call   int13h
13842 00004262 E2F5                    <1>       loop   hdc_reset1
13843 00004264 F8                      <1>       clc    ; 29/05/2016
13844                                   <1> POD_DONE:
13845 00004265 C3                      <1>       RETn
13846                                   <1>
13847                                   <1> ;;-----      POD_ERROR
13848                                   <1>
13849                                   <1> ;;CTL_ERRX:
13850                                   <1> ;     ;MOV   SI,OFFSET F1782     ; CONTROLLER ERROR
13851                                   <1> ;     ;CALL  SET_FAIL            ; DO NOT IPL FROM DISK
13852                                   <1> ;     ;CALL  E_MSG               ; DISPLAY ERROR AND SET (BP) ERROR FLAG
13853                                   <1> ;     ;JMP   short POD_DONE
13854                                   <1>
13855                                   <1> ;;HD_RESET_1:
13856                                   <1> ;;    ;PUSH  BX                  ; SAVE TIMER LIMITS
13857                                   <1> ;;    ;PUSH  CX
13858                                   <1> ;;RES_1: MOV AH,09H              ; SET DRIVE PARAMETERS
13859                                   <1> ;;    INT    13H
13860                                   <1> ;;    JC     short RES_2
13861                                   <1> ;;    MOV    AH,11H              ; RECALIBRATE DRIVE
13862                                   <1> ;;    INT    13H
13863                                   <1> ;;    JNC    short RES_CK        ; DRIVE OK
13864                                   <1> ;;RES_2: ;CALL     POD_TCHK            ; CHECK TIME OUT
13865                                   <1> ;;    cmp    word [wait_count], 6*182 ; waiting time (in timer ticks)
13866                                   <1> ;;                                ; (30 seconds)
13867                                   <1> ;;    ;cmc
13868                                   <1> ;;    ;JNC   short RES_1
13869                                   <1> ;;    jb     short RES_1
```

```
13870                           <1> ;;;RES_FL: ;MOV    SI,OFFSET F1781     ; INDICATE DISK 1 FAILURE;
13871                           <1> ;;    ;TEST  DL,1
13872                           <1> ;;    ;JNZ   RES_E1
13873                           <1> ;;    ;MOV   SI,OFFSET F1780     ; INDICATE DISK 0 FAILURE
13874                           <1> ;;    ;CALL  SET_FAIL            ; DO NOT TRY TO IPL DISK 0
13875                           <1> ;;    ;JMP   SHORT RES_E1
13876                           <1> ;;RES_ER: ; 22/12/2014
13877                           <1> ;;RES_OK:
13878                           <1> ;;    ;POP   CX                  ; RESTORE TIMER LIMITS
13879                           <1> ;;    ;POP   BX
13880                           <1> ;;    RETn
13881                           <1> ;;
13882                           <1> ;;RES_RS: MOV      AH,00H              ; RESET THE DRIVE
13883                           <1> ;;    INT    13H
13884                           <1> ;;RES_CK: MOV      AH,08H              ; GET MAX CYLINDER,HEAD,SECTOR
13885                           <1> ;;    MOV    BL,DL               ; SAVE DRIVE CODE
13886                           <1> ;;    INT    13H
13887                           <1> ;;    JC     short RES_ER
13888                           <1> ;;    MOV    [NEC_STATUS],CX     ; SAVE MAX CYLINDER, SECTOR
13889                           <1> ;;    MOV    DL,BL               ; RESTORE DRIVE CODE
13890                           <1> ;;RES_3: MOV AX,0401H            ; VERIFY THE LAST SECTOR
13891                           <1> ;;    INT    13H
13892                           <1> ;;    JNC    short RES_OK        ; VERIFY OK
13893                           <1> ;;    CMP    AH,BAD_SECTOR       ; OK ALSO IF JUST ID READ
13894                           <1> ;;    JE     short RES_OK
13895                           <1> ;;    CMP    AH,DATA_CORRECTED
13896                           <1> ;;    JE     short RES_OK
13897                           <1> ;;    CMP    AH,BAD_ECC
13898                           <1> ;;    JE     short RES_OK
13899                           <1> ;;    ;CALL  POD_TCHK            ; CHECK FOR TIME OUT
13900                           <1> ;;    cmp    word [wait_count], 6*182 ; waiting time (in timer ticks)
13901                           <1> ;;                                ; (60 seconds)
13902                           <1> ;;    cmc
13903                           <1> ;;    JC     short RES_ER        ; FAILED
13904                           <1> ;;    MOV    CX,[NEC_STATUS]     ; GET SECTOR ADDRESS, AND CYLINDER
13905                           <1> ;;    MOV    AL,CL               ; SEPARATE OUT SECTOR NUMBER
13906                           <1> ;;    AND    AL,3FH
13907                           <1> ;;    DEC    AL                  ; TRY PREVIOUS ONE
13908                           <1> ;;    JZ     short RES_RS        ; WE'VE TRIED ALL SECTORS ON TRACK
13909                           <1> ;;    AND    CL,0C0H             ; KEEP CYLINDER BITS
13910                           <1> ;;    OR     CL,AL               ; MERGE SECTOR WITH CYLINDER BITS
13911                           <1> ;;    MOV    [NEC_STATUS],CX     ; SAVE CYLINDER, NEW SECTOR NUMBER
13912                           <1> ;;    JMP    short RES_3         ; TRY AGAIN
13913                           <1> ;;;RES_ER: MOV     SI,OFFSET F1791     ; INDICATE DISK 1 ERROR
13914                           <1> ;;    ;TEST  DL,1
13915                           <1> ;;    ;JNZ   short RES_E1
13916                           <1> ;;    ;MOV   SI,OFFSET F1790     ; INDICATE DISK 0 ERROR
13917                           <1> ;;;RES_E1:
13918                           <1> ;;    ;CALL  E_MSG               ; DISPLAY ERROR AND SET (BP) ERROR FLAG
13919                           <1> ;;;RES_OK:
13920                           <1> ;;    ;POP   CX                  ; RESTORE TIMER LIMITS
13921                           <1> ;;    ;POP   BX
13922                           <1> ;;    ;RETn
13923                           <1> ;
13924                           <1> ;;SET_FAIL:
13925                           <1> ;     ;MOV   AX,X*(CMOS_DIAG+NMI)      ; GET CMOS ERROR BYTE
13926                           <1> ;     ;CALL  CMOS_READ
13927                           <1> ;     ;OR    AL,HF_FAIL          ; SET DO NOT IPL FROM DISK FLAG
13928                           <1> ;     ;XCHG  AH,AL               ; SAVE IT
13929                           <1> ;     ;CALL  CMOS_WRITE          ; PUT IT OUT
13930                           <1> ;     ;RETn
13931                           <1> ;
13932                           <1> ;;POD_TCHK:                       ; CHECK FOR 30 SECOND TIME OUT
13933                           <1> ;     ;POP   AX                  ; SAVE RETURN
13934                           <1> ;     ;POP   CX                  ; GET TIME OUT LIMITS
13935                           <1> ;     ;POP   BX
13936                           <1> ;     ;PUSH  BX                  ; AND SAVE THEM AGAIN
13937                           <1> ;     ;PUSH  CX
13938                           <1> ;     ;PUSH  AX
13939                           <1> ;     ;push  ds
13940                           <1> ;     ;xor   ax, ax
13941                           <1> ;     ;mov   ds, ax              ; RESTORE RETURN
13942                           <1> ;     ;MOV   AX, [TIMER_LOW]           ; AX = CURRENT TIME
13943                           <1> ;     ;                          ; BX = START TIME
13944                           <1> ;     ;                          ; CX = END TIME
13945                           <1> ;     ;pop   ds
13946                           <1> ;     ;CMP   BX,CX
13947                           <1> ;     ;JB    short TCHK1         ; START < END
13948                           <1> ;     ;CMP   BX,AX
13949                           <1> ;     ;JB    short TCHKG         ; END < START < CURRENT
13950                           <1> ;     ;JMP   SHORT TCHK2         ; END, CURRENT < START
13951                           <1> ;;TCHK1: CMP AX,BX
13952                           <1> ;;    JB     short TCHKNG        ; CURRENT < START < END
13953                           <1> ;;TCHK2: CMP AX,CX
13954                           <1> ;;    JB     short TCHKG         ; START < CURRENT < END
13955                           <1> ;;                               ; OR CURRENT < END < START
13956                           <1> ;;TCHKNG: STC                         ; CARRY SET INDICATES TIME OUT
13957                           <1> ;;    RETn
13958                           <1> ;;TCHKG: CLC                     ; INDICATE STILL TIME
13959                           <1> ;;    RETn
13960                           <1> ;;
13961                           <1> ;;int_13h:
13962                           <1>
13963                           <1> ;----------------------------------------
13964                           <1> ;     FIXED DISK BIOS ENTRY POINT      :
13965                           <1> ;----------------------------------------
13966                           <1>
13967                           <1> ; 15/01/2017
13968                           <1> ; 14/01/2017
13969                           <1> ; 07/01/2017
13970                           <1> ; 02/01/2017
13971                           <1> ; 01/06/2016
```

```
13972                                  <1> ; 16/05/2016, 27/05/2016, 28/05/2016, 29/05/2016
13973                                  <1> ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
13974                                  <1> int33h:  ; DISK I/O
13975                                  <1>       ; 29/05/2016
13976 00004266 80642408FE             <1>       and   byte [esp+8], 11111110b  ; clear carry bit of eflags register
13977                                  <1>       ; 16/05/2016
13978 0000426B 1E                      <1>       push  ds
13979 0000426C 53                      <1>       push  ebx ; user's buffer address (virtual)
13980 0000426D 66BB1000                <1>       mov   bx, KDATA ; System (Kernel's) data segment
13981 00004271 8EDB                    <1>       mov   ds, bx
13982                                  <1>
13983                                  <1>       ;;15/01/2017
13984                                  <1>       ; 14/01/2017
13985                                  <1>       ; 02/01/2017
13986                                  <1>       ;;mov  byte [intflg], 33h  ; disk io interrupt
13987                                  <1>       ;pop  ebx
13988                                  <1>       ;mov  [user_buffer], ebx
13989                                  <1>
13990 00004273 8F05[585B0100]          <1>       pop   dword [user_buffer] ; 01/06/2016
13991                                  <1>
13992 00004279 C605[8E540100]00        <1>       mov   byte [scount], 0 ; sector count for transfer
13993 00004280 80FC03                  <1>       cmp   ah, 03h ; chs write
13994 00004283 7744                    <1>       ja    short int33h_2
13995 00004285 7407                    <1>       je    short int33h_0
13996 00004287 80FC02                  <1>       cmp   ah, 02h ; chs read
13997 0000428A 726A                    <1>       jb    short int33h_5
13998 0000428C EB63                    <1>       jmp   short int33h_4
13999                                  <1> int33h_0:
14000                                  <1>       ; transfer user's buffer content to sector buffer
14001 0000428E 51                      <1>       push  ecx
14002 0000428F 0FB6C8                  <1>       movzx ecx, al
14003                                  <1> int33h_1:
14004 00004292 56                      <1>       push  esi
14005 00004293 8B35[585B0100]          <1>       mov   esi, [user_buffer]
14006                                  <1>       ; esi = user's buffer address (virtual, ebx)
14007 00004299 57                      <1>       push  edi
14008 0000429A 06                      <1>       push  es
14009 0000429B 50                      <1>       push  eax
14010 0000429C 66B81000                <1>       mov   ax, KDATA
14011 000042A0 8EC0                    <1>       mov   es, ax
14012 000042A2 BF00000700              <1>       mov   edi, Cluster_Buffer
14013 000042A7 C1E109                  <1>       shl   ecx, 9 ; * 512
14014 000042AA E81DA60000              <1>       call  transfer_from_user_buffer
14015 000042AF 58                      <1>       pop   eax
14016 000042B0 07                      <1>       pop   es
14017 000042B1 5F                      <1>       pop   edi
14018 000042B2 5E                      <1>       pop   esi
14019 000042B3 59                      <1>       pop   ecx
14020 000042B4 7340                    <1>       jnc   short int33h_5
14021 000042B6 8B1D[585B0100]          <1>       mov   ebx, [user_buffer] ; 01/06/2016
14022 000042BC 1F                      <1>       pop   ds
14023                                  <1>
14024                                  <1>       ;;15/01/2017
14025                                  <1>       ; 02/01/2017
14026                                  <1>       ;cli
14027                                  <1>       ;;mov  byte [ss:intflg], 0 ; 07/01/2017
14028                                  <1>       ;
14029                                  <1>       ; (*) 29/05/2016
14030                                  <1>       ; (*) retf 4 ; skip eflags on stack
14031                                  <1>
14032                                  <1>       ; 29/05/2016 -set carry flag on stack-
14033                                  <1>       ; [esp] = EIP
14034                                  <1>       ; [esp+4] = CS
14035                                  <1>       ; [esp+8] = E-FLAGS
14036 000042BD 804C240801             <1>       or    byte [esp+8], 1  ; set carry bit of eflags register
14037                                  <1>       ; [esp+12] = ESP (user)
14038                                  <1>       ; [esp+16] = SS (User)
14039 000042C2 B8FF000000              <1>       mov   eax, 0FFh ; Unknown error !?
14040                                  <1>       ;iretd
14041 000042C7 EB79                    <1>       jmp   short int33h_7  ; 07/01/2017
14042                                  <1>
14043                                  <1>       ; (*) 29/05/2016 - 'ref 4' intruction causes to stack fault
14044                                  <1>       ; (OUTER-PRIVILEGE-LEVEL)
14045                                  <1>       ; INTEL 80386 PROGRAMMER'S REFERENCE MANUAL 1986
14046                                  <1>       ; // RETF instruction:
14047                                  <1>       ;
14048                                  <1>       ; IF OperandMode=32 THEN
14049                                  <1>       ;    Load CS:EIP from stack;
14050                                  <1>       ;    Set CS RPL to CPL;
14051                                  <1>       ;    Increment eSP by 8 plus the immediate offset if it exists;
14052                                  <1>       ;    Load SS:eSP from stack;
14053                                  <1>       ; ELSE (* OperandMode=16 *)
14054                                  <1>       ;    Load CS:IP from stack;
14055                                  <1>       ;    Set CS RPL to CPL;
14056                                  <1>       ;    Increment eSP by 4 plus the immediate offset if it exists;
14057                                  <1>       ;    Load SS:eSP from stack;
14058                                  <1>       ; FI;
14059                                  <1>       ;
14060                                  <1>       ; //
14061                                  <1>
14062                                  <1> int33h_2:
14063 000042C9 80FC05                  <1>       cmp   ah, 05h ; format track
14064 000042CC 770A                    <1>       ja    short int33h_3
14065 000042CE 7226                    <1>       jb    short int33h_5
14066 000042D0 51                      <1>       push  ecx
14067 000042D1 B901000000              <1>       mov   ecx, 1
14068 000042D6 EBBA                    <1>       jmp   short int33h_1
14069                                  <1> int33h_3:
14070 000042D8 80FC1C                  <1>       cmp   ah, 1Ch ; LBA write
14071 000042DB 7719                    <1>       ja    short int33h_5
14072 000042DD 74AF                    <1>       je    short int33h_0
14073 000042DF 80FC1B                  <1>       cmp   ah, 1Bh ; LBA read
```

```
14074 000042E2 740D                <1>       je    short int33h_4
14075 000042E4 80FC08              <1>       cmp   ah, 08h ; get disk parameters
14076 000042E7 750D                <1>       jne   short int33h_5
14077                              <1>       ; 01/06/2016
14078 000042E9 8B1D[585B0100]      <1>       mov   ebx, [user_buffer] ; user's buffer address
14079 000042EF EB0A                <1>       jmp   short int33h_6
14080                              <1> int33h_4:
14081 000042F1 A2[8E540100]        <1>       mov   byte [scount], al ; <= 128 sectors
14082                              <1> int33h_5:
14083 000042F6 BB00000700          <1>       mov   ebx, Cluster_Buffer ; max. 65536 bytes
14084                              <1>                                 ; buf. addr: 70000h
14085                              <1>       ;mov  byte [ClusterBuffer_Valid], 0
14086                              <1> int33h_6:
14087 000042FB 1F                  <1>       pop   ds
14088 000042FC 9C                  <1>       pushfd
14089 000042FD 0E                  <1>       push  cs
14090 000042FE E84D000000          <1>       call  DISK_IO
14091 00004303 2E8B1D[585B0100]    <1>       mov   ebx, [CS:user_buffer] ; 01/06/2016
14092 0000430A 723D                <1>       jc    short int33h_9
14093                              <1>       ;
14094 0000430C 2E803D[8E540100]00  <1>       cmp   byte [CS:scount], 0
14095 00004314 762C                <1>       jna   short int33h_7
14096                              <1>       ; transfer sector buffer content to user's buffer
14097 00004316 06                  <1>       push  es
14098 00004317 1E                  <1>       push  ds
14099 00004318 50                  <1>       push  eax
14100 00004319 66B81000            <1>       mov   ax, KDATA
14101 0000431D 8ED8                <1>       mov   ds, ax
14102 0000431F 8EC0                <1>       mov   es, ax
14103 00004321 51                  <1>       push  ecx
14104 00004322 56                  <1>       push  esi
14105 00004323 57                  <1>       push  edi
14106 00004324 0FB60D[8E540100]    <1>       movzx ecx, byte [scount]
14107 0000432B C1E109              <1>       shl   ecx, 9 ; * 512 bytes
14108 0000432E 89DF                <1>       mov   edi, ebx ; user's buffer address
14109 00004330 BE00000700          <1>       mov   esi, Cluster_Buffer
14110 00004335 E848A50000          <1>       call  transfer_to_user_buffer
14111 0000433A 5F                  <1>       pop   edi
14112 0000433B 5E                  <1>       pop   esi
14113 0000433C 59                  <1>       pop   ecx
14114 0000433D 58                  <1>       pop   eax
14115 0000433E 1F                  <1>       pop   ds
14116 0000433F 07                  <1>       pop   es
14117 00004340 7202                <1>       jc    short int33h_8
14118                              <1> int33h_7:
14119 00004342 FA                  <1>       cli
14120                              <1>       ;;15/01/2017
14121                              <1>       ;;mov  byte [ss:intflg], 0 ; 07/01/2017
14122                              <1>       ; cf = 0  ; use eflags which is in stack
14123 00004343 CF                  <1>       iretd
14124                              <1> int33h_8:
14125 00004344 B8FF000000          <1>       mov   eax, 0FFh ; Unknown error !?
14126                              <1> int33h_9:
14127                              <1>       ; cf = 1
14128                              <1>
14129                              <1>       ; (*) 29/05/2016
14130                              <1>       ; (*) retf 4 ; skip eflags on stack
14131                              <1>       ; Note: This 'retf 4' was wrong, -it was causing
14132                              <1>       ;       to stack errors in ring 3-
14133                              <1>       ;     POP sequence of 'retf 4' is as
14134                              <1>       ;       "eip, cs, eflags, esp, ss, +4 bytes"
14135                              <1>         ;       it is not as "eip, cs, +4 bytes, esp, ss" !
14136                              <1>
14137                              <1>       ; 29/05/2016 -set carry flag on stack-
14138 00004349 804C240801          <1>       or    byte [esp+8], 1  ; set carry bit of eflags register
14139                              <1>       ;iretd
14140 0000434E EBF2                <1>       jmp   short int33h_7 ; 07/01/2017
14141                              <1>
14142                              <1> ; 29/05/2016
14143                              <1> ; 27/05/2016 - TRDOS 386 (TRDOS v2.0)
14144                              <1>
14145                              <1> DISK_IO:
14146 00004350 80FA80              <1>       CMP   DL,80H              ; TEST FOR FIXED DISK DRIVE
14147                              <1>       ;JAE  short A1            ; YES, HANDLE HERE
14148                              <1>       ;;;INT 40H                ; DISKETTE HANDLER
14149                              <1>       ;;call int40h
14150 00004353 0F8222F0FFFF        <1>       jb    DISKETTE_IO_1
14151                              <1> ;RET_2:
14152                              <1>       ;RETf 2                   ; BACK TO CALLER
14153                              <1> ;     retf  4
14154                              <1> A1:
14155 00004359 FB                  <1>       STI                       ; ENABLE INTERRUPTS
14156                              <1>       ;; 04/01/2015
14157                              <1>       ;;OR  AH,AH
14158                              <1>       ;;JNZ short A2
14159                              <1>       ;;INT 40H                 ; RESET NEC WHEN AH=0
14160                              <1>       ;;SUB AH,AH
14161 0000435A 80FA83              <1>       CMP   DL,(80H + S_MAX_FILE - 1)
14162                              <1>       ;JA   short RET_2
14163 0000435D 7616                <1>       jna   short _A0
14164                              <1>       ; 29/05/2016
14165 0000435F 1E                  <1>       push  ds
14166 00004360 6650                <1>       push  ax
14167 00004362 66B81000            <1>       mov   ax, KDATA
14168 00004366 8ED8                <1>       mov   ds, ax
14169 00004368 6658                <1>       pop   ax
14170 0000436A B4AA                <1>         mov     ah, 0AAh        ; Hard disk drive not ready !
14171                              <1>                               ; (Programmer's guide to AMIBIOS, 1992)
14172 0000436C 8825[634E0100]      <1>       mov     byte [DISK_STATUS1], ah
14173 00004372 1F                  <1>       pop   ds
14174 00004373 EB38                <1>       jmp   short RET_2
14175                              <1> _A0:
```

```
14176                                  <1>       ; 18/01/2015
14177 00004375 08E4                    <1>       or     ah,ah
14178 00004377 743A                    <1>       jz     short A4
14179 00004379 80FC0D                  <1>       cmp    ah, 0Dh      ; Alternate reset
14180 0000437C 7504                    <1>       jne    short A2
14181 0000437E 28E4                    <1>       sub    ah,ah  ; Reset
14182 00004380 EB31                    <1>       jmp    short A4
14183                                  <1> A2:
14184 00004382 80FC08                  <1>       CMP    AH,08H              ; GET PARAMETERS IS A SPECIAL CASE
14185                                  <1>       ;JNZ   short A3
14186                                  <1>         ;JMP    GET_PARM_N
14187 00004385 0F8431030000            <1>       je     GET_PARM_N
14188 0000438B 80FC15                  <1> A3:   CMP    AH,15H              ; READ DASD TYPE IS ALSO
14189                                  <1>       ;JNZ   short A4
14190                                  <1>         ;JMP    READ_DASD_TYPE
14191 0000438E 0F84DA020000            <1>         je     READ_DASD_TYPE
14192                                  <1>       ; 02/02/2015
14193 00004394 80FC1D                  <1>       cmp    ah, 1Dh                  ;(Temporary for Retro UNIX 386 v1)
14194                                  <1>       ; 12/01/2015
14195 00004397 F5                      <1>       cmc
14196 00004398 7319                    <1>       jnc    short A4
14197                                  <1> int33h_bad_cmd:
14198                                  <1>       ; 16/05/2016
14199                                  <1>       ; 30/01/2015
14200                                  <1>       ; 29/05/2016
14201 0000439A 1E                      <1>       push   ds
14202 0000439B 6650                    <1>       push   ax
14203 0000439D 66B81000                <1>       mov    ax, KDATA
14204 000043A1 8ED8                    <1>       mov    ds, ax
14205 000043A3 6658                    <1>       pop    ax
14206 000043A5 B401                    <1>       mov    ah, BAD_CMD
14207 000043A7 8825[634E0100]          <1>       mov     [DISK_STATUS1], ah ; BAD_CMD  ; COMMAND ERROR
14208                                  <1>         ;jmp short RET_2
14209                                  <1> RET_2:
14210                                  <1>       ; (*) 29/05/2016
14211                                  <1>       ; (*) retf 4
14212 000043AD 804C240801             <1>       or     byte [esp+8], 1 ; set carry bit of eflags register
14213 000043B2 CF                      <1>       iretd
14214                                  <1> A4:                                ; SAVE REGISTERS DURING OPERATION
14215 000043B3 C8080000                <1>       ENTER  8,0                  ; SAVE (BP) AND MAKE ROOM FOR @CMD_BLOCK
14216 000043B7 53                      <1>       PUSH   eBX                  ;  IN THE STACK, THE COMMAND BLOCK IS:
14217 000043B8 51                      <1>       PUSH   eCX                  ;   @CMD_BLOCK == BYTE PTR [BP]-8
14218 000043B9 52                      <1>       PUSH   eDX
14219 000043BA 1E                      <1>       PUSH   DS
14220 000043BB 06                      <1>       PUSH   ES
14221 000043BC 56                      <1>       PUSH   eSI
14222 000043BD 57                      <1>       PUSH   eDI
14223                                  <1>       ;;04/01/2015
14224                                  <1>       ;;OR   AH,AH                ; CHECK FOR RESET
14225                                  <1>       ;;JNZ  short A5
14226                                  <1>       ;;MOV  DL,80H               ; FORCE DRIVE 80 FOR RESET
14227                                  <1> ;;A5:
14228                                  <1>       ;push  cs
14229                                  <1>       ;pop   ds
14230                                  <1>       ; 21/02/2015
14231 000043BE 6650                    <1>       push   ax
14232 000043C0 66B81000                <1>       mov    ax, KDATA
14233 000043C4 8ED8                    <1>       mov    ds, ax
14234 000043C6 8EC0                    <1>       mov    es, ax
14235 000043C8 6658                    <1>       pop    ax
14236 000043CA E88D000000              <1>       CALL   DISK_IO_CONT         ; PERFORM THE OPERATION
14237                                  <1>       ;;CALL DDS                  ; ESTABLISH SEGMENT
14238 000043CF 8A25[634E0100]          <1>       MOV    AH,[DISK_STATUS1]    ; GET STATUS FROM OPERATION
14239                                  <1>       ;(*) CMP AH,1               ; SET THE CARRY FLAG TO INDICATE
14240                                  <1>       ;(*) CMC                    ; SUCCESS OR FAILURE
14241 000043D5 5F                      <1>       POP    eDI                  ; RESTORE REGISTERS
14242 000043D6 5E                      <1>       POP    eSI
14243 000043D7 07                      <1>         POP     ES
14244 000043D8 1F                      <1>         POP     DS
14245 000043D9 5A                      <1>       POP    eDX
14246 000043DA 59                      <1>       POP    eCX
14247 000043DB 5B                      <1>       POP    eBX
14248 000043DC C9                      <1>       LEAVE                       ; ADJUST (SP) AND RESTORE (BP)
14249                                  <1>       ;RETf  2                    ; THROW AWAY SAVED FLAGS
14250                                  <1>       ; (*) 29/05/2016
14251                                  <1>       ; (*) retf 4
14252 000043DD 80FC01                  <1>       cmp    ah, 1
14253 000043E0 7205                    <1>       jc     short _A5
14254 000043E2 804C240801              <1>       or     byte [esp+8], 1 ; set carry bit of eflags register
14255                                  <1> _A5:
14256 000043E7 CF                      <1>       iretd
14257                                  <1>
14258                                  <1> ; 21/02/2015
14259                                  <1> ;       dw --> dd
14260                                  <1> D1:                                ; FUNCTION TRANSFER TABLE
14261 000043E8 [AA450000]              <1>       dd     DISK_RESET           ; 000H
14262 000043EC [21460000]              <1>       dd     RETURN_STATUS        ; 001H
14263 000043F0 [2E460000]              <1>       dd     DISK_READ            ; 002H
14264 000043F4 [37460000]              <1>       dd     DISK_WRITE           ; 003H
14265 000043F8 [40460000]              <1>       dd     DISK_VERF            ; 004H
14266 000043FC [58460000]              <1>       dd     FMT_TRK              ; 005H
14267 00004400 [A0450000]              <1>       dd     BAD_COMMAND          ; 006H FORMAT BAD SECTORS
14268 00004404 [A0450000]              <1>       dd     BAD_COMMAND          ; 007H FORMAT DRIVE
14269 00004408 [A0450000]              <1>       dd     BAD_COMMAND          ; 008H RETURN PARAMETERS
14270 0000440C [43470000]              <1>       dd     INIT_DRV             ; 009H
14271 00004410 [A2470000]              <1>       dd     RD_LONG              ; 00AH
14272 00004414 [AB470000]              <1>       dd     WR_LONG              ; 00BH
14273 00004418 [B4470000]              <1>       dd     DISK_SEEK            ; 00CH
14274 0000441C [AA450000]              <1>       dd     DISK_RESET           ; 00DH
14275 00004420 [A0450000]              <1>       dd     BAD_COMMAND          ; 00EH READ BUFFER
14276 00004424 [A0450000]              <1>       dd     BAD_COMMAND          ; 00FH WRITE BUFFER
14277 00004428 [DC470000]              <1>       dd     TST_RDY              ; 010H
```

```
14278 0000442C [00480000]          <1>       dd     HDISK_RECAL          ; 011H
14279 00004430 [A0450000]          <1>       dd     BAD_COMMAND          ; 012H MEMORY DIAGNOSTIC
14280 00004434 [A0450000]          <1>       dd     BAD_COMMAND          ; 013H DRIVE DIAGNOSTIC
14281 00004438 [36480000]          <1>       dd     CTLR_DIAGNOSTIC      ; 014H CONTROLLER DIAGNOSTIC
14282                              <1>       ; 02/02/2015 (Temporary - Retro UNIX 386 v1 - DISK I/O test)
14283 0000443C [A0450000]          <1>       dd     BAD_COMMAND          ; 015h
14284 00004440 [A0450000]          <1>       dd     BAD_COMMAND          ; 016h
14285 00004444 [A0450000]          <1>       dd     BAD_COMMAND          ; 017h
14286 00004448 [A0450000]          <1>       dd     BAD_COMMAND          ; 018h
14287 0000444C [A0450000]          <1>       dd     BAD_COMMAND          ; 019h
14288 00004450 [A0450000]          <1>       dd     BAD_COMMAND          ; 01Ah
14289 00004454 [2E460000]          <1>       dd     DISK_READ            ; 01Bh ; LBA read
14290 00004458 [37460000]          <1>       dd     DISK_WRITE           ; 01Ch ; LBA write
14291                              <1> D1L     EQU    $ - D1
14292                              <1>
14293                              <1> DISK_IO_CONT:
14294                              <1>       ;;CALL DDS                  ; ESTABLISH SEGMENT
14295 0000445C 80FC01              <1>       CMP    AH,01H               ; RETURN STATUS
14296                              <1>       ;;JNZ  short SU0
14297                              <1>         ;;JMP    RETURN_STATUS
14298 0000445F 0F84BC010000        <1>       je     RETURN_STATUS
14299                              <1> SU0:
14300 00004465 C605[634E0100]00    <1>       MOV    byte [DISK_STATUS1],0     ; RESET THE STATUS INDICATOR
14301                              <1>       ;;PUSH BX                   ; SAVE DATA ADDRESS
14302                              <1>       ;mov   si, bx ;; 14/02/2015
14303 0000446C 89DE                <1>       mov    esi, ebx ; 21/02/2015
14304 0000446E 8A1D[644E0100]      <1>       MOV    BL,[HF_NUM]          ; GET NUMBER OF DRIVES
14305                              <1>       ;; 04/01/2015
14306                              <1>       ;;PUSH AX
14307 00004474 80E27F              <1>       AND    DL,7FH               ; GET DRIVE AS 0 OR 1
14308                              <1>                                   ; (get drive number as 0 to 3)
14309 00004477 38D3                <1>       CMP    BL,DL
14310                              <1>         ;;JBE   BAD_COMMAND_POP         ; INVALID DRIVE
14311 00004479 0F8621010000        <1>         jbe     BAD_COMMAND ;; 14/02/2015
14312                              <1>         ;
14313                              <1>       ;;03/01/2015
14314 0000447F 29DB                <1>       sub    ebx, ebx
14315 00004481 88D3                <1>       mov    bl, dl
14316                              <1>       ;sub   bh, bh
14317 00004483 883D[784E0100]      <1>       mov    [LBAMode], bh       ; 0
14318                              <1>       ;;test byte [bx+hd0_type], 1     ; LBA ready ?
14319                              <1>       ;test  byte [ebx+hd0_type], 1
14320                              <1>       ;jz    short su1            ; no
14321                              <1>       ;inc   byte [LBAMode]
14322                              <1> ;su1:
14323                              <1>       ; 21/02/2015 (32 bit modification)
14324                              <1>       ;04/01/2015
14325 00004489 6650                <1>       push   ax ; ***
14326                              <1>       ;PUSH  ES ; **
14327 0000448B 6652                <1>       PUSH   DX ; *
14328 0000448D 6650                <1>       push   ax
14329 0000448F E888060000          <1>       CALL   GET_VEC              ; GET DISK PARAMETERS
14330                              <1>       ; 02/02/2015
14331                              <1>       ;mov   ax, [ES:BX+16] ; I/O port base address (1F0h, 170h)
14332 00004494 668B4310            <1>       mov    ax, [ebx+16]
14333 00004498 66A3[E85C0000]      <1>       mov    [HF_PORT], ax
14334                              <1>       ;mov   dx, [ES:BX+18] ; control port address (3F6h, 376h)
14335 0000449E 668B5312            <1>       mov    dx, [ebx+18]
14336 000044A2 668915[EA5C0000]    <1>       mov    [HF_REG_PORT], dx
14337                              <1>       ;mov   al, [ES:BX+20] ; head register upper nibble (A0h,B0h,E0h,F0h)
14338 000044A9 8A4314              <1>       mov    al, [ebx+20]
14339                              <1>       ; 23/02/2015
14340 000044AC A840                <1>       test   al, 40h       ; LBA bit (bit 6)
14341 000044AE 7406                <1>       jz     short su1
14342 000044B0 FE05[784E0100]      <1>       inc    byte [LBAMode] ; 1
14343                              <1> su1:
14344 000044B6 C0E804              <1>       shr    al, 4
14345 000044B9 2401                <1>       and    al, 1
14346 000044BB A2[EC5C0000]        <1>       mov    [hf_m_s], al
14347                              <1>       ;
14348                              <1>       ; 03/01/2015
14349                              <1>       ;MOV   AL,byte [ES:BX+8]   ; GET CONTROL BYTE MODIFIER
14350 000044C0 8A4308              <1>       mov    al, [ebx+8]
14351                              <1>       ;MOV   DX,[HF_REG_PORT]    ; Device Control register
14352 000044C3 EE                  <1>       OUT    DX,AL                ; SET EXTRA HEAD OPTION
14353                              <1>                                   ; Control Byte:  (= 08h, here)
14354                              <1>                                   ; bit 0 - 0
14355                              <1>                                   ; bit 1 - nIEN (1 = disable irq)
14356                              <1>                                   ; bit 2 - SRST (software RESET)
14357                              <1>                                   ; bit 3 - use extra heads (8 to 15)
14358                              <1>                                   ;         -always set to 1-
14359                              <1>                                   ; (bits 3 to 7 are reserved
14360                              <1>                                   ;          for ATA devices)
14361 000044C4 8A25[654E0100]      <1>       MOV    AH,[CONTROL_BYTE]   ; SET EXTRA HEAD OPTION IN
14362 000044CA 80E4C0              <1>       AND    AH,0C0H             ; CONTROL BYTE
14363 000044CD 08C4                <1>       OR     AH,AL
14364 000044CF 8825[654E0100]      <1>       MOV    [CONTROL_BYTE],AH
14365                              <1>       ; 04/01/2015
14366 000044D5 6658                <1>       pop    ax
14367 000044D7 665A                <1>       pop    dx ; * ;; 14/02/2015
14368 000044D9 20E4                <1>       and    ah, ah ; Reset function ?
14369 000044DB 7507                <1>       jnz    short su2
14370                              <1>       ;;pop  dx ; * ;; 14/02/2015
14371                              <1>       ;pop   es ; **
14372 000044DD 6658                <1>       pop    ax ; ***
14373                              <1>       ;;pop  bx
14374 000044DF E9C6000000          <1>         jmp     DISK_RESET
14375                              <1> su2:
14376 000044E4 803D[784E0100]00    <1>       cmp    byte [LBAMode], 0
14377 000044EB 7661                <1>       jna    short su3
14378                              <1>       ;
14379                              <1>       ; 02/02/2015 (LBA read/write function calls)
```

```
14380 000044ED 80FC1B              <1>       cmp    ah, 1Bh
14381 000044F0 720B                <1>       jb     short lbarw1
14382 000044F2 80FC1C              <1>       cmp    ah, 1Ch
14383 000044F5 775C                <1>       ja     short invldfnc
14384                              <1>       ;;pop  dx ; * ; 14/02/2015
14385                              <1>       ;mov   ax, cx ; Lower word of LBA address (bits 0-15)
14386 000044F7 89C8                <1>       mov    eax, ecx ; LBA address (21/02/2015)
14387                              <1>       ;; 14/02/2015
14388 000044F9 88D1                <1>       mov    cl, dl ; 14/02/2015
14389                              <1>       ;;mov  dx, bx
14390                              <1>       ;mov   dx, si ; higher word of LBA address (bits 16-23)
14391                              <1>       ;;mov  bx, di
14392                              <1>       ;mov   si, di ; Buffer offset
14393 000044FB EB31                <1>       jmp    short lbarw2
14394                              <1> lbarw1:
14395                              <1>       ; convert CHS to LBA
14396                              <1>       ;
14397                              <1>       ; LBA calculation - AWARD BIOS - 1999 - AHDSK.ASM
14398                              <1>       ; LBA = "# of Heads" * Sectors/Track * Cylinder + Head * Sectors/Track
14399                              <1>       ;      + Sector - 1
14400 000044FD 6652                <1>       push   dx ; * ;; 14/02/2015
14401                              <1>       ;xor   dh, dh
14402 000044FF 31D2                <1>       xor    edx, edx
14403                              <1>       ;mov   dl, [ES:BX+14]      ; sectors per track (logical)
14404 00004501 8A530E              <1>       mov    dl, [ebx+14]
14405                              <1>       ;xor   ah, ah
14406 00004504 31C0                <1>       xor    eax, eax
14407                              <1>       ;mov   al, [ES:BX+2]; heads (logical)
14408 00004506 8A4302              <1>       mov    al, [ebx+2]
14409 00004509 FEC8                <1>       dec    al
14410 0000450B 6640                <1>       inc    ax           ; 0 =  256
14411 0000450D 66F7E2              <1>       mul    dx
14412                              <1>              ; AX = # of Heads" * Sectors/Track
14413 00004510 6689CA              <1>       mov    dx, cx
14414                              <1>       ;and   cx, 3Fh       ; sector  (1 to 63)
14415 00004513 83E13F              <1>       and    ecx, 3fh
14416 00004516 86D6                <1>       xchg   dl, dh
14417 00004518 C0EE06              <1>       shr    dh, 6
14418                              <1>              ; DX = cylinder (0 to 1023)
14419                              <1>       ;mul   dx
14420                              <1>              ; DX:AX = # of Heads" * Sectors/Track * Cylinder
14421 0000451B F7E2                <1>       mul    edx
14422 0000451D FEC9                <1>       dec    cl  ; sector - 1
14423                              <1>       ;add   ax, cx
14424                              <1>       ;adc   dx, 0
14425                              <1>              ; DX:AX = # of Heads" * Sectors/Track * Cylinder + Sector -1
14426 0000451F 01C8                <1>       add    eax, ecx
14427 00004521 6659                <1>       pop    cx ; * ; ch = head, cl = drive number (zero based)
14428                              <1>       ;push  dx
14429                              <1>       ;push  ax
14430 00004523 50                  <1>       push   eax
14431                              <1>       ;mov   al, [ES:BX+14]      ; sectors per track (logical)
14432 00004524 8A430E              <1>       mov    al, [ebx+14]
14433 00004527 F6E5                <1>       mul    ch
14434                              <1>              ;  AX = Head * Sectors/Track
14435 00004529 6699                <1>         cwd
14436                              <1>       ;pop   dx
14437 0000452B 5A                  <1>       pop    edx
14438                              <1>       ;add   ax, dx
14439                              <1>       ;pop   dx
14440                              <1>       ;adc   dx, 0 ; add carry bit
14441 0000452C 01D0                <1>       add    eax, edx
14442                              <1> lbarw2:
14443 0000452E 29D2                <1>       sub    edx, edx ; 21/02/2015
14444 00004530 88CA                <1>       mov    dl, cl ; 21/02/2015
14445 00004532 C645F800            <1>         mov     byte [CMD_BLOCK], 0 ; Features Register
14446                              <1>                         ; NOTE: Features register (1F1h, 171h)
14447                              <1>                         ; is not used for ATA device R/W functions.
14448                              <1>                         ; It is old/obsolete 'write precompensation'
14449                              <1>                         ; register and error register
14450                              <1>                         ; for old ATA/IDE devices.
14451                              <1>       ; 18/01/2014
14452                              <1>       ;mov   ch, [hf_m_s] ; Drive 0 (master) or 1 (slave)
14453 00004536 8A0D[EC5C0000]      <1>       mov    cl, [hf_m_s]
14454                              <1>       ;shl   ch, 4        ; bit 4 (drive bit)
14455                              <1>       ;or    ch, 0E0h     ; bit 5 = 1
14456                              <1>                         ; bit 6 = 1 = LBA mode
14457                              <1>                         ; bit 7 = 1
14458 0000453C 80C90E              <1>       or     cl, 0Eh ; 1110b
14459                              <1>       ;and   dh, 0Fh             ; LBA byte 4 (bits 24 to 27)
14460 0000453F 25FFFFFF0F          <1>       and    eax, 0FFFFFFFh
14461 00004544 C1E11C              <1>       shl    ecx, 28 ; 21/02/2015
14462                              <1>       ;or    dh, ch
14463 00004547 09C8                <1>       or     eax, ecx
14464                              <1>       ;;mov  [CMD_BLOCK+2], al ; LBA byte 1 (bits 0 to 7)
14465                              <1>                               ; (Sector Number Register)
14466                              <1>       ;;mov  [CMD_BLOCK+3], ah ; LBA byte 2 (bits 8 to 15)
14467                              <1>                               ; (Cylinder Low Register)
14468                              <1>       ;mov   [CMD_BLOCK+2], ax ; LBA byte 1, 2
14469                              <1>       ;mov   [CMD_BLOCK+4], dl ; LBA byte 3 (bits 16 to 23)
14470                              <1>                               ; (Cylinder High Register)
14471                              <1>       ;;mov  [CMD_BLOCK+5], dh ; LBA byte 4 (bits 24 to 27)
14472                              <1>                               ; (Drive/Head Register)
14473                              <1>
14474                              <1>       ;mov   [CMD_BLOCK+4], dx ; LBA byte 4, LBA & DEV select bits
14475 00004549 8945FA              <1>       mov    [CMD_BLOCK+2], eax ; 21/02/2015
14476                              <1>       ;14/02/2015
14477                              <1>       ;mov   dl, cl ; Drive number (INIT_DRV)
14478 0000454C EB38                <1>       jmp    short su4
14479                              <1> su3:
14480                              <1>       ; 02/02/2015
14481                              <1>       ; (Temporary functions 1Bh & 1Ch are not valid for CHS mode)
```

```
14482 0000454E 80FC14              <1>       cmp    ah, 14h
14483 00004551 7604                <1>       jna    short chsfnc
14484                              <1> invldfnc:
14485                              <1>         ; 14/02/2015
14486                              <1>       ;pop   es ; **
14487 00004553 6658                <1>         pop     ax ; ***
14488                              <1>         ;jmp     short BAD_COMMAND_POP
14489 00004555 EB49                <1>         jmp     short BAD_COMMAND
14490                              <1> chsfnc:
14491                              <1>       ;MOV   AX,[ES:BX+5]        ; GET WRITE PRE-COMPENSATION CYLINDER
14492 00004557 668B4305            <1>       mov    ax, [ebx+5]
14493 0000455B 66C1E802            <1>       SHR    AX,2
14494 0000455F 8845F8              <1>       MOV    [CMD_BLOCK],AL
14495                              <1>       ;;MOV  AL,[ES:BX+8]        ; GET CONTROL BYTE MODIFIER
14496                              <1>       ;;PUSH DX
14497                              <1>       ;;MOV  DX,[HF_REG_PORT]
14498                              <1>       ;;OUT  DX,AL               ; SET EXTRA HEAD OPTION
14499                              <1>       ;;POP  DX ; *
14500                              <1>       ;;POP  ES ; **
14501                              <1>       ;;MOV  AH,[CONTROL_BYTE]   ; SET EXTRA HEAD OPTION IN
14502                              <1>       ;;AND  AH,0C0H             ; CONTROL BYTE
14503                              <1>       ;;OR   AH,AL
14504                              <1>       ;;MOV  [CONTROL_BYTE],AH
14505                              <1>       ;
14506 00004562 88C8                <1>       MOV    AL,CL               ; GET SECTOR NUMBER
14507 00004564 243F                <1>       AND    AL,3FH
14508 00004566 8845FA              <1>       MOV    [CMD_BLOCK+2],AL
14509 00004569 886DFB              <1>       MOV    [CMD_BLOCK+3],CH    ; GET CYLINDER NUMBER
14510 0000456C 88C8                <1>       MOV    AL,CL
14511 0000456E C0E806              <1>       SHR    AL,6
14512 00004571 8845FC              <1>       MOV    [CMD_BLOCK+4],AL    ; CYLINDER HIGH ORDER 2 BITS
14513                              <1>       ;;05/01/2015
14514                              <1>       ;;MOV  AL,DL               ; DRIVE NUMBER
14515 00004574 A0[EC5C0000]        <1>       mov    al, [hf_m_s]
14516 00004579 C0E004              <1>       SHL    AL,4
14517 0000457C 80E60F              <1>       AND    DH,0FH              ; HEAD NUMBER
14518 0000457F 08F0                <1>       OR     AL,DH
14519                              <1>       ;OR    AL,80H or 20H
14520 00004581 0CA0                <1>       OR     AL,80h+20h          ; ECC AND 512 BYTE SECTORS
14521 00004583 8845FD              <1>       MOV    [CMD_BLOCK+5],AL    ; ECC/SIZE/DRIVE/HEAD
14522                              <1> su4:
14523                              <1>       ;POP   ES ; **
14524                              <1>         ;; 14/02/2015
14525                              <1>         ;;POP   AX
14526                              <1>         ;;MOV   [CMD_BLOCK+1],AL        ; SECTOR COUNT
14527                              <1>         ;;PUSH  AX
14528                              <1>         ;;MOV   AL,AH                   ; GET INTO LOW BYTE
14529                              <1>         ;;XOR   AH,AH                   ; ZERO HIGH BYTE
14530                              <1>         ;;SAL   AX,1                    ; *2 FOR TABLE LOOKUP
14531 00004586 6658                <1>         pop     ax ; ***
14532 00004588 8845F9              <1>         mov     [CMD_BLOCK+1], al
14533 0000458B 29DB                <1>         sub  ebx, ebx
14534 0000458D 88E3                <1>       mov     bl, ah
14535                              <1>         ;xor     bh, bh
14536                              <1>         ;sal     bx, 1
14537 0000458F 66C1E302            <1>         sal  bx, 2  ; 32 bit offset (21/02/2015)
14538                              <1>       ;;MOV   SI,AX                   ; PUT INTO SI FOR BRANCH
14539                              <1>         ;;CMP   AX,D1L                  ; TEST WITHIN RANGE
14540                              <1>         ;;JNB   short BAD_COMMAND_POP
14541                              <1>         ;cmp     bx, D1L
14542 00004593 83FB74              <1>       cmp    ebx, D1L
14543 00004596 7308                <1>       jnb    short BAD_COMMAND
14544                              <1>         ;xchg    bx, si
14545 00004598 87DE                <1>         xchg ebx, esi
14546                              <1>       ;;;POP AX                  ; RESTORE AX
14547                              <1>       ;;;POP BX                  ; AND DATA ADDRESS
14548                              <1>
14549                              <1>       ;;PUSH CX
14550                              <1>       ;;PUSH AX                  ; ADJUST ES:BX
14551                              <1>       ;MOV   CX,BX               ; GET 3 HIGH ORDER NIBBLES OF BX
14552                              <1>       ;SHR   CX,4
14553                              <1>       ;MOV   AX,ES
14554                              <1>       ;ADD   AX,CX
14555                              <1>       ;MOV   ES,AX
14556                              <1>       ;AND   BX,000FH            ; ES:BX CHANGED TO ES:000X
14557                              <1>       ;;POP  AX
14558                              <1>       ;;POP  CX
14559                              <1>       ;;JMP  word [CS:SI+D1]
14560                              <1>       ;jmp   word [SI+D1]
14561 0000459A FFA6[E8430000]      <1>       jmp    dword [esi+D1]
14562                              <1> ;;BAD_COMMAND_POP:
14563                              <1> ;;    POP    AX
14564                              <1> ;;    POP    BX
14565                              <1> BAD_COMMAND:
14566 000045A0 C605[634E0100]01    <1>         MOV     byte [DISK_STATUS1],BAD_CMD  ; COMMAND ERROR
14567 000045A7 B000                <1>       MOV    AL,0
14568 000045A9 C3                  <1>       RETn
14569                              <1>
14570                              <1> ;---------------------------------------
14571                              <1> ;     RESET THE DISK SYSTEM  (AH=00H) :
14572                              <1> ;---------------------------------------
14573                              <1>
14574                              <1> ; 18-1-2015 : one controller reset (not other one)
14575                              <1>
14576                              <1> DISK_RESET:
14577 000045AA FA                  <1>       CLI
14578 000045AB E4A1                <1>       IN     AL,INTB01           ; GET THE MASK REGISTER
14579                              <1>       ;JMP   $+2
14580                              <1>       IODELAY
14581 000045AD EB00                <2>  jmp short $+2
14582 000045AF EB00                <2>  jmp short $+2
14583                              <1>       ;AND   AL,0BFH             ; ENABLE FIXED DISK INTERRUPT
```

```
14584 000045B1 243F                <1>       and    al,3Fh              ; 22/12/2014 (IRQ 14 & IRQ 15)
14585 000045B3 E6A1                <1>       OUT    INTB01,AL
14586 000045B5 FB                  <1>       STI                        ; START INTERRUPTS
14587                              <1>       ; 14/02/2015
14588 000045B6 6689D7              <1>       mov    di, dx
14589                              <1>       ; 04/01/2015
14590                              <1>       ;xor   di,di
14591                              <1> drst0:
14592 000045B9 B004                <1>       MOV    AL,04H  ; bit 2 - SRST
14593                              <1>       ;MOV   DX,HF_REG_PORT
14594 000045BB 668B15[EA5C0000]    <1>       MOV    DX,[HF_REG_PORT]
14595 000045C2 EE                  <1>       OUT    DX,AL               ; RESET
14596                              <1> ;     MOV    CX,10               ; DELAY COUNT
14597                              <1> ;DRD: DEC    CX
14598                              <1> ;     JNZ    short DRD           ; WAIT 4.8 MICRO-SEC
14599                              <1>       ;mov   cx,2                ; wait for 30 micro seconds
14600 000045C3 B902000000          <1>         mov  ecx, 2 ; 21/02/2015
14601 000045C8 E820D8FFFF          <1>       call    WAITF                   ; (Award Bios 1999 - WAIT_REFRESH,
14602                              <1>                                         ; 40 micro seconds)
14603 000045CD A0[654E0100]        <1>       mov    al,[CONTROL_BYTE]
14604 000045D2 240F                <1>       AND    AL,0FH              ; SET HEAD OPTION
14605 000045D4 EE                  <1>       OUT    DX,AL               ; TURN RESET OFF
14606 000045D5 E838040000          <1>       CALL   NOT_BUSY
14607 000045DA 7515                <1>       JNZ    short DRERR         ; TIME OUT ON RESET
14608 000045DC 668B15[E85C0000]    <1>       MOV    DX,[HF_PORT]
14609 000045E3 FEC2                <1>       inc    dl  ; HF_PORT+1
14610                              <1>       ; 02/01/2015 - Award BIOS 1999 - AHDSK.ASM
14611                              <1>         ;mov     cl, 10
14612 000045E5 B90A000000          <1>         mov     ecx, 10 ; 21/02/2015
14613                              <1> drst1:
14614 000045EA EC                  <1>       IN     AL,DX               ; GET RESET STATUS
14615 000045EB 3C01                <1>       CMP    AL,1
14616                              <1>       ; 04/01/2015
14617 000045ED 740A                <1>       jz     short drst2
14618                              <1>       ;JNZ   short DRERR         ; BAD RESET STATUS
14619                              <1>              ; Drive/Head Register - bit 4
14620 000045EF E2F9                <1>       loop   drst1
14621                              <1> DRERR:
14622 000045F1 C605[634E0100]05    <1>       MOV    byte [DISK_STATUS1],BAD_RESET ; CARD FAILED
14623 000045F8 C3                  <1>       RETn
14624                              <1> drst2:
14625                              <1>       ; 14/02/2015
14626 000045F9 6689FA              <1>       mov    dx,di
14627                              <1> ;drst3:
14628                              <1> ;     ; 05/01/2015
14629                              <1> ;     shl    di,1
14630                              <1> ;     ; 04/01/2015
14631                              <1> ;     mov    ax,[di+hd_cports]
14632                              <1> ;     cmp    ax,[HF_REG_PORT]
14633                              <1> ;     je     short drst4
14634                              <1> ;     mov    [HF_REG_PORT], ax
14635                              <1> ;     ; 03/01/2015
14636                              <1> ;     mov    ax,[di+hd_ports]
14637                              <1> ;       mov     [HF_PORT], ax
14638                              <1> ;     ; 05/01/2014
14639                              <1> ;     shr    di,1
14640                              <1> ;     ; 04/01/2015
14641                              <1> ;     jmp    short drst0  ; reset other controller
14642                              <1> ;drst4:
14643                              <1> ;     ; 05/01/2015
14644                              <1> ;     shr    di,1
14645                              <1> ;     mov    al,[di+hd_dregs]
14646                              <1> ;     and    al,10h ; bit 4 only
14647                              <1> ;     shr    al,4 ; bit 4  -> bit 0
14648                              <1> ;     mov    [hf_m_s], al ; (0 = master, 1 = slave)
14649                              <1>       ;
14650 000045FC A0[EC5C0000]        <1>       mov    al, [hf_m_s] ; 18/01/2015
14651 00004601 A801                <1>       test   al,1
14652                              <1> ;     jnz    short drst6
14653 00004603 7516                <1>         jnz     short drst4
14654 00004605 8065FDEF            <1>       AND     byte [CMD_BLOCK+5],0EFH ; SET TO DRIVE 0
14655                              <1> ;drst5:
14656                              <1> drst3:
14657 00004609 E835010000          <1>       CALL   INIT_DRV            ; SET MAX HEADS
14658                              <1>       ;mov   dx,di
14659 0000460E E8ED010000          <1>       CALL   HDISK_RECAL         ; RECAL TO RESET SEEK SPEED
14660                              <1>       ; 04/01/2014
14661                              <1> ;     inc    di
14662                              <1> ;     mov    dx,di
14663                              <1> ;     cmp    dl,[HF_NUM]
14664                              <1> ;     jb     short drst3
14665                              <1> ;DRE:
14666 00004613 C605[634E0100]00    <1>       MOV    byte [DISK_STATUS1],0     ; IGNORE ANY SET UP ERRORS
14667 0000461A C3                  <1>       RETn
14668                              <1> ;drst6:
14669                              <1> drst4:             ; Drive/Head Register - bit 4
14670 0000461B 804DFD10            <1>       OR      byte [CMD_BLOCK+5],010H ; SET TO DRIVE 1
14671                              <1>         ;jmp    short drst5
14672 0000461F EBE8                <1>         jmp     short drst3
14673                              <1>
14674                              <1> ;----------------------------------------
14675                              <1> ;     DISK STATUS ROUTINE  (AH = 01H) :
14676                              <1> ;----------------------------------------
14677                              <1>
14678                              <1> RETURN_STATUS:
14679 00004621 A0[634E0100]        <1>       MOV    AL,[DISK_STATUS1]   ; OBTAIN PREVIOUS STATUS
14680 00004626 C605[634E0100]00    <1>         MOV     byte [DISK_STATUS1],0   ; RESET STATUS
14681 0000462D C3                  <1>       RETn
14682                              <1>
14683                              <1> ;----------------------------------------
14684                              <1> ;     DISK READ ROUTINE    (AH = 02H) :
14685                              <1> ;----------------------------------------
```

```
14686                                  <1>
14687                                  <1> DISK_READ:
14688 0000462E C645FE20                <1>       MOV    byte [CMD_BLOCK+6],READ_CMD
14689 00004632 E954020000              <1>         JMP     COMMANDI
14690                                  <1>
14691                                  <1> ;---------------------------------------
14692                                  <1> ;     DISK WRITE ROUTINE   (AH = 03H) :
14693                                  <1> ;---------------------------------------
14694                                  <1>
14695                                  <1> DISK_WRITE:
14696 00004637 C645FE30                <1>       MOV    byte [CMD_BLOCK+6],WRITE_CMD
14697 0000463B E9A6020000              <1>         JMP     COMMANDO
14698                                  <1>
14699                                  <1> ;---------------------------------------
14700                                  <1> ;     DISK VERIFY       (AH = 04H) :
14701                                  <1> ;---------------------------------------
14702                                  <1>
14703                                  <1> DISK_VERF:
14704 00004640 C645FE40                <1>       MOV    byte [CMD_BLOCK+6],VERIFY_CMD
14705 00004644 E814030000              <1>       CALL   COMMAND
14706 00004649 750C                    <1>       JNZ    short VERF_EXIT           ; CONTROLLER STILL BUSY
14707 0000464B E886030000              <1>       CALL   _WAIT               ; (Original: CALL WAIT)
14708 00004650 7505                    <1>       JNZ    short VERF_EXIT           ; TIME OUT
14709 00004652 E813040000              <1>       CALL   CHECK_STATUS
14710                                  <1> VERF_EXIT:
14711 00004657 C3                      <1>       RETn
14712                                  <1>
14713                                  <1> ;---------------------------------------
14714                                  <1> ;     FORMATTING        (AH = 05H) :
14715                                  <1> ;---------------------------------------
14716                                  <1>
14717                                  <1> FMT_TRK:                          ; FORMAT TRACK      (AH = 005H)
14718 00004658 C645FE50                <1>       MOV    byte [CMD_BLOCK+6],FMTTRK_CMD
14719                                  <1>       ;PUSH  ES
14720                                  <1>       ;PUSH  BX
14721 0000465C 53                      <1>       push   ebx
14722 0000465D E8BA040000              <1>       CALL   GET_VEC            ; GET DISK PARAMETERS ADDRESS
14723                                  <1>       ;MOV   AL,[ES:BX+14]      ; GET SECTORS/TRACK
14724 00004662 8A430E                  <1>       mov    al, [ebx+14]
14725 00004665 8845F9                  <1>       MOV    [CMD_BLOCK+1],AL    ; SET SECTOR COUNT IN COMMAND
14726 00004668 5B                      <1>       pop    ebx
14727                                  <1>       ;POP   BX
14728                                  <1>       ;POP   ES
14729 00004669 E97F020000              <1>         JMP     CMD_OF                  ; GO EXECUTE THE COMMAND
14730                                  <1>
14731                                  <1> ;---------------------------------------
14732                                  <1> ;     READ DASD TYPE          (AH = 15H) :
14733                                  <1> ;---------------------------------------
14734                                  <1>
14735                                  <1> READ_DASD_TYPE:
14736                                  <1> READ_D_T:                         ; GET DRIVE PARAMETERS
14737 0000466E 1E                      <1>       PUSH   DS                 ; SAVE REGISTERS
14738                                  <1>       ;PUSH  ES
14739 0000466F 53                      <1>       PUSH   eBX
14740                                  <1>       ;CALL  DDS                ; ESTABLISH ADDRESSING
14741                                  <1>       ;push  cs
14742                                  <1>       ;pop   ds
14743 00004670 66BB1000                <1>         mov  bx, KDATA
14744 00004674 8EDB                    <1>       mov    ds, bx
14745                                  <1>       ;mov   es, bx
14746 00004676 C605[634E0100]00        <1>       MOV     byte [DISK_STATUS1],0
14747 0000467D 8A1D[644E0100]          <1>       MOV    BL,[HF_NUM]        ; GET NUMBER OF DRIVES
14748 00004683 80E27F                  <1>       AND    DL,7FH             ; GET DRIVE NUMBER
14749 00004686 38D3                    <1>       CMP    BL,DL
14750 00004688 7627                    <1>       JBE    short RDT_NOT_PRESENT     ; RETURN DRIVE NOT PRESENT
14751 0000468A E88D040000              <1>       CALL   GET_VEC            ; GET DISK PARAMETER ADDRESS
14752                                  <1>       ;MOV   AL,[ES:BX+2]       ; HEADS
14753 0000468F 8A4302                  <1>       mov    al, [ebx+2]
14754                                  <1>       ;MOV   CL,[ES:BX+14]
14755 00004692 8A4B0E                  <1>       mov    cl, [ebx+14]
14756 00004695 F6E9                    <1>       IMUL   CL                 ; * NUMBER OF SECTORS
14757                                  <1>       ;MOV   CX,[ES:BX]         ; MAX NUMBER OF CYLINDERS
14758 00004697 668B0B                  <1>       mov    cx ,[ebx]
14759                                  <1>       ;
14760                                  <1>       ; 02/01/2015
14761                                  <1>       ; ** leave the last cylinder as reserved for diagnostics **
14762                                  <1>       ; (Also in Award BIOS - 1999, AHDSK.ASM, FUN15 -> sub ax, 1)
14763 0000469A 6649                    <1>       DEC    CX                 ; LEAVE ONE FOR DIAGNOSTICS
14764                                  <1>       ;
14765 0000469C 66F7E9                  <1>       IMUL   CX                 ; NUMBER OF SECTORS
14766 0000469F 6689D1                  <1>       MOV    CX,DX              ; HIGH ORDER HALF
14767 000046A2 6689C2                  <1>       MOV    DX,AX              ; LOW ORDER HALF
14768                                  <1>       ;SUB   AX,AX
14769 000046A5 28C0                    <1>       sub    al, al
14770 000046A7 B403                    <1>       MOV    AH,03H             ; INDICATE FIXED DISK
14771 000046A9 5B                      <1> RDT2: POP    eBX                ; RESTORE REGISTERS
14772                                  <1>       ;POP   ES
14773 000046AA 1F                      <1>       POP    DS
14774                                  <1>       ; (*) CLC                 ; CLEAR CARRY
14775                                  <1>       ;RETf  2
14776                                  <1>       ; (*) 29/05/2016
14777                                  <1>       ; (*) retf 4
14778 000046AB 80642408FE              <1>       and    byte [esp+8], 0FEh ; clear carry bit of eflags register
14779 000046B0 CF                      <1>       iretd
14780                                  <1>
14781                                  <1> RDT_NOT_PRESENT:
14782 000046B1 6629C0                  <1>       SUB    AX,AX              ; DRIVE NOT PRESENT RETURN
14783 000046B4 6689C1                  <1>       MOV    CX,AX              ; ZERO BLOCK COUNT
14784 000046B7 6689C2                  <1>       MOV    DX,AX
14785 000046BA EBED                    <1>       JMP    short RDT2
14786                                  <1>
14787                                  <1> ; 28/05/2016
```

```
14788                                  <1> ; 27/05/2016 - TRDOS 386 (TRDOS v2.0)
14789                                  <1>
14790                                  <1> ;----------------------------------------
14791                                  <1> ;     GET PARAMETERS          (AH = 08H) :
14792                                  <1> ;----------------------------------------
14793                                  <1>
14794                                  <1> GET_PARM_N:
14795                                  <1>       ; ebx = user's buffer address for parameters table
14796                                  <1> ;GET_PARM:                      ; GET DRIVE PARAMETERS
14797 000046BC 1E                      <1>       PUSH   DS                 ; SAVE REGISTERS
14798 000046BD 06                      <1>       PUSH   ES
14799 000046BE 53                      <1>       PUSH   eBX
14800                                  <1>       ;MOV   AX,ABS0            ; ESTABLISH ADDRESSING
14801                                  <1>       ;MOV   DS,AX
14802                                  <1>       ;TEST  DL,1               ; CHECK FOR DRIVE 1
14803                                  <1>       ;JZ    short G0
14804                                  <1>       ;LES   BX,@HF1_TBL_VEC
14805                                  <1>       ;JMP   SHORT G1
14806                                  <1> ;G0:  LES    BX,@HF_TBL_VEC
14807                                  <1> ;G1:
14808                                  <1>       ;CALL  DDS                ; ESTABLISH SEGMENT
14809                                  <1>       ; 22/12/2014
14810                                  <1>       ;push  cs
14811                                  <1>       ;pop   ds
14812 000046BF 66BB1000               <1>       mov    bx, KDATA
14813 000046C3 8EDB                    <1>       mov    ds, bx
14814 000046C5 8EC3                    <1>       mov    es, bx ; 27/05/2016
14815                                  <1>       ;
14816 000046C7 80EA80                  <1>       SUB    DL,80H
14817 000046CA 80FA04                  <1>       CMP    DL,MAX_FILE        ; TEST WITHIN RANGE
14818 000046CD 7361                    <1>       JAE    short G4
14819                                  <1>       ;
14820 000046CF 31DB                    <1>       xor    ebx, ebx ; 21/02/2015
14821                                  <1>       ; 22/12/2014
14822 000046D1 88D3                    <1>       mov    bl, dl
14823                                  <1>       ;xor   bh, bh
14824 000046D3 C0E302                  <1>       shl    bl, 2              ; convert index to offset
14825                                  <1>       ;add   bx, HF_TBL_VEC
14826 000046D6 81C3[684E0100]          <1>       add    ebx, HF_TBL_VEC
14827                                  <1>       ;mov   ax, [bx+2]
14828                                  <1>       ;mov   es, ax             ; dpt segment
14829                                  <1>       ;mov   bx, [bx]           ; dpt offset
14830 000046DC 8B1B                    <1>       mov    ebx, [ebx] ; 32 bit offset
14831                                  <1>
14832 000046DE C605[634E0100]00        <1>       MOV    byte [DISK_STATUS1],0
14833                                  <1>         ;MOV     AX,[ES:BX]             ; MAX NUMBER OF CYLINDERS
14834 000046E5 668B03                  <1>       mov    ax, [ebx]
14835                                  <1>       ;;SUB  AX,2               ; ADJUST FOR 0-N
14836 000046E8 6648                    <1>       dec    ax                 ; max. cylinder number
14837 000046EA 88C5                    <1>       MOV    CH,AL
14838 000046EC 66250003                <1>       AND    AX,0300H           ; HIGH TWO BITS OF CYLINDER
14839 000046F0 66D1E8                  <1>       SHR    AX,1
14840 000046F3 66D1E8                  <1>       SHR    AX,1
14841                                  <1>       ;OR    AL,[ES:BX+14]      ; SECTORS
14842 000046F6 0A430E                  <1>       or     al, [ebx+14]
14843 000046F9 88C1                    <1>       MOV    CL,AL
14844                                  <1>       ;MOV   DH,[ES:BX+2]       ; HEADS
14845 000046FB 8A7302                  <1>       mov    dh, [ebx+2]
14846 000046FE FECE                    <1>       DEC    DH                 ; 0-N RANGE
14847 00004700 8A15[644E0100]          <1>       MOV    DL,[HF_NUM]        ; DRIVE COUNT
14848 00004706 6629C0                  <1>       SUB    AX,AX
14849                                  <1>         ;27/12/2014
14850                                  <1>       ;mov   di, bx             ; HDPT offset
14851                                  <1>
14852                                  <1>       ; 27/05/2016
14853                                  <1>       ; return fixed disk parameters table to user
14854                                  <1>       ; in user's buffer, which is pointed by EBX
14855                                  <1>       ;
14856 00004709 873C24                  <1>       xchg   edi, [esp]         ; ebx (input)-> edi, edi -> [esp]
14857 0000470C 56                      <1>       push   esi
14858 0000470D 89DE                    <1>       mov    esi, ebx           ; hard disk parameter table (32 bytes)
14859 0000470F 89FB                    <1>       mov    ebx, edi           ; ebx = user's buffer address
14860 00004711 51                      <1>       push   ecx
14861 00004712 50                      <1>       push   eax
14862 00004713 B920000000              <1>       mov    ecx, 32 ; 32 bytes
14863 00004718 E865A10000              <1>       call   transfer_to_user_buffer ; trdosk6.s (16/05/2016)
14864 0000471D 58                      <1>       pop    eax
14865 0000471E 59                      <1>       pop    ecx
14866 0000471F 5E                      <1>       pop    esi
14867 00004720 5F                      <1>       pop    edi
14868 00004721 730A                    <1>       jnc    short G5
14869                                  <1>       ; 29/05/2016 (*)
14870 00004723 B8FF000000              <1>       mov    eax, 0FFh ; unknown error !
14871                                  <1> _G6:
14872 00004728 804C241001              <1>       or     byte [esp+16], 1 ; set carry bit of eflags register
14873                                  <1> G5:
14874                                  <1>       ; 27/05/2016
14875                                  <1>       ;POP   eBX                ; RESTORE REGISTERS
14876 0000472D 07                      <1>       POP    ES
14877 0000472E 1F                      <1>       POP    DS
14878                                  <1>       ;RETf  2
14879                                  <1>       ; (*) 29/05/2016
14880                                  <1>       ; (*) retf 4
14881                                  <1>       ; (*) or byte [esp+8], 1 ; set carry bit of eflags register
14882 0000472F CF                      <1>       iretd
14883                                  <1> G4:
14884 00004730 C605[634E0100]07        <1>       MOV     byte [DISK_STATUS1],INIT_FAIL ; OPERATION FAILED
14885 00004737 B407                    <1>       MOV    AH,INIT_FAIL
14886 00004739 28C0                    <1>       SUB    AL,AL
14887 0000473B 6629D2                  <1>       SUB    DX,DX
14888 0000473E 6629C9                  <1>       SUB    CX,CX
14889                                  <1>       ; 29/05/2016 (*)
```

```
14890                           <1>       ;STC                       ; SET ERROR FLAG
14891                           <1>       ;JMP   short G5
14892 00004741 EBE5             <1>       jmp    short _G6
14893                           <1>
14894                           <1> ;--------------------------------------
14895                           <1> ;     INITIALIZE DRIVE     (AH = 09H) :
14896                           <1> ;--------------------------------------
14897                           <1>       ; 03/01/2015
14898                           <1>       ; According to ATA-ATAPI specification v2.0 to v5.0
14899                           <1>       ; logical sector per logical track
14900                           <1>       ; and logical heads - 1 would be set but
14901                           <1>       ; it is seen as it will be good
14902                           <1>       ; if physical parameters will be set here
14903                           <1>       ; because, number of heads <= 16.
14904                           <1>       ; (logical heads usually more than 16)
14905                           <1>       ; NOTE: ATA logical parameters (software C, H, S)
14906                           <1>       ;     == INT 13h physical parameters
14907                           <1>
14908                           <1> ;INIT_DRV:
14909                           <1> ;     MOV    byte [CMD_BLOCK+6],SET_PARM_CMD
14910                           <1> ;     CALL   GET_VEC             ; ES:BX -> PARAMETER BLOCK
14911                           <1> ;     MOV    AL,[ES:BX+2]        ; GET NUMBER OF HEADS
14912                           <1> ;     DEC    AL                  ; CONVERT TO 0-INDEX
14913                           <1> ;     MOV    AH,[CMD_BLOCK+5]    ; GET SDH REGISTER
14914                           <1> ;     AND    AH,0F0H             ; CHANGE HEAD NUMBER
14915                           <1> ;     OR     AH,AL               ; TO MAX HEAD
14916                           <1> ;     MOV    [CMD_BLOCK+5],AH
14917                           <1> ;     MOV    AL,[ES:BX+14]       ; MAX SECTOR NUMBER
14918                           <1> ;     MOV    [CMD_BLOCK+1],AL
14919                           <1> ;     SUB    AX,AX
14920                           <1> ;     MOV    [CMD_BLOCK+3],AL    ; ZERO FLAGS
14921                           <1> ;     CALL   COMMAND             ; TELL CONTROLLER
14922                           <1> ;     JNZ    short INIT_EXIT           ; CONTROLLER BUSY ERROR
14923                           <1> ;     CALL   NOT_BUSY           ; WAIT FOR IT TO BE DONE
14924                           <1> ;     JNZ    short INIT_EXIT           ; TIME OUT
14925                           <1> ;     CALL   CHECK_STATUS
14926                           <1> ;INIT_EXIT:
14927                           <1> ;     RETn
14928                           <1>
14929                           <1> ; 04/01/2015
14930                           <1> ; 02/01/2015 - Derived from from AWARD BIOS 1999
14931                           <1> ;                          AHDSK.ASM - INIT_DRIVE
14932                           <1> INIT_DRV:
14933                           <1>       ;xor   ah,ah
14934 00004743 31C0             <1>       xor    eax, eax ; 21/02/2015
14935 00004745 B00B             <1>       mov    al,11 ; Physical heads from translated HDPT
14936 00004747 3825[784E0100]   <1>         cmp     [LBAMode], ah   ; 0
14937 0000474D 7702             <1>       ja     short idrv0
14938 0000474F B002             <1>       mov    al,2  ; Physical heads from standard HDPT
14939                           <1> idrv0:
14940                           <1>       ; DL = drive number (0 based)
14941 00004751 E8C6030000       <1>       call   GET_VEC
14942                           <1>       ;push  bx
14943 00004756 53               <1>       push   ebx ; 21/02/2015
14944                           <1>       ;add   bx,ax
14945 00004757 01C3             <1>       add    ebx, eax
14946                           <1>       ;; 05/01/2015
14947 00004759 8A25[EC5C0000]   <1>       mov    ah, [hf_m_s] ; drive number (0= master, 1= slave)
14948                           <1>       ;;and  ah,1
14949 0000475F C0E404           <1>       shl    ah,4
14950 00004762 80CCA0           <1>       or     ah,0A0h  ; Drive/Head register - 10100000b (A0h)
14951                           <1>       ;mov   al,[es:bx]
14952 00004765 8A03             <1>       mov    al, [ebx] ; 21/02/2015
14953 00004767 FEC8             <1>       dec    al      ; last head number
14954                           <1>       ;and   al,0Fh
14955 00004769 08E0             <1>       or     al,ah   ; lower 4 bits for head number
14956                           <1>       ;
14957 0000476B C645FE91         <1>       mov    byte [CMD_BLOCK+6],SET_PARM_CMD
14958 0000476F 8845FD           <1>       mov    [CMD_BLOCK+5],al
14959                           <1>       ;pop   bx
14960 00004772 5B               <1>       pop    ebx
14961 00004773 29C0             <1>       sub    eax, eax ; 21/02/2015
14962 00004775 B004             <1>       mov    al,4 ; Physical sec per track from translated HDPT
14963 00004777 803D[784E0100]00 <1>       cmp    byte [LBAMode], 0
14964 0000477E 7702             <1>       ja     short idrv1
14965 00004780 B00E             <1>       mov    al,14 ; Physical sec per track from standard HDPT
14966                           <1> idrv1:
14967                           <1>       ;xor   ah,ah
14968                           <1>       ;add   bx,ax
14969 00004782 01C3             <1>       add    ebx, eax ; 21/02/2015
14970                           <1>       ;mov   al,[es:bx]
14971                           <1>                    ; sector number
14972 00004784 8A03             <1>       mov    al, [ebx]
14973 00004786 8845F9           <1>       mov    [CMD_BLOCK+1],al
14974 00004789 28C0             <1>       sub    al,al
14975 0000478B 8845FB           <1>       mov    [CMD_BLOCK+3],al  ; ZERO FLAGS
14976 0000478E E8CA010000       <1>       call   COMMAND         ; TELL CONTROLLER
14977 00004793 750C             <1>       jnz    short INIT_EXIT       ; CONTROLLER BUSY ERROR
14978 00004795 E878020000       <1>       call   NOT_BUSY       ; WAIT FOR IT TO BE DONE
14979 0000479A 7505             <1>       jnz    short INIT_EXIT       ; TIME OUT
14980 0000479C E8C9020000       <1>       call   CHECK_STATUS
14981                           <1> INIT_EXIT:
14982 000047A1 C3               <1>       RETn
14983                           <1>
14984                           <1> ;--------------------------------------
14985                           <1> ;     READ LONG         (AH = 0AH) :
14986                           <1> ;--------------------------------------
14987                           <1>
14988                           <1> RD_LONG:
14989                           <1>       ;MOV   @CMD_BLOCK+6,READ_CMD OR ECC_MODE
14990 000047A2 C645FE22         <1>         mov     byte [CMD_BLOCK+6],READ_CMD + ECC_MODE
14991 000047A6 E9E0000000       <1>         JMP     COMMANDI
```

```
14992                                  <1>
14993                                  <1> ;----------------------------------------
14994                                  <1> ;     WRITE LONG        (AH = 0BH) :
14995                                  <1> ;----------------------------------------
14996                                  <1>
14997                                  <1> WR_LONG:
14998                                  <1>       ;MOV   @CMD_BLOCK+6,WRITE_CMD OR ECC_MODE
14999 000047AB C645FE32                <1>         MOV     byte [CMD_BLOCK+6],WRITE_CMD + ECC_MODE
15000 000047AF E932010000              <1>         JMP     COMMANDO
15001                                  <1>
15002                                  <1> ;----------------------------------------
15003                                  <1> ;     SEEK              (AH = 0CH) :
15004                                  <1> ;----------------------------------------
15005                                  <1>
15006                                  <1> DISK_SEEK:
15007 000047B4 C645FE70                <1>         MOV     byte [CMD_BLOCK+6],SEEK_CMD
15008 000047B8 E8A0010000              <1>       CALL   COMMAND
15009 000047BD 751C                    <1>       JNZ    short DS_EXIT             ; CONTROLLER BUSY ERROR
15010 000047BF E812020000              <1>       CALL   _WAIT
15011 000047C4 7515                    <1>         JNZ     DS_EXIT                  ; TIME OUT ON SEEK
15012 000047C6 E89F020000              <1>       CALL   CHECK_STATUS
15013 000047CB 803D[634E0100]40        <1>         CMP     byte [DISK_STATUS1],BAD_SEEK
15014 000047D2 7507                    <1>       JNE    short DS_EXIT
15015 000047D4 C605[634E0100]00        <1>         MOV     byte [DISK_STATUS1],0
15016                                  <1> DS_EXIT:
15017 000047DB C3                      <1>       RETn
15018                                  <1>
15019                                  <1> ;----------------------------------------
15020                                  <1> ;     TEST DISK READY      (AH = 10H) :
15021                                  <1> ;----------------------------------------
15022                                  <1>
15023                                  <1> TST_RDY:                          ; WAIT FOR CONTROLLER
15024 000047DC E831020000              <1>       CALL   NOT_BUSY
15025 000047E1 751C                    <1>       JNZ    short TR_EX
15026 000047E3 8A45FD                  <1>       MOV    AL,[CMD_BLOCK+5]    ; SELECT DRIVE
15027 000047E6 668B15[E85C0000]        <1>       MOV    DX,[HF_PORT]
15028 000047ED 80C206                  <1>       add    dl,6
15029 000047F0 EE                      <1>       OUT    DX,AL
15030 000047F1 E88C020000              <1>       CALL   CHECK_ST            ; CHECK STATUS ONLY
15031 000047F6 7507                    <1>       JNZ    short TR_EX
15032 000047F8 C605[634E0100]00        <1>       MOV    byte [DISK_STATUS1],0     ; WIPE OUT DATA CORRECTED ERROR
15033                                  <1> TR_EX:
15034 000047FF C3                      <1>       RETn
15035                                  <1>
15036                                  <1> ;----------------------------------------
15037                                  <1> ;     RECALIBRATE       (AH = 11H) :
15038                                  <1> ;----------------------------------------
15039                                  <1>
15040                                  <1> HDISK_RECAL:
15041 00004800 C645FE10                <1>         MOV     byte [CMD_BLOCK+6],RECAL_CMD ; 10h, 16
15042 00004804 E854010000              <1>       CALL   COMMAND             ; START THE OPERATION
15043 00004809 7523                    <1>       JNZ    short RECAL_EXIT    ; ERROR
15044 0000480B E8C6010000              <1>       CALL   _WAIT               ; WAIT FOR COMPLETION
15045 00004810 7407                    <1>       JZ     short RECAL_X             ; TIME OUT ONE OK ?
15046 00004812 E8BF010000              <1>       CALL   _WAIT               ; WAIT FOR COMPLETION LONGER
15047 00004817 7515                    <1>       JNZ    short RECAL_EXIT    ; TIME OUT TWO TIMES IS ERROR
15048                                  <1> RECAL_X:
15049 00004819 E84C020000              <1>       CALL   CHECK_STATUS
15050 0000481E 803D[634E0100]40        <1>       CMP    byte [DISK_STATUS1],BAD_SEEK ; SEEK NOT COMPLETE
15051 00004825 7507                    <1>       JNE    short RECAL_EXIT    ; IS OK
15052 00004827 C605[634E0100]00        <1>       MOV    byte [DISK_STATUS1],0
15053                                  <1> RECAL_EXIT:
15054 0000482E 803D[634E0100]00        <1>         CMP     byte [DISK_STATUS1],0
15055 00004835 C3                      <1>       RETn
15056                                  <1>
15057                                  <1> ;----------------------------------------
15058                                  <1> ;      CONTROLLER DIAGNOSTIC (AH = 14H) :
15059                                  <1> ;----------------------------------------
15060                                  <1>
15061                                  <1> CTLR_DIAGNOSTIC:
15062 00004836 FA                      <1>         CLI                           ; DISABLE INTERRUPTS WHILE CHANGING MASK
15063 00004837 E4A1                    <1>       IN     AL,INTB01          ; TURN ON SECOND INTERRUPT CHIP
15064                                  <1>       ;AND   AL,0BFH
15065 00004839 243F                    <1>       and    al, 3Fh                  ; enable IRQ 14 & IRQ 15
15066                                  <1>       ;JMP   $+2
15067                                  <1>       IODELAY
15068 0000483B EB00                    <2>  jmp short $+2
15069 0000483D EB00                    <2>  jmp short $+2
15070 0000483F E6A1                    <1>       OUT    INTB01,AL
15071                                  <1>       IODELAY
15072 00004841 EB00                    <2>  jmp short $+2
15073 00004843 EB00                    <2>  jmp short $+2
15074 00004845 E421                    <1>       IN     AL,INTA01          ; LET INTERRUPTS PASS THRU TO
15075 00004847 24FB                    <1>       AND    AL,0FBH            ;  SECOND CHIP
15076                                  <1>       ;JMP   $+2
15077                                  <1>       IODELAY
15078 00004849 EB00                    <2>  jmp short $+2
15079 0000484B EB00                    <2>  jmp short $+2
15080 0000484D E621                    <1>       OUT    INTA01,AL
15081 0000484F FB                      <1>       STI
15082 00004850 E8BD010000              <1>       CALL   NOT_BUSY           ; WAIT FOR CARD
15083 00004855 752B                    <1>       JNZ    short CD_ERR       ; BAD CARD
15084                                  <1>       ;MOV   DX, HF_PORT+7
15085 00004857 668B15[E85C0000]        <1>       mov    dx, [HF_PORT]
15086 0000485E 80C207                  <1>       add    dl, 7
15087 00004861 B090                    <1>       MOV    AL,DIAG_CMD        ; START DIAGNOSE
15088 00004863 EE                      <1>       OUT    DX,AL
15089 00004864 E8A9010000              <1>       CALL   NOT_BUSY           ; WAIT FOR IT TO COMPLETE
15090 00004869 B480                    <1>       MOV    AH,TIME_OUT
15091 0000486B 7517                    <1>       JNZ    short CD_EXIT             ; TIME OUT ON DIAGNOSTIC
15092                                  <1>       ;MOV   DX,HF_PORT+1       ; GET ERROR REGISTER
15093 0000486D 668B15[E85C0000]        <1>       mov    dx, [HF_PORT]
```

```
15094 00004874 FEC2                <1>       inc    dl
15095 00004876 EC                  <1>       IN     AL,DX
15096 00004877 A2[5A4E0100]        <1>       MOV    [HF_ERROR],AL       ; SAVE IT
15097 0000487C B400                <1>       MOV    AH,0
15098 0000487E 3C01                <1>       CMP    AL,1                ; CHECK FOR ALL OK
15099 00004880 7402                <1>       JE     SHORT CD_EXIT
15100 00004882 B420                <1> CD_ERR: MOV  AH,BAD_CNTLR
15101                              <1> CD_EXIT:
15102 00004884 8825[634E0100]      <1>       MOV    [DISK_STATUS1],AH
15103 0000488A C3                  <1>       RETn
15104                              <1>
15105                              <1> ;---------------------------------------
15106                              <1> ; COMMANDI                        :
15107                              <1> ;     REPEATEDLY INPUTS DATA TILL      :
15108                              <1> ;     NSECTOR RETURNS ZERO             :
15109                              <1> ;---------------------------------------
15110                              <1> COMMANDI:
15111 0000488B E862020000          <1>       CALL   CHECK_DMA           ; CHECK 64K BOUNDARY ERROR
15112 00004890 7253                <1>       JC     short CMD_ABORT
15113                              <1>       ;MOV   DI,BX
15114 00004892 89DF                <1>       mov    edi, ebx ; 21/02/2015
15115 00004894 E8C4000000          <1>       CALL   COMMAND             ; OUTPUT COMMAND
15116 00004899 754A                <1>       JNZ    short CMD_ABORT
15117                              <1> CMD_I1:
15118 0000489B E836010000          <1>       CALL   _WAIT               ; WAIT FOR DATA REQUEST INTERRUPT
15119 000048A0 7543                <1>       JNZ    short TM_OUT        ; TIME OUT
15120                              <1> cmd_i1x: ; 18/02/2016
15121                              <1>       ;MOV   CX,256              ; SECTOR SIZE IN WORDS
15122 000048A2 B900010000          <1>       mov    ecx, 256 ; 21/02/2015
15123                              <1>       ;MOV   DX,HF_PORT
15124 000048A7 668B15[E85C0000]    <1>       mov    dx,[HF_PORT]
15125 000048AE FA                  <1>       CLI
15126 000048AF FC                  <1>       CLD
15127 000048B0 F3666D              <1>       REP    INSW                ; GET THE SECTOR
15128 000048B3 FB                  <1>       STI
15129 000048B4 F645FE02            <1>       TEST   byte [CMD_BLOCK+6],ECC_MODE ; CHECK FOR NORMAL INPUT
15130 000048B8 7419                <1>       JZ     short CMD_I3
15131 000048BA E880010000          <1>       CALL   WAIT_DRQ            ; WAIT FOR DATA REQUEST
15132 000048BF 7224                <1>       JC     short TM_OUT
15133                              <1>       ;MOV   DX,HF_PORT
15134 000048C1 668B15[E85C0000]    <1>       mov    dx,[HF_PORT]
15135                              <1>       ;MOV   CX,4                ; GET ECC BYTES
15136 000048C8 B904000000          <1>       mov    ecx, 4 ; mov cx, 4
15137 000048CD EC                  <1> CMD_I2: IN   AL,DX
15138                              <1>       ;MOV   [ES:DI],AL          ; GO SLOW FOR BOARD
15139 000048CE 8807                <1>       mov    [edi], al ; 21/02/2015
15140 000048D0 47                  <1>       INC    eDI
15141 000048D1 E2FA                <1>       LOOP   CMD_I2
15142                              <1> CMD_I3:
15143                              <1>       ; wait for 400 ns
15144 000048D3 80C207              <1>       add    dl, 7
15145 000048D6 EC                  <1>       in     al, dx
15146 000048D7 EC                  <1>       in     al, dx
15147 000048D8 EC                  <1>       in     al, dx
15148                              <1>       ;
15149 000048D9 E88C010000          <1>       CALL   CHECK_STATUS
15150 000048DE 7505                <1>       JNZ    short CMD_ABORT           ; ERROR RETURNED
15151 000048E0 FE4DF9              <1>       DEC    byte [CMD_BLOCK+1]  ; CHECK FOR MORE
15152                              <1>       ;JNZ   SHORT CMD_I1
15153 000048E3 75BD                <1>       jnz    short cmd_i1x ; 18/02/2016
15154                              <1> CMD_ABORT:
15155 000048E5 C3                  <1> TM_OUT: RETn
15156                              <1>
15157                              <1> ;---------------------------------------
15158                              <1> ; COMMANDO                        :
15159                              <1> ;     REPEATEDLY OUTPUTS DATA TILL     :
15160                              <1> ;     NSECTOR RETURNS ZERO             :
15161                              <1> ;---------------------------------------
15162                              <1> COMMANDO:
15163 000048E6 E807020000          <1>       CALL   CHECK_DMA           ; CHECK 64K BOUNDARY ERROR
15164 000048EB 72F8                <1>       JC     short CMD_ABORT
15165 000048ED 89DE                <1> CMD_OF: MOV  eSI,eBX ; 21/02/2015
15166 000048EF E869000000          <1>       CALL   COMMAND             ; OUTPUT COMMAND
15167 000048F4 75EF                <1>       JNZ    short CMD_ABORT
15168 000048F6 E844010000          <1>       CALL   WAIT_DRQ            ; WAIT FOR DATA REQUEST
15169 000048FB 72E8                <1>       JC     short TM_OUT              ; TOO LONG
15170                              <1> CMD_O1: ;PUSH      DS
15171                              <1>       ;PUSH  ES                  ; MOVE ES TO DS
15172                              <1>       ;POP   DS
15173                              <1>       ;MOV   CX,256              ; PUT THE DATA OUT TO THE CARD
15174                              <1>       ;MOV   DX,HF_PORT
15175                              <1>       ; 01/02/2015
15176 000048FD 668B15[E85C0000]    <1>       mov    dx, [HF_PORT]
15177                              <1>       ;push  es
15178                              <1>       ;pop   ds
15179                              <1>       ;mov   cx, 256
15180 00004904 B900010000          <1>       mov    ecx, 256 ; 21/02/2015
15181 00004909 FA                  <1>       CLI
15182 0000490A FC                  <1>       CLD
15183 0000490B F3666F              <1>       REP    OUTSW
15184 0000490E FB                  <1>       STI
15185                              <1>       ;POP   DS                  ; RESTORE DS
15186 0000490F F645FE02            <1>       TEST   byte [CMD_BLOCK+6],ECC_MODE ; CHECK FOR NORMAL OUTPUT
15187 00004913 7419                <1>       JZ     short CMD_O3
15188 00004915 E825010000          <1>       CALL   WAIT_DRQ            ; WAIT FOR DATA REQUEST
15189 0000491A 72C9                <1>       JC     short TM_OUT
15190                              <1>       ;MOV   DX,HF_PORT
15191 0000491C 668B15[E85C0000]    <1>       mov    dx, [HF_PORT]
15192                              <1>       ;MOV   CX,4                ; OUTPUT THE ECC BYTES
15193 00004923 B904000000          <1>       mov    ecx, 4  ; mov cx, 4
15194                              <1> CMD_O2: ;MOV AL,[ES:SI]
15195 00004928 8A06                <1>       mov    al, [esi]
```

```
15196 0000492A EE                  <1>       OUT    DX,AL
15197 0000492B 46                  <1>       INC    eSI
15198 0000492C E2FA                <1>       LOOP   CMD_O2
15199                              <1> CMD_O3:
15200 0000492E E8A3000000          <1>       CALL   _WAIT               ; WAIT FOR SECTOR COMPLETE INTERRUPT
15201 00004933 75B0                <1>       JNZ    short TM_OUT        ; ERROR RETURNED
15202 00004935 E830010000          <1>       CALL   CHECK_STATUS
15203 0000493A 75A9                <1>       JNZ    short CMD_ABORT
15204 0000493C F605[594E0100]08    <1>       TEST   byte [HF_STATUS],ST_DRQ   ; CHECK FOR MORE
15205 00004943 75B8                <1>       JNZ    SHORT CMD_O1
15206                              <1>       ;MOV   DX,HF_PORT+2        ; CHECK RESIDUAL SECTOR COUNT
15207 00004945 668B15[E85C0000]    <1>       mov    dx, [HF_PORT]
15208                              <1>       ;add   dl, 2
15209 0000494C FEC2                <1>       inc    dl
15210 0000494E FEC2                <1>       inc    dl
15211 00004950 EC                  <1>       IN     AL,DX               ;
15212 00004951 A8FF                <1>       TEST   AL,0FFH             ;
15213 00004953 7407                <1>       JZ     short CMD_O4              ; COUNT = 0  OK
15214 00004955 C605[634E0100]BB    <1>       MOV    byte [DISK_STATUS1],UNDEF_ERR
15215                              <1>                                  ; OPERATION ABORTED - PARTIAL TRANSFER
15216                              <1> CMD_O4:
15217 0000495C C3                  <1>       RETn
15218                              <1>
15219                              <1> ;--------------------------------------------------------
15220                              <1> ; COMMAND                                          :
15221                              <1> ;     THIS ROUTINE OUTPUTS THE COMMAND BLOCK        :
15222                              <1> ; OUTPUT                                           :
15223                              <1> ;     BL = STATUS                                  :
15224                              <1> ;     BH = ERROR REGISTER                          :
15225                              <1> ;--------------------------------------------------------
15226                              <1>
15227                              <1> COMMAND:
15228 0000495D 53                  <1>       PUSH   eBX                 ; WAIT FOR SEEK COMPLETE AND READY
15229                              <1>       ;;MOV  CX,DELAY_2          ; SET INITIAL DELAY BEFORE TEST
15230                              <1> COMMAND1:
15231                              <1>       ;;PUSH CX                  ; SAVE LOOP COUNT
15232 0000495E E879FEFFFF          <1>       CALL   TST_RDY             ; CHECK DRIVE READY
15233                              <1>       ;;POP  CX
15234 00004963 7419                <1>       JZ     short COMMAND2             ; DRIVE IS READY
15235 00004965 803D[634E0100]80    <1>         CMP     byte [DISK_STATUS1],TIME_OUT ; TST_RDY TIMED OUT--GIVE UP
15236                              <1>       ;JZ    short CMD_TIMEOUT
15237                              <1>       ;;LOOP COMMAND1            ; KEEP TRYING FOR A WHILE
15238                              <1>       ;JMP   SHORT COMMAND4             ; ITS NOT GOING TO GET READY
15239 0000496C 7507                <1>       jne    short COMMAND4
15240                              <1> CMD_TIMEOUT:
15241 0000496E C605[634E0100]20    <1>       MOV    byte [DISK_STATUS1],BAD_CNTLR
15242                              <1> COMMAND4:
15243 00004975 5B                  <1>       POP    eBX
15244 00004976 803D[634E0100]00    <1>         CMP     byte [DISK_STATUS1],0   ; SET CONDITION CODE FOR CALLER
15245 0000497D C3                  <1>       RETn
15246                              <1> COMMAND2:
15247 0000497E 5B                  <1>       POP    eBX
15248 0000497F 57                  <1>       PUSH   eDI
15249 00004980 C605[5B4E0100]00    <1>       MOV    byte [HF_INT_FLAG],0       ; RESET INTERRUPT FLAG
15250 00004987 FA                  <1>       CLI                        ; INHIBIT INTERRUPTS WHILE CHANGING MASK
15251 00004988 E4A1                <1>       IN     AL,INTB01           ; TURN ON SECOND INTERRUPT CHIP
15252                              <1>       ;AND   AL,0BFH
15253 0000498A 243F                <1>       and    al, 3Fh                    ; Enable IRQ 14 & 15
15254                              <1>       ;JMP   $+2
15255                              <1>       IODELAY
15256 0000498C EB00                <2>  jmp short $+2
15257 0000498E EB00                <2>  jmp short $+2
15258 00004990 E6A1                <1>       OUT    INTB01,AL
15259 00004992 E421                <1>       IN     AL,INTA01           ; LET INTERRUPTS PASS THRU TO
15260 00004994 24FB                <1>       AND    AL,0FBH             ;  SECOND CHIP
15261                              <1>       ;JMP   $+2
15262                              <1>       IODELAY
15263 00004996 EB00                <2>  jmp short $+2
15264 00004998 EB00                <2>  jmp short $+2
15265 0000499A E621                <1>       OUT    INTA01,AL
15266 0000499C FB                  <1>       STI
15267 0000499D 31FF                <1>       XOR    eDI,eDI                    ; INDEX THE COMMAND TABLE
15268                              <1>       ;MOV   DX,HF_PORT+1        ; DISK ADDRESS
15269 0000499F 668B15[E85C0000]    <1>       mov    dx, [HF_PORT]
15270 000049A6 FEC2                <1>       inc    dl
15271 000049A8 F605[654E0100]C0    <1>       TEST   byte [CONTROL_BYTE],0C0H ; CHECK FOR RETRY SUPPRESSION
15272 000049AF 7411                <1>       JZ     short COMMAND3
15273 000049B1 8A45FE              <1>       MOV    AL, [CMD_BLOCK+6]   ; YES-GET OPERATION CODE
15274 000049B4 24F0                <1>       AND    AL,0F0H             ; GET RID OF MODIFIERS
15275 000049B6 3C20                <1>       CMP    AL,20H              ; 20H-40H IS READ, WRITE, VERIFY
15276 000049B8 7208                <1>       JB     short COMMAND3
15277 000049BA 3C40                <1>       CMP    AL,40H
15278 000049BC 7704                <1>       JA     short COMMAND3
15279 000049BE 804DFE01            <1>       OR     byte [CMD_BLOCK+6],NO_RETRIES
15280                              <1>                                   ; VALID OPERATION FOR RETRY SUPPRESS
15281                              <1> COMMAND3:
15282 000049C2 8A443DF8            <1>       MOV    AL,[CMD_BLOCK+eDI]  ; GET THE COMMAND STRING BYTE
15283 000049C6 EE                  <1>       OUT    DX,AL               ; GIVE IT TO CONTROLLER
15284                              <1>       IODELAY
15285 000049C7 EB00                <2>  jmp short $+2
15286 000049C9 EB00                <2>  jmp short $+2
15287 000049CB 47                  <1>       INC    eDI                 ; NEXT BYTE IN COMMAND BLOCK
15288 000049CC 6642                <1>       INC    DX                  ; NEXT DISK ADAPTER REGISTER
15289 000049CE 6683FF07            <1>       cmp    di, 7  ; 1/1/2015   ; ALL DONE?
15290 000049D2 75EE                <1>       JNZ    short COMMAND3            ; NO--GO DO NEXT ONE
15291 000049D4 5F                  <1>       POP    eDI
15292 000049D5 C3                  <1>       RETn                       ; ZERO FLAG IS SET
15293                              <1>
15294                              <1> ;CMD_TIMEOUT:
15295                              <1> ;     MOV    byte [DISK_STATUS1],BAD_CNTLR
15296                              <1> ;COMMAND4:
15297                              <1> ;     POP    BX
```

```
15298                                  <1> ;     CMP    [DISK_STATUS1],0    ; SET CONDITION CODE FOR CALLER
15299                                  <1> ;     RETn
15300                                  <1>
15301                                  <1> ;---------------------------------------
15302                                  <1> ;     WAIT FOR INTERRUPT        :
15303                                  <1> ;---------------------------------------
15304                                  <1> ;WAIT:
15305                                  <1> _WAIT:
15306 000049D6 FB                      <1>       STI                       ; MAKE SURE INTERRUPTS ARE ON
15307                                  <1>       ;SUB   CX,CX              ; SET INITIAL DELAY BEFORE TEST
15308                                  <1>       ;CLC
15309                                  <1>       ;MOV   AX,9000H           ; DEVICE WAIT INTERRUPT
15310                                  <1>       ;INT   15H
15311                                  <1>       ;JC    WT2                ; DEVICE TIMED OUT
15312                                  <1>       ;MOV   BL,DELAY_1         ; SET DELAY COUNT
15313                                  <1>
15314                                  <1>       ;mov   bl, WAIT_HDU_INT_HI
15315                                  <1>       ;; 21/02/2015
15316                                  <1>       ;;mov  bl, WAIT_HDU_INT_HI + 1
15317                                  <1>       ;;mov  cx, WAIT_HDU_INT_LO
15318 000049D7 B915160500              <1>       mov    ecx, WAIT_HDU_INT_LH
15319                                  <1>                                 ; (AWARD BIOS -> WAIT_FOR_MEM)
15320                                  <1> ;-----      WAIT LOOP
15321                                  <1>
15322                                  <1> WT1:
15323                                  <1>       ;TEST  byte [HF_INT_FLAG],80H    ; TEST FOR INTERRUPT
15324 000049DC F605[5B4E0100]C0        <1>       test   byte [HF_INT_FLAG],0C0h
15325                                  <1>       ;LOOPZ WT1
15326 000049E3 7517                    <1>       JNZ    short WT3          ; INTERRUPT--LETS GO
15327                                  <1>       ;DEC   BL
15328                                  <1>       ;JNZ   short WT1          ; KEEP TRYING FOR A WHILE
15329                                  <1>
15330                                  <1> WT1_hi:
15331 000049E5 E461                    <1>       in     al, SYS1 ; 61h (PORT_B)   ; wait for lo to hi
15332 000049E7 A810                    <1>       test   al, 10h                   ; transition on memory
15333 000049E9 75FA                    <1>       jnz    short WT1_hi       ; refresh.
15334                                  <1> WT1_lo:
15335 000049EB E461                    <1>       in     al, SYS1           ; 061h (PORT_B)
15336 000049ED A810                    <1>       test   al, 10h
15337 000049EF 74FA                    <1>       jz     short WT1_lo
15338 000049F1 E2E9                    <1>       loop   WT1
15339                                  <1>       ;;or   bl, bl
15340                                  <1>       ;;jz   short WT2
15341                                  <1>       ;;dec  bl
15342                                  <1>       ;;jmp  short WT1
15343                                  <1>       ;dec   bl
15344                                  <1>       ;jnz   short WT1
15345                                  <1>
15346 000049F3 C605[634E0100]80        <1> WT2:  MOV    byte [DISK_STATUS1],TIME_OUT ; REPORT TIME OUT ERROR
15347 000049FA EB0E                    <1>       JMP    SHORT WT4
15348 000049FC C605[634E0100]00        <1> WT3:  MOV    byte [DISK_STATUS1],0
15349 00004A03 C605[5B4E0100]00        <1>       MOV    byte [HF_INT_FLAG],0
15350 00004A0A 803D[634E0100]00        <1> WT4:  CMP    byte [DISK_STATUS1],0     ; SET CONDITION CODE FOR CALLER
15351 00004A11 C3                      <1>       RETn
15352                                  <1>
15353                                  <1> ;---------------------------------------
15354                                  <1> ;     WAIT FOR CONTROLLER NOT BUSY     :
15355                                  <1> ;---------------------------------------
15356                                  <1> NOT_BUSY:
15357 00004A12 FB                      <1>       STI                       ; MAKE SURE INTERRUPTS ARE ON
15358                                  <1>       ;PUSH  eBX
15359                                  <1>       ;SUB   CX,CX              ; SET INITIAL DELAY BEFORE TEST
15360 00004A13 668B15[E85C0000]        <1>       mov    DX, [HF_PORT]
15361 00004A1A 80C207                  <1>       add    dl, 7              ; Status port (HF_PORT+7)
15362                                  <1>       ;MOV   BL,DELAY_1
15363                                  <1>                                 ; wait for 10 seconds
15364                                  <1>       ;mov   cx, WAIT_HDU_INT_LO ; 1615h
15365                                  <1>       ;;mov  bl, WAIT_HDU_INT_HI ;   05h
15366                                  <1>       ;mov   bl, WAIT_HDU_INT_HI + 1
15367 00004A1D B915160500              <1>       mov    ecx, WAIT_HDU_INT_LH  ; 21/02/2015
15368                                  <1>       ;
15369                                  <1> ;;      mov     byte [wait_count], 0    ; Reset wait counter
15370                                  <1> NB1:
15371 00004A22 EC                      <1>       IN     AL,DX              ; CHECK STATUS
15372                                  <1>       ;TEST  AL,ST_BUSY
15373 00004A23 2480                    <1>       and    al, ST_BUSY
15374                                  <1>       ;LOOPNZ      NB1
15375 00004A25 7410                    <1>       JZ     short NB2          ; NOT BUSY--LETS GO
15376                                  <1>       ;DEC   BL
15377                                  <1>       ;JNZ   short NB1          ; KEEP TRYING FOR A WHILE
15378                                  <1>
15379 00004A27 E461                    <1> NB1_hi: IN   AL,SYS1                  ; wait for hi to lo
15380 00004A29 A810                    <1>       TEST   AL,010H                  ; transition on memory
15381 00004A2B 75FA                    <1>       JNZ    SHORT NB1_hi       ; refresh.
15382 00004A2D E461                    <1> NB1_lo: IN   AL,SYS1
15383 00004A2F A810                    <1>       TEST   AL,010H
15384 00004A31 74FA                    <1>       JZ     short NB1_lo
15385 00004A33 E2ED                    <1>       LOOP   NB1
15386                                  <1>       ;dec   bl
15387                                  <1>       ;jnz   short NB1
15388                                  <1>       ;
15389                                  <1> ;;      cmp     byte [wait_count], 182  ; 10 seconds (182 timer ticks)
15390                                  <1> ;;    jb     short NB1
15391                                  <1>       ;
15392                                  <1>       ;MOV   [DISK_STATUS1],TIME_OUT   ; REPORT TIME OUT ERROR
15393                                  <1>       ;JMP   SHORT NB3
15394 00004A35 B080                    <1>       mov    al, TIME_OUT
15395                                  <1> NB2:
15396                                  <1>       ;MOV   byte [DISK_STATUS1],0
15397                                  <1> ;NB3:
15398                                  <1>       ;POP   eBX
15399 00004A37 A2[634E0100]            <1>       mov    [DISK_STATUS1], al  ;;; will be set after return
```

```
15400                                  <1>       ;CMP   byte [DISK_STATUS1],0     ; SET CONDITION CODE FOR CALLER
15401 00004A3C 08C0                    <1>       or     al, al              ; (zf = 0 --> timeout)
15402 00004A3E C3                      <1>       RETn
15403                                  <1>
15404                                  <1> ;---------------------------------------
15405                                  <1> ;     WAIT FOR DATA REQUEST             :
15406                                  <1> ;---------------------------------------
15407                                  <1> WAIT_DRQ:
15408                                  <1>       ;MOV   CX,DELAY_3
15409                                  <1>       ;MOV   DX,HF_PORT+7
15410 00004A3F 668B15[E85C0000]        <1>       mov    dx, [HF_PORT]
15411 00004A46 80C207                  <1>       add    dl, 7
15412                                  <1>       ;;MOV  bl, WAIT_HDU_DRQ_HI ; 0
15413                                  <1>       ;MOV   cx, WAIT_HDU_DRQ_LO ; 1000 (30 milli seconds)
15414                                  <1>                                  ; (but it is written as 2000
15415                                  <1>                                  ; micro seconds in ATORGS.ASM file
15416                                  <1>                                  ; of Award Bios - 1999, D1A0622)
15417 00004A49 B9E8030000              <1>       mov    ecx, WAIT_HDU_DRQ_LH ; 21/02/2015
15418 00004A4E EC                      <1> WQ_1: IN     AL,DX               ; GET STATUS
15419 00004A4F A808                    <1>       TEST   AL,ST_DRQ           ; WAIT FOR DRQ
15420 00004A51 7516                    <1>       JNZ    short WQ_OK
15421                                  <1>       ;LOOP  WQ_1                ; KEEP TRYING FOR A SHORT WHILE
15422                                  <1> WQ_hi:
15423 00004A53 E461                    <1>       IN     AL,SYS1                   ; wait for hi to lo
15424 00004A55 A810                    <1>       TEST   AL,010H                   ; transition on memory
15425 00004A57 75FA                    <1>       JNZ    SHORT WQ_hi         ; refresh.
15426 00004A59 E461                    <1> WQ_lo:  IN      AL,SYS1
15427 00004A5B A810                    <1>       TEST   AL,010H
15428 00004A5D 74FA                    <1>       JZ     SHORT WQ_lo
15429 00004A5F E2ED                    <1>       LOOP   WQ_1
15430                                  <1>
15431 00004A61 C605[634E0100]80        <1>         MOV     byte [DISK_STATUS1],TIME_OUT ; ERROR
15432 00004A68 F9                      <1>       STC
15433                                  <1> WQ_OK:
15434 00004A69 C3                      <1>       RETn
15435                                  <1> ;WQ_OK:      ;CLC
15436                                  <1> ;     RETn
15437                                  <1>
15438                                  <1> ;---------------------------------------
15439                                  <1> ;     CHECK FIXED DISK STATUS   :
15440                                  <1> ;---------------------------------------
15441                                  <1> CHECK_STATUS:
15442 00004A6A E813000000              <1>       CALL   CHECK_ST            ; CHECK THE STATUS BYTE
15443 00004A6F 7509                    <1>       JNZ    short CHECK_S1             ; AN ERROR WAS FOUND
15444 00004A71 A801                    <1>       TEST   AL,ST_ERROR         ; WERE THERE ANY OTHER ERRORS
15445 00004A73 7405                    <1>       JZ     short CHECK_S1             ; NO ERROR REPORTED
15446 00004A75 E849000000              <1>       CALL   CHECK_ER            ; ERROR REPORTED
15447                                  <1> CHECK_S1:
15448 00004A7A 803D[634E0100]00        <1>       CMP    byte [DISK_STATUS1],0     ; SET STATUS FOR CALLER
15449 00004A81 C3                      <1>       RETn
15450                                  <1>
15451                                  <1> ;---------------------------------------
15452                                  <1> ;     CHECK FIXED DISK STATUS BYTE     :
15453                                  <1> ;---------------------------------------
15454                                  <1> CHECK_ST:
15455                                  <1>       ;MOV   DX,HF_PORT+7        ; GET THE STATUS
15456 00004A82 668B15[E85C0000]        <1>       mov    dx, [HF_PORT]
15457 00004A89 80C207                  <1>       add    dl, 7
15458                                  <1>
15459                                  <1>       ; 17/02/2016
15460                                  <1>       ;(http://wiki.osdev.org/ATA_PIO_Mode)
15461                                  <1>       ;"delay 400ns to allow drive to set new values of BSY and DRQ"
15462 00004A8C EC                      <1>       IN     AL,DX
15463                                  <1>       ;in    al, dx ; 100ns
15464                                  <1>       ;in    al, dx ; 100ns
15465                                  <1>       ;in    al, dx ; 100ns
15466                                  <1>       NEWIODELAY ; 18/02/2016 (AWARD BIOS - 1999, 'CKST' in AHSDK.ASM)
15467 00004A8D E6EB                    <2>  out 0ebh,al
15468                                  <1>       ;
15469 00004A8F A2[594E0100]            <1>       MOV    [HF_STATUS],AL
15470 00004A94 B400                    <1>       MOV    AH,0
15471 00004A96 A880                    <1>       TEST   AL,ST_BUSY          ; IF STILL BUSY
15472 00004A98 751A                    <1>       JNZ    short CKST_EXIT           ;  REPORT OK
15473 00004A9A B4CC                    <1>       MOV    AH,WRITE_FAULT
15474 00004A9C A820                    <1>       TEST   AL,ST_WRT_FLT       ; CHECK FOR WRITE FAULT
15475 00004A9E 7514                    <1>       JNZ    short CKST_EXIT
15476 00004AA0 B4AA                    <1>       MOV    AH,NOT_RDY
15477 00004AA2 A840                    <1>       TEST   AL,ST_READY         ; CHECK FOR NOT READY
15478 00004AA4 740E                    <1>       JZ     short CKST_EXIT
15479 00004AA6 B440                    <1>       MOV    AH,BAD_SEEK
15480 00004AA8 A810                    <1>       TEST   AL,ST_SEEK_COMPL    ; CHECK FOR SEEK NOT COMPLETE
15481 00004AAA 7408                    <1>       JZ     short CKST_EXIT
15482 00004AAC B411                    <1>       MOV    AH,DATA_CORRECTED
15483 00004AAE A804                    <1>       TEST   AL,ST_CORRCTD       ; CHECK FOR CORRECTED ECC
15484 00004AB0 7502                    <1>       JNZ    short CKST_EXIT
15485 00004AB2 B400                    <1>       MOV    AH,0
15486                                  <1> CKST_EXIT:
15487 00004AB4 8825[634E0100]          <1>       MOV    [DISK_STATUS1],AH   ; SET ERROR FLAG
15488 00004ABA 80FC11                  <1>       CMP    AH,DATA_CORRECTED   ; KEEP GOING WITH DATA CORRECTED
15489 00004ABD 7403                    <1>       JZ     short CKST_EX1
15490 00004ABF 80FC00                  <1>       CMP    AH,0
15491                                  <1> CKST_EX1:
15492 00004AC2 C3                      <1>       RETn
15493                                  <1>
15494                                  <1> ;---------------------------------------
15495                                  <1> ;     CHECK FIXED DISK ERROR REGISTER :
15496                                  <1> ;---------------------------------------
15497                                  <1> CHECK_ER:
15498                                  <1>       ;MOV   DX, HF_PORT+1       ; GET THE ERROR REGISTER
15499 00004AC3 668B15[E85C0000]        <1>       mov    dx, [HF_PORT]       ;
15500 00004ACA FEC2                    <1>       inc    dl
15501 00004ACC EC                      <1>       IN     AL,DX
```

```
15502 00004ACD A2[5A4E0100]        <1>       MOV   [HF_ERROR],AL
15503 00004AD2 53                  <1>       PUSH  eBX ; 21/02/2015
15504 00004AD3 B908000000          <1>       MOV   eCX,8               ; TEST ALL 8 BITS
15505 00004AD8 D0E0                <1> CK1:  SHL   AL,1                ; MOVE NEXT ERROR BIT TO CARRY
15506 00004ADA 7202                <1>       JC    short CK2           ; FOUND THE ERROR
15507 00004ADC E2FA                <1>       LOOP  CK1                 ; KEEP TRYING
15508 00004ADE BB[DC5C0000]        <1> CK2:  MOV   eBX, ERR_TBL        ; COMPUTE ADDRESS OF
15509 00004AE3 01CB                <1>       ADD   eBX,eCX                   ; ERROR CODE
15510                              <1>       ;;MOV AH,BYTE [CS:BX]           ; GET ERROR CODE
15511                              <1>       ;mov  ah, [bx]
15512 00004AE5 8A23                <1>       mov   ah, [ebx] ; 21/02/2015
15513 00004AE7 8825[634E0100]      <1> CKEX: MOV   [DISK_STATUS1],AH   ; SAVE ERROR CODE
15514 00004AED 5B                  <1>       POP   eBX
15515 00004AEE 80FC00              <1>       CMP   AH,0
15516 00004AF1 C3                  <1>       RETn
15517                              <1>
15518                              <1> ;--------------------------------------------------
15519                              <1> ; CHECK_DMA                                  :
15520                              <1> ;  -CHECK ES:BX AND # SECTORS TO MAKE SURE THAT IT WILL :
15521                              <1> ;   FIT WITHOUT SEGMENT OVERFLOW.                  :
15522                              <1> ;  -ES:BX HAS BEEN REVISED TO THE FORMAT SSSS:000X  :
15523                              <1> ;  -OK IF # SECTORS < 80H (7FH IF LONG READ OR WRITE)     :
15524                              <1> ;  -OK IF # SECTORS = 80H (7FH) AND BX <= 00H (04H) :
15525                              <1> ;  -ERROR OTHERWISE                                :
15526                              <1> ;--------------------------------------------------
15527                              <1> CHECK_DMA:
15528 00004AF2 6650                <1>       PUSH  AX                  ; SAVE REGISTERS
15529 00004AF4 66B80080            <1>       MOV   AX,8000H            ; AH = MAX # SECTORS AL = MAX OFFSET
15530 00004AF8 F645FE02            <1>         TEST    byte [CMD_BLOCK+6],ECC_MODE
15531 00004AFC 7404                <1>       JZ    short CKD1
15532 00004AFE 66B8047F            <1>       MOV   AX,7F04H            ; ECC IS 4 MORE BYTES
15533 00004B02 3A65F9              <1> CKD1: CMP   AH, [CMD_BLOCK+1]   ; NUMBER OF SECTORS
15534 00004B05 7706                <1>       JA    short CKDOK         ; IT WILL FIT
15535 00004B07 7208                <1>       JB    short CKDERR        ; TOO MANY
15536 00004B09 38D8                <1>       CMP   AL,BL               ; CHECK OFFSET ON MAX SECTORS
15537 00004B0B 7204                <1>       JB    short CKDERR        ; ERROR
15538 00004B0D F8                  <1> CKDOK:      CLC                       ; CLEAR CARRY
15539 00004B0E 6658                <1>       POP   AX
15540 00004B10 C3                  <1>       RETn                      ; NORMAL RETURN
15541 00004B11 F9                  <1> CKDERR: STC                     ; INDICATE ERROR
15542 00004B12 C605[634E0100]09    <1>         MOV     byte [DISK_STATUS1],DMA_BOUNDARY
15543 00004B19 6658                <1>       POP   AX
15544 00004B1B C3                  <1>       RETn
15545                              <1>
15546                              <1> ;----------------------------------------
15547                              <1> ;     SET UP ES:BX-> DISK PARMS :
15548                              <1> ;----------------------------------------
15549                              <1>
15550                              <1> ; INPUT -> DL = 0 based drive number
15551                              <1> ; OUTPUT -> ES:BX = disk parameter table address
15552                              <1>
15553                              <1> GET_VEC:
15554                              <1>       ;SUB  AX,AX               ; GET DISK PARAMETER ADDRESS
15555                              <1>       ;MOV  ES,AX
15556                              <1>       ;TEST DL,1
15557                              <1>       ;JZ   short GV_0
15558                              <1> ;     LES   BX,[HF1_TBL_VEC]    ; ES:BX -> DRIVE PARAMETERS
15559                              <1> ;     JMP   SHORT GV_EXIT
15560                              <1> ;GV_0:
15561                              <1> ;     LES   BX,[HF_TBL_VEC]           ; ES:BX -> DRIVE PARAMETERS
15562                              <1> ;
15563                              <1>       ;xor  bh, bh
15564 00004B1C 31DB                <1>       xor   ebx, ebx
15565 00004B1E 88D3                <1>       mov   bl, dl
15566                              <1>       ;;02/01/2015
15567                              <1>       ;;shl bl, 1               ; port address offset
15568                              <1>       ;;mov ax, [bx+hd_ports]   ; Base port address (1F0h, 170h)
15569                              <1>       ;;shl bl, 1               ; dpt pointer offset
15570 00004B20 C0E302              <1>       shl   bl, 2 ;;
15571                              <1>       ;add  bx, HF_TBL_VEC            ; Disk parameter table pointer
15572 00004B23 81C3[684E0100]      <1>       add   ebx, HF_TBL_VEC ; 21/02/2015
15573                              <1>       ;push word [bx+2]         ; dpt segment
15574                              <1>       ;pop  es
15575                              <1>       ;mov  bx, [bx]            ; dpt offset
15576 00004B29 8B1B                <1>       mov   ebx, [ebx]
15577                              <1> ;GV_EXIT:
15578 00004B2B C3                  <1>       RETn
15579                              <1>
15580                              <1> hdc1_int: ; 21/02/2015
15581                              <1> ;--- HARDWARE INT 76H -- ( IRQ LEVEL  14 ) ----------------------
15582                              <1> ;                                                 :
15583                              <1> ;     FIXED DISK INTERRUPT ROUTINE                      :
15584                              <1> ;                                                 :
15585                              <1> ;-------------------------------------------------------------
15586                              <1>
15587                              <1> ; 22/12/2014
15588                              <1> ; IBM PC-XT Model 286 System BIOS Source Code - DISK.ASM (HD_INT)
15589                              <1> ;      '11/15/85'
15590                              <1> ; AWARD BIOS 1999 (D1A0622)
15591                              <1> ;     Source Code - ATORGS.ASM (INT_HDISK, INT_HDISK1)
15592                              <1>
15593                              <1> ;int_76h:
15594                              <1> HD_INT:
15595 00004B2C 6650                <1>       PUSH  AX
15596 00004B2E 1E                  <1>       PUSH  DS
15597                              <1>       ;CALL DDS
15598                              <1>       ; 21/02/2015 (32 bit, 386 pm modification)
15599 00004B2F 66B81000            <1>       mov   ax, KDATA
15600 00004B33 8ED8                <1>       mov   ds, ax
15601                              <1>       ;
15602                              <1>       ;;MOV @HF_INT_FLAG,0FFH   ; ALL DONE
15603                              <1>         ;mov     byte [CS:HF_INT_FLAG], 0FFh
```

```
15604 00004B35 C605[5B4E0100]FF    <1>       mov    byte [HF_INT_FLAG], 0FFh
15605                              <1>       ;
15606 00004B3C 6652                <1>       push   dx
15607 00004B3E 66BAF701            <1>       mov    dx, HDC1_BASEPORT+7 ; Status Register (1F7h)
15608                              <1>                                  ; Clear Controller
15609                              <1> Clear_IRQ1415:                        ; (Award BIOS - 1999)
15610 00004B42 EC                  <1>       in     al, dx              ;
15611 00004B43 665A                <1>       pop    dx
15612                              <1>       NEWIODELAY
15613 00004B45 E6EB                <2>  out 0ebh,al
15614                              <1>       ;
15615 00004B47 B020                <1>       MOV    AL,EOI              ; NON-SPECIFIC END OF INTERRUPT
15616 00004B49 E6A0                <1>       OUT    INTB00,AL           ; FOR CONTROLLER #2
15617                              <1>       ;JMP   $+2                 ; WAIT
15618                              <1>       NEWIODELAY
15619 00004B4B E6EB                <2>  out 0ebh,al
15620 00004B4D E620                <1>       OUT    INTA00,AL           ; FOR CONTROLLER #1
15621 00004B4F 1F                  <1>       POP    DS
15622                              <1>       ;STI                       ; RE-ENABLE INTERRUPTS
15623                              <1>       ;MOV   AX,9100H            ; DEVICE POST
15624                              <1>       ;INT   15H                 ;  INTERRUPT
15625                              <1> irq15_iret: ; 25/02/2015
15626 00004B50 6658                <1>       POP    AX
15627 00004B52 CF                  <1>       IRETd                      ; RETURN FROM INTERRUPT
15628                              <1>
15629                              <1> hdc2_int: ; 21/02/2015
15630                              <1> ;++++ HARDWARE INT 77H ++ ( IRQ LEVEL  15 ) +++++++++++++++++++++
15631                              <1> ;                                                      :
15632                              <1> ;     FIXED DISK INTERRUPT ROUTINE                           :
15633                              <1> ;                                                      :
15634                              <1> ;++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
15635                              <1>
15636                              <1> ;int_77h:
15637                              <1> HD1_INT:
15638 00004B53 6650                <1>       PUSH   AX
15639                              <1>       ; Check if that is a spurious IRQ (from slave PIC)
15640                              <1>       ; 25/02/2015 (source: http://wiki.osdev.org/8259_PIC)
15641 00004B55 B00B                <1>       mov    al, 0Bh  ; In-Service Register
15642 00004B57 E6A0                <1>       out    0A0h, al
15643 00004B59 EB00                <1>         jmp short $+2
15644 00004B5B EB00                <1>       jmp short $+2
15645 00004B5D E4A0                <1>       in     al, 0A0h
15646 00004B5F 2480                <1>       and    al, 80h ; bit 7 (is it real IRQ 15 or fake?)
15647 00004B61 74ED                <1>       jz     short irq15_iret ; Fake (spurious)IRQ, do not send EOI)
15648                              <1>       ;
15649 00004B63 1E                  <1>       PUSH   DS
15650                              <1>       ;CALL  DDS
15651                              <1>       ; 21/02/2015 (32 bit, 386 pm modification)
15652 00004B64 66B81000            <1>       mov    ax, KDATA
15653 00004B68 8ED8                <1>       mov    ds, ax
15654                              <1>       ;
15655                              <1>       ;;MOV  @HF_INT_FLAG,0FFH   ; ALL DONE
15656                              <1>         ;or      byte [CS:HF_INT_FLAG],0C0h
15657 00004B6A 800D[5B4E0100]C0    <1>       or     byte [HF_INT_FLAG], 0C0h
15658                              <1>       ;
15659 00004B71 6652                <1>       push   dx
15660 00004B73 66BA7701            <1>       mov    dx, HDC2_BASEPORT+7 ; Status Register (177h)
15661                              <1>                                  ; Clear Controller (Award BIOS 1999)
15662 00004B77 EBC9                <1>       jmp    short Clear_IRQ1415
15663                              <1>
15664                              <1>
15665                              <1> ;%include 'diskdata.inc' ; 11/03/2015
15666                              <1> ;%include 'diskbss.inc' ; 11/03/2015
15667                              <1>
15668                              <1>
15669                              <1> ;////////////////////////////////////////////////////////////////
15670                              <1> ;; END OF DISK I/O SYTEM ///
15671                                  %include 'memory.s'  ; 09/03/2015
15672                              <1> ; ============================================================================
15673                              <1> ; TRDOS386.ASM (TRDOS 386 Kernel) - v2.0.0 - memory.s
15674                              <1> ; ----------------------------------------------------------------------------
15675                              <1> ; Last Update: 22/07/2017
15676                              <1> ; ----------------------------------------------------------------------------
15677                              <1> ; Beginning: 24/01/2016
15678                              <1> ; ----------------------------------------------------------------------------
15679                              <1> ; Assembler: NASM version 2.11 (trdos386.s)
15680                              <1> ; ----------------------------------------------------------------------------
15681                              <1> ; Turkish Rational DOS
15682                              <1> ; Operating System Project v2.0 by ERDOGAN TAN (Beginning: 04/01/2016)
15683                              <1> ;
15684                              <1> ; Derived from 'Retro UNIX 386 Kernel - v0.2.1.0' source code by Erdogan Tan
15685                              <1> ; memory.inc (18/10/2015)
15686                              <1> ; ============================================================================
15687                              <1>
15688                              <1> ; MEMORY.ASM - Retro UNIX 386 v1 MEMORY MANAGEMENT FUNCTIONS (PROCEDURES)
15689                              <1> ; Retro UNIX 386 v1 Kernel (unix386.s, v0.2.0.14) - MEMORY.INC
15690                              <1> ; Last Modification: 18/10/2015
15691                              <1>
15692                              <1> ; ///////// MEMORY MANAGEMENT FUNCTIONS (PROCEDURES) ///////////////
15693                              <1>
15694                              <1> ;;04/11/2014 (unix386.s)
15695                              <1> ;PDE_A_PRESENT     equ    1            ; Present flag for PDE
15696                              <1> ;PDE_A_WRITE equ    2            ; Writable (write permission) flag
15697                              <1> ;PDE_A_USER  equ    4            ; User (non-system/kernel) page flag
15698                              <1> ;;
15699                              <1> ;PTE_A_PRESENT     equ    1            ; Present flag for PTE (bit 0)
15700                              <1> ;PTE_A_WRITE equ    2            ; Writable (write permission) flag (bit 1)
15701                              <1> ;PTE_A_USER  equ    4            ; User (non-system/kernel) page flag (bit 2)
15702                              <1> ;PTE_A_ACCESS   equ      32           ; Accessed flag (bit 5) ; 09/03/2015
15703                              <1>
15704                              <1> ; 27/04/2015
15705                              <1> ; 09/03/2015
```

```
15706                                  <1> PAGE_SIZE    equ 4096           ; page size in bytes
15707                                  <1> PAGE_SHIFT   equ 12             ; page table shift count
15708                                  <1> PAGE_D_SHIFT       equ 22 ; 12 + 10    ; page directory shift count
15709                                  <1> PAGE_OFF     equ 0FFFh          ; 12 bit byte offset in page frame
15710                                  <1> PTE_MASK     equ 03FFh          ; page table entry mask
15711                                  <1> PTE_DUPLICATED  equ 200h        ; duplicated page sign (AVL bit 0)
15712                                  <1> PDE_A_CLEAR  equ 0F000h         ; to clear PDE attribute bits
15713                                  <1> PTE_A_CLEAR  equ 0F000h         ; to clear PTE attribute bits
15714                                  <1> LOGIC_SECT_SIZE equ 512                ; logical sector size
15715                                  <1> ERR_MAJOR_PF equ 0E0h           ; major error: page fault
15716                                  <1> ERR_MINOR_IM equ 4 ;15/10/2016 (1->4); insufficient (out of) memory
15717                                  <1> ERR_MINOR_PV equ 6 ;15/10/2016 (1->4); protection violation
15718                                  <1> SWP_DISK_READ_ERR     equ 40
15719                                  <1> SWP_DISK_NOT_PRESENT_ERR   equ 41
15720                                  <1> SWP_SECTOR_NOT_PRESENT_ERR equ 42
15721                                  <1> SWP_NO_FREE_SPACE_ERR      equ 43
15722                                  <1> SWP_DISK_WRITE_ERR         equ 44
15723                                  <1> SWP_NO_PAGE_TO_SWAP_ERR    equ 45
15724                                  <1> PTE_A_ACCESS_BIT equ 5  ; Bit 5 (accessed flag)
15725                                  <1> SECTOR_SHIFT     equ 3  ; sector shift (to convert page block number)
15726                                  <1> ; 12/07/2016
15727                                  <1> PTE_SHARED   equ 400h           ; AVL bit 1, direct memory access bit
15728                                  <1>                                 ; (Indicates that the page is not allocated
15729                                  <1>                                 ; for the process, it is a shared or system
15730                                  <1>                                        ; page, it must not be deallocated!)
15731                                  <1> ;
15732                                  <1> ;; Retro Unix 386 v1 - paging method/principles
15733                                  <1> ;;
15734                                  <1> ;; 10/10/2014
15735                                  <1> ;; RETRO UNIX 386 v1 - PAGING METHOD/PRINCIPLES
15736                                  <1> ;;
15737                                  <1> ;; KERNEL PAGE MAP: 1 to 1 physical memory page map
15738                                  <1> ;;    (virtual address = physical address)
15739                                  <1> ;; KERNEL PAGE TABLES:
15740                                  <1> ;;    Kernel page directory and all page tables are
15741                                  <1> ;;    on memory as initialized, as equal to physical memory
15742                                  <1> ;;    layout. Kernel pages can/must not be swapped out/in.
15743                                  <1> ;;
15744                                  <1> ;;    what for: User pages may be swapped out, when accessing
15745                                  <1> ;;    a page in kernel/system mode, if it would be swapped out,
15746                                  <1> ;;    kernel would have to swap it in! But it is also may be
15747                                  <1> ;;    in use by a user process. (In system/kernel mode
15748                                  <1> ;;    kernel can access all memory pages even if they are
15749                                  <1> ;;    reserved/allocated for user processes. Swap out/in would
15750                                  <1> ;;    cause conflicts.)
15751                                  <1> ;;
15752                                  <1> ;;    As result of these conditions,
15753                                  <1> ;;    all kernel pages must be initialized as equal to
15754                                  <1> ;;    physical layout for preventing page faults.
15755                                  <1> ;;    Also, calling "allocate page" procedure after
15756                                  <1> ;;    a page fault can cause another page fault (double fault)
15757                                  <1> ;;    if all kernel page tables would not be initialized.
15758                                  <1> ;;
15759                                  <1> ;;    [first_page] = Beginning of users space, as offset to
15760                                  <1> ;;    memory allocation table. (double word aligned)
15761                                  <1> ;;
15762                                  <1> ;;    [next_page] = first/next free space to be searched
15763                                  <1> ;;    as offset to memory allocation table. (dw aligned)
15764                                  <1> ;;
15765                                  <1> ;;    [last_page] = End of memory (users space), as offset
15766                                  <1> ;;    to memory allocation table. (double word aligned)
15767                                  <1> ;;
15768                                  <1> ;; USER PAGE TABLES:
15769                                  <1> ;;    Demand paging (& 'copy on write' allocation method) ...
15770                                  <1> ;;           'ready only' marked copies of the
15771                                  <1> ;;           parent process's page table entries (for
15772                                  <1> ;;           same physical memory).
15773                                  <1> ;;           (A page will be copied to a new page after
15774                                  <1> ;;            if it causes R/W page fault.)
15775                                  <1> ;;
15776                                  <1> ;;    Every user process has own (different)
15777                                  <1> ;;    page directory and page tables.
15778                                  <1> ;;
15779                                  <1> ;;    Code starts at virtual address 0, always.
15780                                  <1> ;;    (Initial value of EIP is 0 in user mode.)
15781                                  <1> ;;    (Programs can be written/developed as simple
15782                                  <1> ;;     flat memory programs.)
15783                                  <1> ;;
15784                                  <1> ;; MEMORY ALLOCATION STRATEGY:
15785                                  <1> ;;    Memory page will be allocated by kernel only
15786                                  <1> ;;          (in kernel/system mode only).
15787                                  <1> ;;    * After a
15788                                  <1> ;;      - 'not present' page fault
15789                                  <1> ;;      - 'writing attempt on read only page' page fault
15790                                  <1> ;;    * For loading (opening, reading) a file or disk/drive
15791                                  <1> ;;    * As responce to 'allocate additional memory blocks'
15792                                  <1> ;;      request by running process.
15793                                  <1> ;;    * While creating a process, allocating a new buffer,
15794                                  <1> ;;      new page tables etc.
15795                                  <1> ;;
15796                                  <1> ;;    At first,
15797                                  <1> ;;    - 'allocate page' procedure will be called;
15798                                  <1> ;,       if it will return with a valid (>0) physical address
15799                                  <1> ;;       (that means the relevant M.A.T. bit has been RESET)
15800                                  <1> ;;       relevant memory page/block will be cleared (zeroed).
15801                                  <1> ;;    - 'allocate page' will be called for allocating page
15802                                  <1> ;;       directory, page table and running space (data/code).
15803                                  <1> ;;    - every successful 'allocate page' call will decrease
15804                                  <1> ;;      'free_pages' count (pointer).
15805                                  <1> ;;    - 'out of (insufficient) memory error' will be returned
15806                                  <1> ;;      if 'free_pages' points to a ZERO.
15807                                  <1> ;;    - swapping out and swapping in (if it is not a new page)
```

```
15808                           <1> ;;      procedures will be called as responce to 'out of memory'
15809                           <1> ;;      error except errors caused by attribute conflicts.
15810                           <1> ;;     (swapper functions)
15811                           <1> ;;
15812                           <1> ;;    At second,
15813                           <1> ;;    - page directory entry will be updated then page table
15814                           <1> ;;      entry will be updated.
15815                           <1> ;;
15816                           <1> ;; MEMORY ALLOCATION TABLE FORMAT:
15817                           <1> ;;    - M.A.T. has a size according to available memory as
15818                           <1> ;;      follows:
15819                           <1> ;;             - 1 (allocation) bit per 1 page (4096 bytes)
15820                           <1> ;;             - a bit with value of 0 means allocated page
15821                           <1> ;;             - a bit with value of 1 means a free page
15822                           <1> ;,    - 'free_pages' pointer holds count of free pages
15823                           <1> ;;      depending on M.A.T.
15824                           <1> ;;          (NOTE: Free page count will not be checked
15825                           <1> ;;          again -on M.A.T.- after initialization.
15826                           <1> ;;          Kernel will trust on initial count.)
15827                           <1> ;,    - 'free_pages' count will be decreased by allocation
15828                           <1> ;;      and it will be increased by deallocation procedures.
15829                           <1> ;;
15830                           <1> ;;    - Available memory will be calculated during
15831                           <1> ;;      the kernel's initialization stage (in real mode).
15832                           <1> ;;      Memory allocation table and kernel page tables
15833                           <1> ;;      will be formatted/sized as result of available
15834                           <1> ;;      memory calculation before paging is enabled.
15835                           <1> ;;
15836                           <1> ;; For 4GB Available/Present Memory: (max. possible memory size)
15837                           <1> ;;    - Memory Allocation Table size will be 128 KB.
15838                           <1> ;;    - Memory allocation for kernel page directory size
15839                           <1> ;;      is always 4 KB. (in addition to total allocation size
15840                           <1> ;;      for page tables)
15841                           <1> ;;    - Memory allocation for kernel page tables (1024 tables)
15842                           <1> ;;      is 4 MB (1024*4*1024 bytes).
15843                           <1> ;;    - User (available) space will be started
15844                           <1> ;;      at 6th MB of the memory (after 1MB+4MB).
15845                           <1> ;;    - The first 640 KB is for kernel's itself plus
15846                           <1> ;;      memory allocation table and kernel's page directory
15847                           <1> ;;      (D0000h-EFFFFh may be used as kernel space...)
15848                           <1> ;;    - B0000h to B7FFFh address space (32 KB) will be used
15849                           <1> ;;      for buffers.
15850                           <1> ;;    - ROMBIOS, VIDEO BUFFER and VIDEO ROM space are reserved.
15851                           <1> ;,      (A0000h-AFFFFh, C0000h-CFFFFh, F0000h-FFFFFh)
15852                           <1> ;;    - Kernel page tables start at 100000h (2nd MB)
15853                           <1> ;;
15854                           <1> ;; For 1GB Available Memory:
15855                           <1> ;;    - Memory Allocation Table size will be 32 KB.
15856                           <1> ;;    - Memory allocation for kernel page directory size
15857                           <1> ;;      is always 4 KB. (in addition to total allocation size
15858                           <1> ;;      for page tables)
15859                           <1> ;;    - Memory allocation for kernel page tables (256 tables)
15860                           <1> ;;      is 1 MB (256*4*1024 bytes).
15861                           <1> ;;    - User (available) space will be started
15862                           <1> ;;      at 3th MB of the memory (after 1MB+1MB).
15863                           <1> ;;    - The first 640 KB is for kernel's itself plus
15864                           <1> ;;      memory allocation table and kernel's page directory
15865                           <1> ;;      (D0000h-EFFFFh may be used as kernel space...)
15866                           <1> ;;    - B0000h to B7FFFh address space (32 KB) will be used
15867                           <1> ;;      for buffers.
15868                           <1> ;;    - ROMBIOS, VIDEO BUFFER and VIDEO ROM space are reserved.
15869                           <1> ;,      (A0000h-AFFFFh, C0000h-CFFFFh, F0000h-FFFFFh)
15870                           <1> ;;    - Kernel page tables start at 100000h (2nd MB).
15871                           <1> ;;
15872                           <1> ;;
15873                           <1>
15874                           <1>
15875                           <1> ;;*****************************************************************************
15876                           <1> ;;
15877                           <1> ;; RETRO UNIX 386 v1 - Paging (Method for Copy On Write paging principle)
15878                           <1> ;; DEMAND PAGING - PARENT&CHILD PAGE TABLE DUPLICATION PRINCIPLES (23/04/2015)
15879                           <1>
15880                           <1> ;; Main factor: "sys fork" system call
15881                           <1> ;;
15882                           <1> ;;          FORK
15883                           <1> ;;                        |----> parent - duplicated PTEs, read only pages
15884                           <1> ;;  writable pages ---->|
15885                           <1> ;;                        |----> child - duplicated PTEs, read only pages
15886                           <1> ;;
15887                           <1> ;; AVL bit (0) of Page Table Entry is used as duplication sign
15888                           <1> ;;
15889                           <1> ;; AVL Bit 0 [PTE Bit 9] = 'Duplicated PTE belongs to child' sign/flag (if it is set)
15890                           <1> ;; Note: Dirty bit (PTE bit 6) may be used instead of AVL bit 0 (PTE bit 9)
15891                           <1> ;;       -while R/W bit is 0-.
15892                           <1> ;;
15893                           <1> ;; Duplicate page tables with writable pages (the 1st sys fork in the process):
15894                           <1> ;; # Parent's Page Table Entries are updated to point same pages as read only,
15895                           <1> ;;   as duplicated PTE bit  -AVL bit 0, PTE bit 9- are reset/clear.
15896                           <1> ;; # Then Parent's Page Table is copied to Child's Page Table.
15897                           <1> ;; # Child's Page Table Entries are updated as duplicated child bit
15898                           <1> ;;   -AVL bit 0, PTE bit 9- is set.
15899                           <1> ;;
15900                           <1> ;; Duplicate page tables with read only pages (several sys fork system calls):
15901                           <1> ;; # Parent's read only pages are copied to new child pages.
15902                           <1> ;;   Parent's PTE attributes are not changed.
15903                           <1> ;;   (Because, there is another parent-child fork before this fork! We must not
15904                           <1> ;;    destroy/mix previous fork result).
15905                           <1> ;; # Child's Page Table Entries (which are corresponding to Parent's
15906                           <1> ;;   read only pages) are set as writable (while duplicated PTE bit is clear).
15907                           <1> ;; # Parent's PTEs with writable page attribute are updated to point same pages
15908                           <1> ;;   as read only, (while) duplicated PTE bit is reset (clear).
15909                           <1> ;; # Parent's Page Table Entries (with writable page attribute) are duplicated
```

```
15910                                  <1> ;;   as Child's Page Table Entries without copying actual page.
15911                                  <1> ;; # Child 's Page Table Entries (which are corresponding to Parent's writable
15912                                  <1> ;;   pages) are updated as duplicated PTE bit (AVL bit 0, PTE bit 9- is set.
15913                                  <1> ;;
15914                                  <1> ;; !? WHAT FOR (duplication after duplication):
15915                                  <1> ;; In UNIX method for sys fork (a typical 'fork' application in /etc/init)
15916                                  <1> ;; program/executable code continues from specified location as child process,
15917                                  <1> ;; returns back previous code location as parent process, every child after
15918                                  <1> ;; every sys fork uses last image of code and data just prior the fork.
15919                                  <1> ;; Even if the parent code changes data, the child will not see the changed data
15920                                  <1> ;; after the fork. In Retro UNIX 8086 v1, parent's process segment (32KB)
15921                                  <1> ;; was copied to child's process segment (all of code and data) according to
15922                                  <1> ;; original UNIX v1 which copies all of parent process code and data -core-
15923                                  <1> ;; to child space -core- but swaps that core image -of child- on to disk.
15924                                  <1> ;; If I (Erdogan Tan) would use a method of to copy parent's core
15925                                  <1> ;; (complete running image of parent process) to the child process;
15926                                  <1> ;; for big sizes, i would force Retro UNIX 386 v1 to spend many memory pages
15927                                  <1> ;; and times only for a sys fork. (It would excessive reservation for sys fork,
15928                                  <1> ;; because sys fork usually is prior to sys exec; sys exec always establishes
15929                                  <1> ;; a new/fresh core -running space-, by clearing all code/data content).
15930                                  <1> ;; 'Read Only' page flag ensures page fault handler is needed only for a few write
15931                                  <1> ;; attempts between sys fork and sys exec, not more... (I say so by thinking
15932                                  <1> ;; of "/etc/init" content, specially.) sys exec will clear page tables and
15933                                  <1> ;; new/fresh pages will be used to load and run new executable/program.
15934                                  <1> ;; That is what for i have preferred "copy on write", "duplication" method
15935                                  <1> ;; for sharing same read only pages between parent and child processes.
15936                                  <1> ;; That is a pitty i have to use new private flag (AVL bit 0, "duplicated PTE
15937                                  <1> ;; belongs to child" sign) for cooperation on duplicated pages between a parent
15938                                  <1> ;; and it's child processes; otherwise parent process would destroy data belongs
15939                                  <1> ;; to its child or vice versa; or some pages would remain unclaimed
15940                                  <1> ;; -deallocation problem-.
15941                                  <1> ;; Note: to prevent conflicts, read only pages must not be swapped out...
15942                                  <1> ;;
15943                                  <1> ;; WHEN PARENT TRIES TO WRITE IT'S READ ONLY (DUPLICATED) PAGE:
15944                                  <1> ;; # Page fault handler will do those:
15945                                  <1> ;;   - 'Duplicated PTE' flag (PTE bit 9) is checked (on the failed PTE).
15946                                  <1> ;;   - If it is reset/clear, there is a child uses same page.
15947                                  <1> ;;   - Parent's read only page -previous page- is copied to a new writable page.
15948                                  <1> ;;   - Parent's PTE is updated as writable page, as unique page (AVL=0)
15949                                  <1> ;;   - (Page fault handler whill check this PTE later, if child process causes to
15950                                  <1> ;;     page fault due to write attempt on read only page. Of course, the previous
15951                                  <1> ;;     read only page will be converted to writable and unique page which belongs
15952                                  <1> ;;     to child process.)
15953                                  <1> ;; WHEN CHILD TRIES TO WRITE IT'S READ ONLY (DUPLICATED) PAGE:
15954                                  <1> ;; # Page fault handler will do those:
15955                                  <1> ;;   - 'Duplicated PTE' flag (PTE bit 9) is checked (on the failed PTE).
15956                                  <1> ;;   - If it is set, there is a parent uses -or was using- same page.
15957                                  <1> ;;   - Same PTE address within parent's page table is checked if it has same page
15958                                  <1> ;;     address or not.
15959                                  <1> ;;   - If parent's PTE has same address, child will continue with a new writable page.
15960                                  <1> ;;     Parent's PTE will point to same (previous) page as writable, unique (AVL=0).
15961                                  <1> ;;   - If parent's PTE has different address, child will continue with it's
15962                                  <1> ;;     own/same page but read only flag (0) will be changed to writable flag (1) and
15963                                  <1> ;;     'duplicated PTE (belongs to child)' flag/sign will be cleared/reset.
15964                                  <1> ;;
15965                                  <1> ;; NOTE: When a child process is terminated, read only flags of parent's page tables
15966                                  <1> ;;       will be set as writable (and unique) in case of child process was using
15967                                  <1> ;;       same pages with duplicated child PTE sign... Depending on sys fork and
15968                                  <1> ;;       duplication method details, it is not possible multiple child processes
15969                                  <1> ;;       were using same page with duplicated PTEs.
15970                                  <1> ;;
15971                                  <1> ;;//////////////////////////////////////////////////////////////////////////////
15972                                  <1>
15973                                  <1> ;; 08/10/2014
15974                                  <1> ;; 11/09/2014 - Retro UNIX 386 v1 PAGING (further) draft
15975                                  <1> ;;          by Erdogan Tan (Based on KolibriOS 'memory.inc')
15976                                  <1>
15977                                  <1> ;; 'allocate_page' code is derived and modified from KolibriOS
15978                                  <1> ;; 'alloc_page' procedure in 'memory.inc'
15979                                  <1> ;; (25/08/2014, Revision: 5057) file
15980                                  <1> ;; by KolibriOS Team (2004-2012)
15981                                  <1>
15982                                  <1> allocate_page:
15983                                  <1>       ; 01/07/2015
15984                                  <1>       ; 05/05/2015
15985                                  <1>       ; 30/04/2015
15986                                  <1>       ; 16/10/2014
15987                                  <1>       ; 08/10/2014
15988                                  <1>       ; 09/09/2014 (Retro UNIX 386 v1 - beginning)
15989                                  <1>       ;
15990                                  <1>       ; INPUT -> none
15991                                  <1>       ;
15992                                  <1>       ; OUTPUT ->
15993                                  <1>       ;     EAX = PHYSICAL (real/flat) ADDRESS OF THE ALLOCATED PAGE
15994                                  <1>       ;     (corresponding MEMORY ALLOCATION TABLE bit is RESET)
15995                                  <1>       ;
15996                                  <1>       ;     CF = 1 and EAX = 0
15997                                  <1>       ;              if there is not a free page to be allocated
15998                                  <1>       ;
15999                                  <1>       ; Modified Registers -> none (except EAX)
16000                                  <1>       ;
16001 00004B79 A1[D04D0100]            <1>       mov   eax, [free_pages]
16002 00004B7E 21C0                    <1>       and   eax, eax
16003 00004B80 7438                    <1>       jz    short out_of_memory
16004                                  <1>       ;
16005 00004B82 53                      <1>       push  ebx
16006 00004B83 51                      <1>       push  ecx
16007                                  <1>       ;
16008 00004B84 BB00001000              <1>       mov   ebx, MEM_ALLOC_TBL   ; Memory Allocation Table offset
16009 00004B89 89D9                    <1>       mov   ecx, ebx
16010                                  <1>                                  ; NOTE: 32 (first_page) is initial
16011                                  <1>                                  ; value of [next_page].
```

```
16012                                   <1>                                 ; It points to the first available
16013                                   <1>                                 ; page block for users (ring 3) ...
16014                                   <1>                                 ; (MAT offset 32 = 1024/32)
16015                                   <1>                                 ; (at the of the first 4 MB)
16016 00004B8B 031D[D44D0100]           <1>         add     ebx, [next_page] ; Free page searching starts from here
16017                                   <1>                                 ; next_free_page >> 5
16018 00004B91 030D[D84D0100]           <1>         add     ecx, [last_page] ; Free page searching ends here
16019                                   <1>                                 ; (total_pages - 1) >> 5
16020                                   <1> al_p_scan:
16021 00004B97 39CB                     <1>         cmp     ebx, ecx
16022 00004B99 770A                     <1>         ja      short al_p_notfound
16023                                   <1>         ;
16024                                   <1>         ; 01/07/2015
16025                                   <1>         ; AMD64 Architecture Programmer's Manual
16026                                   <1>         ; Volume 3:
16027                                   <1>         ; General-Purpose and System Instructions
16028                                   <1>         ;
16029                                   <1>         ; BSF - Bit Scan Forward
16030                                   <1>         ;
16031                                   <1>         ;   Searches the value in a register or a memory location
16032                                   <1>         ;   (second operand) for the least-significant set bit.
16033                                   <1>         ;   If a set bit is found, the instruction clears the zero flag (ZF)
16034                                   <1>         ;   and stores the index of the least-significant set bit in a destination
16035                                   <1>         ;   register (first operand). If the second operand contains 0,
16036                                   <1>         ;   the instruction sets ZF to 1 and does not change the contents of the
16037                                   <1>         ;   destination register. The bit index is an unsigned offset from bit 0
16038                                   <1>         ;   of the searched value
16039                                   <1>         ;
16040 00004B9B 0FBC03                   <1>         bsf     eax, [ebx] ; Scans source operand for first bit set (1).
16041                                   <1>                            ; Clear ZF if a bit is found set (1) and
16042                                   <1>                            ; loads the destination with an index to
16043                                   <1>                            ; first set bit. (0 -> 31)
16044                                   <1>                            ; Sets ZF to 1 if no bits are found set.
16045 00004B9E 7525                     <1>         jnz     short al_p_found ; ZF = 0 -> a free page has been found
16046                                   <1>                            ;
16047                                   <1>                            ; NOTE:  a Memory Allocation Table bit
16048                                   <1>                            ;       with value of 1 means
16049                                   <1>                            ;       the corresponding page is free
16050                                   <1>                            ;       (Retro UNIX 386 v1 feature only!)
16051 00004BA0 83C304                   <1>         add     ebx, 4
16052                                   <1>                            ; We return back for searching next page block
16053                                   <1>                            ; NOTE: [free_pages] is not ZERO; so,
16054                                   <1>                            ;     we always will find at least 1 free page here.
16055 00004BA3 EBF2                     <1>           jmp     short al_p_scan
16056                                   <1>         ;
16057                                   <1> al_p_notfound:
16058 00004BA5 81E900001000             <1>         sub     ecx, MEM_ALLOC_TBL
16059 00004BAB 890D[D44D0100]           <1>         mov     [next_page], ecx ; next/first free page = last page
16060                                   <1>                                  ; (deallocate_page procedure will change it)
16061 00004BB1 31C0                     <1>         xor     eax, eax
16062 00004BB3 A3[D04D0100]             <1>         mov     [free_pages], eax ; 0
16063 00004BB8 59                       <1>         pop     ecx
16064 00004BB9 5B                       <1>         pop     ebx
16065                                   <1>         ;
16066                                   <1> out_of_memory:
16067 00004BBA E85B040000               <1>         call    swap_out
16068 00004BBF 7325                     <1>         jnc     short al_p_ok  ; [free_pages] = 0, re-allocation by swap_out
16069                                   <1>         ;
16070 00004BC1 29C0                     <1>         sub     eax, eax ; 0
16071 00004BC3 F9                       <1>         stc
16072 00004BC4 C3                       <1>         retn
16073                                   <1>
16074                                   <1> al_p_found:
16075 00004BC5 89D9                     <1>         mov     ecx, ebx
16076 00004BC7 81E900001000             <1>         sub     ecx, MEM_ALLOC_TBL
16077 00004BCD 890D[D44D0100]           <1>         mov     [next_page], ecx ; Set first free page searching start
16078                                   <1>                                  ; address/offset (to the next)
16079 00004BD3 FF0D[D04D0100]           <1>           dec     dword [free_pages] ; 1 page has been allocated (X = X-1)
16080                                   <1>         ;
16081 00004BD9 0FB303                   <1>         btr     [ebx], eax     ; The destination bit indexed by the source value
16082                                   <1>                                ; is copied into the Carry Flag and then cleared
16083                                   <1>                                ; in the destination.
16084                                   <1>                                ;
16085                                   <1>                                ; Reset the bit which is corresponding to the
16086                                   <1>                                ; (just) allocated page.
16087                                   <1>         ; 01/07/2015 (4*8 = 32, 1 allocation byte = 8 pages)
16088 00004BDC C1E103                   <1>         shl     ecx, 3         ; (page block offset * 32) + page index
16089 00004BDF 01C8                     <1>         add     eax, ecx       ; = page number
16090 00004BE1 C1E00C                   <1>         shl     eax, 12              ; physical address of the page (flat/real value)
16091                                   <1>         ; EAX = physical address of memory page
16092                                   <1>         ;
16093                                   <1>         ; NOTE: The relevant page directory and page table entry will be updated
16094                                   <1>         ;       according to this EAX value...
16095 00004BE4 59                       <1>         pop     ecx
16096 00004BE5 5B                       <1>         pop     ebx
16097                                   <1> al_p_ok:
16098 00004BE6 C3                       <1>         retn
16099                                   <1>
16100                                   <1>
16101                                   <1> make_page_dir:
16102                                   <1>         ; 18/04/2015
16103                                   <1>         ; 12/04/2015
16104                                   <1>         ; 23/10/2014
16105                                   <1>         ; 16/10/2014
16106                                   <1>         ; 09/10/2014 ; (Retro UNIX 386 v1 - beginning)
16107                                   <1>         ;
16108                                   <1>         ; INPUT ->
16109                                   <1>         ;     none
16110                                   <1>         ; OUTPUT ->
16111                                   <1>         ;     (EAX = 0)
16112                                   <1>         ;     cf = 1 -> insufficient (out of) memory error
16113                                   <1>         ;     cf = 0 ->
```

```
16114                                  <1>       ;     u.pgdir = page directory (physical) address of the current
16115                                  <1>       ;              process/user.
16116                                  <1>       ;
16117                                  <1>       ; Modified Registers -> EAX
16118                                  <1>       ;
16119 00004BE7 E88DFFFFFF               <1>       call  allocate_page
16120 00004BEC 7216                     <1>       jc    short mkpd_error
16121                                  <1>       ;
16122 00004BEE A3[B8030600]             <1>       mov   [u.pgdir], eax    ; Page dir address for current user/process
16123                                  <1>                                 ; (Physical address)
16124                                  <1> clear_page:
16125                                  <1>       ; 18/04/2015
16126                                  <1>       ; 09/10/2014 ; (Retro UNIX 386 v1 - beginning)
16127                                  <1>       ;
16128                                  <1>       ; INPUT ->
16129                                  <1>       ;     EAX = physical address of the page
16130                                  <1>       ; OUTPUT ->
16131                                  <1>       ;     all bytes of the page will be cleared
16132                                  <1>       ;
16133                                  <1>       ; Modified Registers -> none
16134                                  <1>       ;
16135 00004BF3 57                       <1>       push  edi
16136 00004BF4 51                       <1>       push  ecx
16137 00004BF5 50                       <1>       push  eax
16138 00004BF6 B900040000               <1>       mov   ecx, PAGE_SIZE / 4
16139 00004BFB 89C7                     <1>       mov   edi, eax
16140 00004BFD 31C0                     <1>       xor   eax, eax
16141 00004BFF F3AB                     <1>       rep   stosd
16142 00004C01 58                       <1>       pop   eax
16143 00004C02 59                       <1>       pop   ecx
16144 00004C03 5F                       <1>       pop   edi
16145                                  <1> mkpd_error:
16146                                  <1> mkpt_error:
16147 00004C04 C3                       <1>       retn
16148                                  <1>
16149                                  <1> make_page_table:
16150                                  <1>       ; 23/06/2015
16151                                  <1>       ; 18/04/2015
16152                                  <1>       ; 12/04/2015
16153                                  <1>       ; 16/10/2014
16154                                  <1>       ; 09/10/2014 ; (Retro UNIX 386 v1 - beginning)
16155                                  <1>       ;
16156                                  <1>       ; INPUT ->
16157                                  <1>       ;     EBX = virtual (linear) address
16158                                  <1>       ;     ECX = page table attributes (lower 12 bits)
16159                                  <1>       ;           (higher 20 bits must be ZERO)
16160                                  <1>       ;           (bit 0 must be 1)
16161                                  <1>       ;     u.pgdir = page directory (physical) address
16162                                  <1>       ; OUTPUT ->
16163                                  <1>       ;     EDX = Page directory entry address
16164                                  <1>       ;     EAX = Page table address
16165                                  <1>       ;     cf = 1 -> insufficient (out of) memory error
16166                                  <1>       ;     cf = 0 -> page table address in the PDE (EDX)
16167                                  <1>       ;
16168                                  <1>       ; Modified Registers -> EAX, EDX
16169                                  <1>       ;
16170 00004C05 E86FFFFFFF               <1>       call  allocate_page
16171 00004C0A 72F8                     <1>       jc    short mkpt_error
16172 00004C0C E811000000               <1>       call  set_pde
16173 00004C11 EBE0                     <1>       jmp   short clear_page
16174                                  <1>
16175                                  <1> make_page:
16176                                  <1>       ; 24/07/2015
16177                                  <1>       ; 23/06/2015 ; (Retro UNIX 386 v1 - beginning)
16178                                  <1>       ;
16179                                  <1>       ; INPUT ->
16180                                  <1>       ;     EBX = virtual (linear) address
16181                                  <1>       ;     ECX = page attributes (lower 12 bits)
16182                                  <1>       ;           (higher 20 bits must be ZERO)
16183                                  <1>       ;           (bit 0 must be 1)
16184                                  <1>       ;     u.pgdir = page directory (physical) address
16185                                  <1>       ; OUTPUT ->
16186                                  <1>       ;     EBX = Virtual address
16187                                  <1>       ;     (EDX = PTE value)
16188                                  <1>       ;     EAX = Physical address
16189                                  <1>       ;     cf = 1 -> insufficient (out of) memory error
16190                                  <1>       ;
16191                                  <1>       ; Modified Registers -> EAX, EDX
16192                                  <1>       ;
16193 00004C13 E861FFFFFF               <1>       call  allocate_page
16194 00004C18 7207                     <1>       jc    short mkp_err
16195 00004C1A E821000000               <1>       call  set_pte
16196 00004C1F 73D2                     <1>       jnc   short clear_page ; 18/04/2015
16197                                  <1> mkp_err:
16198 00004C21 C3                       <1>       retn
16199                                  <1>
16200                                  <1>
16201                                  <1> set_pde:    ; Set page directory entry (PDE)
16202                                  <1>       ; 20/07/2015
16203                                  <1>       ; 18/04/2015
16204                                  <1>       ; 12/04/2015
16205                                  <1>       ; 23/10/2014
16206                                  <1>       ; 10/10/2014 ; (Retro UNIX 386 v1 - beginning)
16207                                  <1>       ;
16208                                  <1>       ; INPUT ->
16209                                  <1>       ;     EAX = physical address
16210                                  <1>       ;           (use present value if EAX = 0)
16211                                  <1>       ;     EBX = virtual (linear) address
16212                                  <1>       ;     ECX = page table attributes (lower 12 bits)
16213                                  <1>       ;           (higher 20 bits must be ZERO)
16214                                  <1>       ;           (bit 0 must be 1)
16215                                  <1>       ;     u.pgdir = page directory (physical) address
```

```
16216                                  <1>       ; OUTPUT ->
16217                                  <1>       ;      EDX = PDE address
16218                                  <1>       ;      EAX = page table address (physical)
16219                                  <1>       ;      ;(CF=1 -> Invalid page address)
16220                                  <1>       ;
16221                                  <1>       ; Modified Registers -> EDX
16222                                  <1>       ;
16223 00004C22 89DA                   <1>       mov    edx, ebx
16224 00004C24 C1EA16                 <1>       shr    edx, PAGE_D_SHIFT ; 22
16225 00004C27 C1E202                 <1>       shl    edx, 2 ; offset to page directory (1024*4)
16226 00004C2A 0315[B8030600]         <1>       add    edx, [u.pgdir]
16227                                  <1>       ;
16228 00004C30 21C0                   <1>       and    eax, eax
16229 00004C32 7506                   <1>       jnz    short spde_1
16230                                  <1>       ;
16231 00004C34 8B02                   <1>       mov    eax, [edx]  ; old PDE value
16232                                  <1>       ;test  al, 1
16233                                  <1>       ;jz    short spde_2
16234 00004C36 662500F0               <1>       and    ax, PDE_A_CLEAR ; 0F000h  ; clear lower 12 bits
16235                                  <1> spde_1:
16236                                  <1>       ;and   cx, 0FFFh
16237 00004C3A 8902                   <1>       mov    [edx], eax
16238 00004C3C 66090A                 <1>       or     [edx], cx
16239 00004C3F C3                     <1>       retn
16240                                  <1> ;spde_2: ; error
16241                                  <1> ;     stc
16242                                  <1> ;     retn
16243                                  <1>
16244                                  <1> set_pte:     ; Set page table entry (PTE)
16245                                  <1>       ; 24/07/2015
16246                                  <1>       ; 20/07/2015
16247                                  <1>       ; 23/06/2015
16248                                  <1>       ; 18/04/2015
16249                                  <1>       ; 12/04/2015
16250                                  <1>       ; 10/10/2014 ; (Retro UNIX 386 v1 - beginning)
16251                                  <1>       ;
16252                                  <1>       ; INPUT ->
16253                                  <1>       ;      EAX = physical page address
16254                                  <1>       ;            (use present value if EAX = 0)
16255                                  <1>       ;      EBX = virtual (linear) address
16256                                  <1>       ;      ECX = page attributes (lower 12 bits)
16257                                  <1>       ;            (higher 20 bits must be ZERO)
16258                                  <1>       ;            (bit 0 must be 1)
16259                                  <1>       ;      u.pgdir = page directory (physical) address
16260                                  <1>       ; OUTPUT ->
16261                                  <1>       ;      EAX = physical page address
16262                                  <1>       ;      (EDX = PTE value)
16263                                  <1>       ;      EBX = virtual address
16264                                  <1>       ;
16265                                  <1>       ;      CF = 1 -> error
16266                                  <1>       ;
16267                                  <1>       ; Modified Registers -> EAX, EDX
16268                                  <1>       ;
16269 00004C40 50                     <1>       push   eax
16270 00004C41 A1[B8030600]           <1>       mov    eax, [u.pgdir] ; 20/07/2015
16271 00004C46 E837000000             <1>       call   get_pde
16272                                  <1>              ; EDX = PDE address
16273                                  <1>              ; EAX = PDE value
16274 00004C4B 5A                     <1>       pop    edx ; physical page address
16275 00004C4C 722A                   <1>       jc     short spte_err ; PDE not present
16276                                  <1>       ;
16277 00004C4E 53                     <1>       push   ebx ; 24/07/2015
16278 00004C4F 662500F0               <1>       and    ax, PDE_A_CLEAR ; 0F000h ; clear lower 12 bits
16279                                  <1>                            ; EDX = PT address (physical)
16280 00004C53 C1EB0C                 <1>       shr    ebx, PAGE_SHIFT ; 12
16281 00004C56 81E3FF030000           <1>       and    ebx, PTE_MASK; 03FFh
16282                                  <1>                           ; clear higher 10 bits (PD bits)
16283 00004C5C C1E302                 <1>       shl    ebx, 2   ; offset to page table (1024*4)
16284 00004C5F 01C3                   <1>       add    ebx, eax
16285                                  <1>       ;
16286 00004C61 8B03                   <1>       mov    eax, [ebx] ; Old PTE value
16287 00004C63 A801                   <1>       test   al, 1
16288 00004C65 740C                   <1>       jz     short spte_0
16289 00004C67 09D2                   <1>       or     edx, edx
16290 00004C69 750F                   <1>       jnz    short spte_1
16291 00004C6B 662500F0               <1>       and    ax, PTE_A_CLEAR ; 0F000h ; clear lower 12 bits
16292 00004C6F 89C2                   <1>       mov    edx, eax
16293 00004C71 EB09                   <1>       jmp    short spte_2
16294                                  <1> spte_0:
16295                                  <1>       ; If this PTE contains a swap (disk) address,
16296                                  <1>       ; it can be updated by using 'swap_in' procedure
16297                                  <1>       ; only!
16298 00004C73 21C0                   <1>       and    eax, eax
16299 00004C75 7403                   <1>       jz     short spte_1
16300                                  <1>       ; 24/07/2015
16301                                  <1>       ; swapped page ! (on disk)
16302 00004C77 5B                     <1>       pop    ebx
16303                                  <1> spte_err:
16304 00004C78 F9                     <1>       stc
16305 00004C79 C3                     <1>       retn
16306                                  <1> spte_1:
16307 00004C7A 89D0                   <1>       mov    eax, edx
16308                                  <1> spte_2:
16309 00004C7C 09CA                   <1>       or     edx, ecx
16310                                  <1>       ; 23/06/2015
16311 00004C7E 8913                   <1>       mov    [ebx], edx ; PTE value in EDX
16312                                  <1>       ; 24/07/2015
16313 00004C80 5B                     <1>       pop    ebx
16314 00004C81 C3                     <1>       retn
16315                                  <1>
16316                                  <1> get_pde:     ; Get present value of the relevant PDE
16317                                  <1>       ; 20/07/2015
```

```
16318                                   <1>       ; 18/04/2015
16319                                   <1>       ; 12/04/2015
16320                                   <1>       ; 10/10/2014 ; (Retro UNIX 386 v1 - beginning)
16321                                   <1>       ;
16322                                   <1>       ; INPUT ->
16323                                   <1>       ;     EBX = virtual (linear) address
16324                                   <1>       ;     EAX = page directory (physical) address
16325                                   <1>       ; OUTPUT ->
16326                                   <1>       ;     EDX = Page directory entry address
16327                                   <1>       ;     EAX = Page directory entry value
16328                                   <1>       ;     CF = 1 -> PDE not present or invalid ?
16329                                   <1>       ; Modified Registers -> EDX, EAX
16330                                   <1>       ;
16331 00004C82 89DA                    <1>       mov   edx, ebx
16332 00004C84 C1EA16                  <1>       shr   edx, PAGE_D_SHIFT ; 22  (12+10)
16333 00004C87 C1E202                  <1>       shl   edx, 2 ; offset to page directory (1024*4)
16334 00004C8A 01C2                    <1>       add   edx, eax ; page directory address (physical)
16335 00004C8C 8B02                    <1>       mov   eax, [edx]
16336 00004C8E A801                    <1>       test  al, PDE_A_PRESENT ; page table is present or not !
16337 00004C90 751F                    <1>       jnz   short gpte_retn
16338 00004C92 F9                      <1>       stc
16339                                  <1> gpde_retn:
16340 00004C93 C3                      <1>       retn
16341                                  <1>
16342                                  <1> get_pte:
16343                                  <1>             ; Get present value of the relevant PTE
16344                                  <1>       ; 29/07/2015
16345                                  <1>       ; 20/07/2015
16346                                  <1>       ; 18/04/2015
16347                                  <1>       ; 12/04/2015
16348                                  <1>       ; 10/10/2014 ; (Retro UNIX 386 v1 - beginning)
16349                                  <1>       ;
16350                                  <1>       ; INPUT ->
16351                                  <1>       ;     EBX = virtual (linear) address
16352                                  <1>       ;     EAX = page directory (physical) address
16353                                  <1>       ; OUTPUT ->
16354                                  <1>       ;     EDX = Page table entry address (if CF=0)
16355                                  <1>       ;           Page directory entry address (if CF=1)
16356                                  <1>       ;            (Bit 0 value is 0 if PT is not present)
16357                                  <1>       ;     EAX = Page table entry value (page address)
16358                                  <1>       ;     CF = 1 -> PDE not present or invalid ?
16359                                  <1>       ; Modified Registers -> EAX, EDX
16360                                  <1>       ;
16361 00004C94 E8E9FFFFFF              <1>       call  get_pde
16362 00004C99 72F8                    <1>       jc    short gpde_retn     ; page table is not present
16363                                  <1>       ;jnc  short gpte_1
16364                                  <1>       ;retn
16365                                  <1> ;gpte_1:
16366 00004C9B 662500F0                <1>       and   ax, PDE_A_CLEAR ; 0F000h ; clear lower 12 bits
16367 00004C9F 89DA                    <1>       mov   edx, ebx
16368 00004CA1 C1EA0C                  <1>       shr   edx, PAGE_SHIFT ; 12
16369 00004CA4 81E2FF030000            <1>       and   edx, PTE_MASK; 03FFh
16370                                  <1>                   ; clear higher 10 bits (PD bits)
16371 00004CAA C1E202                  <1>       shl   edx, 2 ; offset from start of page table (1024*4)
16372 00004CAD 01C2                    <1>       add   edx, eax
16373 00004CAF 8B02                    <1>       mov   eax, [edx]
16374                                  <1> gpte_retn:
16375 00004CB1 C3                      <1>       retn
16376                                  <1>
16377                                  <1> deallocate_page_dir:
16378                                  <1>       ; 15/09/2015
16379                                  <1>       ; 05/08/2015
16380                                  <1>       ; 30/04/2015
16381                                  <1>       ; 28/04/2015
16382                                  <1>       ; 17/10/2014
16383                                  <1>       ; 12/10/2014 (Retro UNIX 386 v1 - beginning)
16384                                  <1>       ;
16385                                  <1>       ; INPUT ->
16386                                  <1>       ;     EAX = PHYSICAL ADDRESS OF THE PAGE DIRECTORY (CHILD)
16387                                  <1>       ;     EBX = PHYSICAL ADDRESS OF THE PARENT'S PAGE DIRECTORY
16388                                  <1>       ; OUTPUT ->
16389                                  <1>       ;     All of page tables in the page directory
16390                                  <1>       ;     and page dir's itself will be deallocated
16391                                  <1>       ;     except 'read only' duplicated pages (will be converted
16392                                  <1>       ;     to writable pages).
16393                                  <1>       ;
16394                                  <1>       ; Modified Registers -> EAX
16395                                  <1>       ;
16396                                  <1>       ;
16397 00004CB2 56                      <1>       push  esi
16398 00004CB3 51                      <1>       push  ecx
16399 00004CB4 50                      <1>       push  eax
16400 00004CB5 89C6                    <1>       mov   esi, eax
16401 00004CB7 31C9                    <1>       xor   ecx, ecx
16402                                  <1>       ; The 1st PDE points to Kernel Page Table 0 (the 1st 4MB),
16403                                  <1>       ; it must not be deallocated
16404 00004CB9 890E                    <1>       mov   [esi], ecx ; 0 ; clear PDE 0
16405                                  <1> dapd_0:
16406 00004CBB AD                      <1>       lodsd
16407 00004CBC A801                    <1>       test  al, PDE_A_PRESENT ; bit 0, present flag (must be 1)
16408 00004CBE 7409                    <1>       jz    short dapd_1
16409 00004CC0 662500F0                <1>       and   ax, PDE_A_CLEAR ; 0F000h ; clear lower 12 (attribute) bits
16410 00004CC4 E812000000              <1>       call  deallocate_page_table
16411                                  <1> dapd_1:
16412 00004CC9 41                      <1>       inc   ecx ; page directory entry index
16413 00004CCA 81F900040000            <1>       cmp   ecx, PAGE_SIZE / 4 ; 1024
16414 00004CD0 72E9                    <1>       jb    short dapd_0
16415                                  <1> dapd_2:
16416 00004CD2 58                      <1>       pop   eax
16417 00004CD3 E87F000000              <1>       call  deallocate_page     ; deallocate the page dir's itself
16418 00004CD8 59                      <1>       pop   ecx
16419 00004CD9 5E                      <1>       pop   esi
```

```
16420 00004CDA C3                <1>       retn
16421                            <1>
16422                            <1> deallocate_page_table:
16423                            <1>       ; 12/07/2016
16424                            <1>       ; 19/09/2015
16425                            <1>       ; 15/09/2015
16426                            <1>       ; 05/08/2015
16427                            <1>       ; 30/04/2015
16428                            <1>       ; 28/04/2015
16429                            <1>       ; 24/10/2014
16430                            <1>       ; 23/10/2014
16431                            <1>       ; 12/10/2014 (Retro UNIX 386 v1 - beginning)
16432                            <1>       ;
16433                            <1>       ; INPUT ->
16434                            <1>       ;     EAX = PHYSICAL (real/flat) ADDRESS OF THE PAGE TABLE
16435                            <1>       ;     EBX = PHYSICAL ADDRESS OF THE PARENT'S PAGE DIRECTORY
16436                            <1>       ;     (ECX = page directory entry index)
16437                            <1>       ; OUTPUT ->
16438                            <1>       ;     All of pages in the page table and page table's itself
16439                            <1>       ;     will be deallocated except 'read only' duplicated pages
16440                            <1>       ;     (will be converted to writable pages).
16441                            <1>       ;
16442                            <1>       ; Modified Registers -> EAX
16443                            <1>       ;
16444 00004CDB 56                <1>       push   esi
16445 00004CDC 57                <1>       push   edi
16446 00004CDD 52                <1>       push   edx
16447 00004CDE 50                <1>       push   eax ; *
16448 00004CDF 89C6              <1>       mov    esi, eax
16449 00004CE1 31FF              <1>       xor    edi, edi ; 0
16450                            <1> dapt_0:
16451 00004CE3 AD                <1>       lodsd
16452 00004CE4 A801              <1>       test   al, PTE_A_PRESENT ; bit 0, present flag (must be 1)
16453 00004CE6 7441              <1>       jz     short dapt_1
16454                            <1>       ;
16455 00004CE8 A802              <1>       test   al, PTE_A_WRITE   ; bit 1, writable (r/w) flag
16456                            <1>                                ; (must be 1)
16457 00004CEA 754C              <1>       jnz    short dapt_3
16458                            <1>       ; Read only -duplicated- page (belongs to a parent or a child)
16459 00004CEC 66A90002          <1>         test    ax, PTE_DUPLICATED ; Was this page duplicated
16460                            <1>                                    ; as child's page ?
16461 00004CF0 7451              <1>       jz     short dapt_4 ; Clear PTE but don't deallocate the page!
16462                            <1>       ; check the parent's PTE value is read only & same page or not..
16463                            <1>       ; ECX = page directory entry index (0-1023)
16464 00004CF2 53                <1>       push   ebx
16465 00004CF3 51                <1>       push   ecx
16466 00004CF4 66C1E102          <1>       shl    cx, 2 ; *4
16467 00004CF8 01CB              <1>       add    ebx, ecx ; PDE offset (for the parent)
16468 00004CFA 8B0B              <1>       mov    ecx, [ebx]
16469 00004CFC F6C101            <1>       test   cl, PDE_A_PRESENT ; present (valid) or not ?
16470 00004CFF 7435              <1>       jz     short dapt_2 ; parent process does not use this page
16471 00004D01 6681E100F0        <1>       and    cx, PDE_A_CLEAR ; 0F000h ; Clear attribute bits
16472                            <1>       ; EDI = page table entry index (0-1023)
16473 00004D06 89FA              <1>       mov    edx, edi
16474 00004D08 66C1E202          <1>       shl    dx, 2 ; *4
16475 00004D0C 01CA              <1>       add    edx, ecx ; PTE offset (for the parent)
16476 00004D0E 8B1A              <1>       mov    ebx, [edx]
16477 00004D10 F6C301            <1>       test   bl, PTE_A_PRESENT ; present or not ?
16478 00004D13 7421              <1>       jz     short dapt_2 ; parent process does not use this page
16479 00004D15 662500F0          <1>       and    ax, PTE_A_CLEAR ; 0F000h ; Clear attribute bits
16480 00004D19 6681E300F0        <1>       and    bx, PTE_A_CLEAR ; 0F000h ; Clear attribute bits
16481 00004D1E 39D8              <1>       cmp    eax, ebx     ; parent's and child's pages are same ?
16482 00004D20 7514              <1>       jne    short dapt_2 ; not same page
16483                            <1>                           ; deallocate the child's page
16484 00004D22 800A02            <1>         or      byte [edx], PTE_A_WRITE ; convert to writable page (parent)
16485 00004D25 59                <1>       pop    ecx
16486 00004D26 5B                <1>       pop    ebx
16487 00004D27 EB1A              <1>       jmp    short dapt_4
16488                            <1> dapt_1:
16489 00004D29 09C0              <1>       or     eax, eax     ; swapped page ?
16490 00004D2B 741D              <1>       jz     short dapt_5 ; no
16491                            <1>                           ; yes
16492 00004D2D D1E8              <1>       shr    eax, 1
16493 00004D2F E8CA040000        <1>       call   unlink_swap_block ; Deallocate swapped page block
16494                            <1>                                ; on the swap disk (or in file)
16495 00004D34 EB14              <1>       jmp    short dapt_5
16496                            <1> dapt_2:
16497 00004D36 59                <1>       pop    ecx
16498 00004D37 5B                <1>       pop    ebx
16499                            <1> dapt_3:
16500                            <1>       ; 12/07/2016
16501 00004D38 66A90004          <1>       test   ax, PTE_SHARED ; shared or direct memory access indicator
16502 00004D3C 7505              <1>       jnz    short dapt_4   ; AVL bit 1 = 1, do not deallocate this page!
16503                            <1>       ;
16504                            <1>       ;and   ax, PTE_A_CLEAR ; 0F000h ; clear lower 12 (attribute) bits
16505 00004D3E E814000000        <1>       call   deallocate_page ; set the mem allocation bit of this page
16506                            <1> dapt_4:
16507 00004D43 C746FC00000000    <1>       mov    dword [esi-4], 0 ; clear/reset PTE (child, dupl. as parent)
16508                            <1> dapt_5:
16509 00004D4A 47                <1>       inc    edi ; page table entry index
16510 00004D4B 81FF00040000      <1>       cmp    edi, PAGE_SIZE / 4 ; 1024
16511 00004D51 7290              <1>       jb     short dapt_0
16512                            <1>       ;
16513 00004D53 58                <1>       pop    eax ; *
16514 00004D54 5A                <1>       pop    edx
16515 00004D55 5F                <1>       pop    edi
16516 00004D56 5E                <1>       pop    esi
16517                            <1>       ;
16518                            <1>       ;call  deallocate_page     ; deallocate the page table's itself
16519                            <1>       ;retn
16520                            <1>
16521                            <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>       ;
00004D57 53                     <1>       push   ebx
00004D58 52                     <1>       push   edx
                                <1>       ;
00004D59 C1E80C                 <1>       shr    eax, PAGE_SHIFT      ; shift physical address to
                                <1>                                 ; 12 bits right
                                <1>                                 ; to get page number
00004D5C 89C2                   <1>       mov    edx, eax
                                <1>       ; 15/09/2015
00004D5E C1EA03                 <1>       shr    edx, 3               ; to get offset to M.A.T.
                                <1>                                 ; (1 allocation bit = 1 page)
                                <1>                                 ; (1 allocation bytes = 8 pages)
00004D61 80E2FC                 <1>       and    dl, 0FCh             ; clear lower 2 bits
                                <1>                                 ; (to get 32 bit position)
                                <1>       ;
00004D64 BB00001000             <1>       mov    ebx, MEM_ALLOC_TBL   ; Memory Allocation Table address
00004D69 01D3                   <1>       add    ebx, edx
00004D6B 83E01F                 <1>       and    eax, 1Fh             ; lower 5 bits only
                                <1>                                 ; (allocation bit position)
00004D6E 3B15[D44D0100]         <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)
00004D74 7306                   <1>       jnb    short dap_1          ; no
00004D76 8915[D44D0100]         <1>       mov    [next_page], edx     ; yes
                                <1> dap_1:
00004D7C 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.
00004D7F FF05[D04D0100]         <1>         inc     dword [free_pages]
                                <1> dap_2:
00004D85 5A                     <1>       pop    edx
00004D86 5B                     <1>       pop    ebx
00004D87 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:
```

```
16624                                  <1> ;;        mov     [pg_data.pages_free], 1
16625                                  <1> ;;        xor     eax, eax
16626                                  <1> ;;        pop     ebx
16627                                  <1> ;;        popfd
16628                                  <1> ;;        ret
16629                                  <1> ;;;//-
16630                                  <1> ;;endp
16631                                  <1>
16632                                  <1> duplicate_page_dir:
16633                                  <1>       ; 21/09/2015
16634                                  <1>       ; 31/08/2015
16635                                  <1>       ; 20/07/2015
16636                                  <1>       ; 28/04/2015
16637                                  <1>       ; 27/04/2015
16638                                  <1>       ; 18/04/2015
16639                                  <1>       ; 12/04/2015
16640                                  <1>       ; 18/10/2014
16641                                  <1>       ; 16/10/2014 (Retro UNIX 386 v1 - beginning)
16642                                  <1>       ;
16643                                  <1>       ; INPUT ->
16644                                  <1>       ;     [u.pgdir] = PHYSICAL (real/flat) ADDRESS of the parent's
16645                                  <1>       ;                 page directory.
16646                                  <1>       ; OUTPUT ->
16647                                  <1>       ;     EAX =  PHYSICAL (real/flat) ADDRESS of the child's
16648                                  <1>       ;            page directory.
16649                                  <1>       ;     (New page directory with new page table entries.)
16650                                  <1>       ;     (New page tables with read only copies of the parent's
16651                                  <1>       ;     pages.)
16652                                  <1>       ;     EAX = 0 -> Error (CF = 1)
16653                                  <1>       ;
16654                                  <1>       ; Modified Registers -> none (except EAX)
16655                                  <1>       ;
16656 00004D88 E8ECFDFFFF             <1>       call  allocate_page
16657 00004D8D 723E                   <1>       jc    short dpd_err
16658                                  <1>       ;
16659 00004D8F 55                     <1>       push  ebp ; 20/07/2015
16660 00004D90 56                     <1>       push  esi
16661 00004D91 57                     <1>       push  edi
16662 00004D92 53                     <1>       push  ebx
16663 00004D93 51                     <1>       push  ecx
16664 00004D94 8B35[B8030600]         <1>       mov   esi, [u.pgdir]
16665 00004D9A 89C7                   <1>       mov   edi, eax
16666 00004D9C 50                     <1>       push  eax ; save child's page directory address
16667                                  <1>       ; 31/08/2015
16668                                  <1>       ; copy PDE 0 from the parent's page dir to the child's page dir
16669                                  <1>       ; (use same system space for all user page tables)
16670 00004D9D A5                     <1>       movsd
16671 00004D9E BD00004000             <1>       mov   ebp, 1024*4096 ; pass the 1st 4MB (system space)
16672 00004DA3 B9FF030000             <1>       mov   ecx, (PAGE_SIZE / 4) - 1 ; 1023
16673                                  <1> dpd_0:
16674 00004DA8 AD                     <1>       lodsd
16675                                  <1>       ;or   eax, eax
16676                                  <1>         ;jnz     short dpd_1
16677 00004DA9 A801                   <1>       test  al, PDE_A_PRESENT ;  bit 0 =  1
16678 00004DAB 7508                   <1>       jnz   short dpd_1
16679                                  <1>       ; 20/07/2015 (virtual address at the end of the page table)
16680 00004DAD 81C500004000           <1>       add   ebp, 1024*4096 ; page size * PTE count
16681 00004DB3 EB0F                   <1>       jmp   short dpd_2
16682                                  <1> dpd_1:
16683 00004DB5 662500F0               <1>       and   ax, PDE_A_CLEAR ; 0F000h ; clear attribute bits
16684 00004DB9 89C3                   <1>       mov   ebx, eax
16685                                  <1>       ; EBX = Parent's page table address
16686 00004DBB E81F000000             <1>       call  duplicate_page_table
16687 00004DC0 720C                   <1>       jc    short dpd_p_err
16688                                  <1>       ; EAX = Child's page table address
16689 00004DC2 0C07                   <1>       or    al, PDE_A_PRESENT + PDE_A_WRITE + PDE_A_USER
16690                                  <1>                     ; set bit 0, bit 1 and bit 2 to 1
16691                                  <1>                     ; (present, writable, user)
16692                                  <1> dpd_2:
16693 00004DC4 AB                     <1>       stosd
16694 00004DC5 E2E1                   <1>       loop  dpd_0
16695                                  <1>       ;
16696 00004DC7 58                     <1>       pop   eax  ; restore child's page directory address
16697                                  <1> dpd_3:
16698 00004DC8 59                     <1>       pop   ecx
16699 00004DC9 5B                     <1>       pop   ebx
16700 00004DCA 5F                     <1>       pop   edi
16701 00004DCB 5E                     <1>       pop   esi
16702 00004DCC 5D                     <1>       pop   ebp ; 20/07/2015
16703                                  <1> dpd_err:
16704 00004DCD C3                     <1>       retn
16705                                  <1> dpd_p_err:
16706                                  <1>       ; release the allocated pages missing (recover free space)
16707 00004DCE 58                     <1>       pop   eax  ; the new page directory address (physical)
16708 00004DCF 8B1D[B8030600]         <1>       mov   ebx, [u.pgdir] ; parent's page directory address
16709 00004DD5 E8D8FEFFFF             <1>       call  deallocate_page_dir
16710 00004DDA 29C0                   <1>       sub   eax, eax ; 0
16711 00004DDC F9                     <1>       stc
16712 00004DDD EBE9                   <1>       jmp   short dpd_3
16713                                  <1>
16714                                  <1> duplicate_page_table:
16715                                  <1>       ; 20/02/2017
16716                                  <1>       ; 21/09/2015
16717                                  <1>       ; 20/07/2015
16718                                  <1>       ; 05/05/2015
16719                                  <1>       ; 28/04/2015
16720                                  <1>       ; 27/04/2015
16721                                  <1>       ; 18/04/2015
16722                                  <1>       ; 18/10/2014
16723                                  <1>       ; 16/10/2014 (Retro UNIX 386 v1 - beginning)
16724                                  <1>       ;
16725                                  <1>       ; INPUT ->
```

```
16726                                  <1>       ;      EBX = PHYSICAL (real/flat) ADDRESS of the parent's page table.
16727                                  <1>       ;        20/02/2017
16728                                  <1>       ;      EBP = Linear address of the page (from 'duplicate_page_dir')
16729                                  <1>       ;            (Linear address = CORE + user's virtual address)
16730                                  <1>       ; OUTPUT ->
16731                                  <1>       ;      EAX = PHYSICAL (real/flat) ADDRESS of the child's page table.
16732                                  <1>       ;            (with 'read only' attribute of page table entries)
16733                                  <1>       ;      20/02/2017
16734                                  <1>       ;      EBP = Next linear page address (for 'duplicate_page_dir')
16735                                  <1>       ;
16736                                  <1>       ;      CF = 1 -> error
16737                                  <1>       ;
16738                                  <1>       ; Modified Registers -> EBP (except EAX)
16739                                  <1>       ;
16740 00004DDF E895FDFFFF              <1>       call   allocate_page
16741 00004DE4 726A                    <1>       jc     short dpt_err
16742                                  <1>       ;
16743 00004DE6 50                      <1>       push   eax ; *
16744 00004DE7 56                      <1>       push   esi
16745 00004DE8 57                      <1>       push   edi
16746 00004DE9 52                      <1>       push   edx
16747 00004DEA 51                      <1>       push   ecx
16748                                  <1>       ;
16749 00004DEB 89DE                    <1>       mov    esi, ebx
16750 00004DED 89C7                    <1>       mov    edi, eax
16751 00004DEF 89C2                    <1>       mov    edx, eax
16752 00004DF1 81C200100000            <1>       add    edx, PAGE_SIZE
16753                                  <1> dpt_0:
16754 00004DF7 AD                      <1>       lodsd
16755 00004DF8 21C0                    <1>       and    eax, eax
16756 00004DFA 7444                    <1>       jz     short dpt_3
16757 00004DFC A801                    <1>       test   al, PTE_A_PRESENT ;  bit 0 =  1
16758 00004DFE 7507                    <1>       jnz    short dpt_1
16759                                  <1>       ; 20/07/2015
16760                                  <1>       ; ebp = virtual (linear) address of the memory page
16761 00004E00 E83F050000              <1>       call   reload_page ; 28/04/2015
16762 00004E05 7244                    <1>       jc     short dpt_p_err
16763                                  <1> dpt_1:
16764                                  <1>       ; 21/09/2015
16765 00004E07 89C1                    <1>       mov    ecx, eax
16766 00004E09 662500F0                <1>       and    ax, PTE_A_CLEAR ; 0F000h ; clear attribute bits
16767 00004E0D F6C102                  <1>       test   cl, PTE_A_WRITE ; writable page ?
16768 00004E10 7525                    <1>       jnz    short dpt_2
16769                                  <1>       ; Read only (parent) page
16770                                  <1>       ;      - there is a third process which uses this page -
16771                                  <1>       ; Allocate a new page for the child process
16772 00004E12 E862FDFFFF              <1>       call   allocate_page
16773 00004E17 7232                    <1>       jc     short dpt_p_err
16774 00004E19 57                      <1>       push   edi
16775 00004E1A 56                      <1>       push   esi
16776 00004E1B 89CE                    <1>       mov    esi, ecx
16777 00004E1D 89C7                    <1>       mov    edi, eax
16778 00004E1F B900040000              <1>       mov    ecx, PAGE_SIZE/4
16779 00004E24 F3A5                    <1>       rep    movsd  ; copy page (4096 bytes)
16780 00004E26 5E                      <1>       pop    esi
16781 00004E27 5F                      <1>       pop    edi
16782                                  <1>       ;
16783 00004E28 53                      <1>       push   ebx
16784 00004E29 50                      <1>       push   eax
16785                                  <1>       ; 20/07/2015
16786 00004E2A 89EB                    <1>       mov    ebx, ebp
16787                                  <1>       ; ebx = virtual (linear) address of the memory page
16788 00004E2C E887030000              <1>       call   add_to_swap_queue
16789 00004E31 58                      <1>       pop    eax
16790 00004E32 5B                      <1>       pop    ebx
16791                                  <1>       ; 21/09/2015
16792 00004E33 0C07                    <1>       or     al, PTE_A_USER+PTE_A_WRITE+PTE_A_PRESENT
16793                                  <1>              ; user + writable + present page
16794 00004E35 EB09                    <1>       jmp    short dpt_3
16795                                  <1> dpt_2:
16796                                  <1>       ;or    ax, PTE_A_USER+PTE_A_PRESENT
16797 00004E37 0C05                    <1>       or     al, PTE_A_USER+PTE_A_PRESENT
16798                                  <1>              ; (read only page!)
16799 00004E39 8946FC                  <1>       mov    [esi-4], eax ; update parent's PTE
16800 00004E3C 660D0002                <1>       or      ax, PTE_DUPLICATED  ; (read only page & duplicated PTE!)
16801                                  <1> dpt_3:
16802 00004E40 AB                      <1>       stosd  ; EDI points to child's PTE
16803                                  <1>       ;
16804 00004E41 81C500100000            <1>       add    ebp, 4096 ; 20/07/2015 (next page)
16805                                  <1>       ;
16806 00004E47 39D7                    <1>       cmp    edi, edx
16807 00004E49 72AC                    <1>       jb     short dpt_0
16808                                  <1> dpt_p_err:
16809 00004E4B 59                      <1>       pop    ecx
16810 00004E4C 5A                      <1>       pop    edx
16811 00004E4D 5F                      <1>       pop    edi
16812 00004E4E 5E                      <1>       pop    esi
16813 00004E4F 58                      <1>       pop    eax ; *
16814                                  <1> dpt_err:
16815 00004E50 C3                      <1>       retn
16816                                  <1>
16817                                  <1> page_fault_handler:      ; CPU EXCEPTION 0Eh (14) : Page Fault !
16818                                  <1>       ; 21/09/2015
16819                                  <1>       ; 19/09/2015
16820                                  <1>       ; 17/09/2015
16821                                  <1>       ; 28/08/2015
16822                                  <1>       ; 20/07/2015
16823                                  <1>       ; 28/06/2015
16824                                  <1>       ; 03/05/2015
16825                                  <1>       ; 30/04/2015
16826                                  <1>       ; 18/04/2015
16827                                  <1>       ; 12/04/2015
```

```
16828                           <1>     ; 30/10/2014
16829                           <1>     ; 11/09/2014
16830                           <1>     ; 10/09/2014 (Retro UNIX 386 v1 - beginning)
16831                           <1>     ;
16832                           <1>     ; Note: This is not an interrupt/exception handler.
16833                           <1>     ;     This is a 'page fault remedy' subroutine
16834                           <1>     ;     which will be called by standard/uniform
16835                           <1>     ;     exception handler.
16836                           <1>     ;
16837                           <1>     ; INPUT ->
16838                           <1>     ;     [error_code] = 32 bit ERROR CODE (lower 5 bits are valid)
16839                           <1>     ;
16840                           <1>     ;     cr2 = the virtual (linear) address
16841                           <1>     ;           which has caused to page fault (19/09/2015)
16842                           <1>     ;
16843                           <1>     ; OUTPUT ->
16844                           <1>     ;     (corresponding PAGE TABLE ENTRY is mapped/set)
16845                           <1>     ;     EAX = 0 -> no error
16846                           <1>     ;     EAX > 0 -> error code in EAX (also CF = 1)
16847                           <1>     ;
16848                           <1>     ; Modified Registers -> none (except EAX)
16849                           <1>     ;
16850                           <1>       ;
16851                           <1>       ; ERROR CODE:
16852                           <1>     ;      31  .....   4   3  2   1  0
16853                           <1>     ;     +---+-- --+---+---+---+---+---+---+
16854                           <1>     ;     |   Reserved  | I | R | U | W | P |
16855                           <1>     ;     +---+-- --+---+---+---+---+---+---+
16856                           <1>     ;
16857                           <1>     ; P : PRESENT -     When set, the page fault was caused by
16858                           <1>     ;             a page-protection violation. When not set,
16859                           <1>     ;             it was caused by a non-present page.
16860                           <1>     ; W : WRITE   -     When set, the page fault was caused by
16861                           <1>     ;             a page write. When not set, it was caused
16862                           <1>     ;             by a page read.
16863                           <1>     ; U : USER    -     When set, the page fault was caused
16864                           <1>     ;             while CPL = 3.
16865                           <1>     ;             This does not necessarily mean that
16866                           <1>     ;             the page fault was a privilege violation.
16867                           <1>     ; R : RESERVD -     When set, the page fault was caused by
16868                           <1>     ;     WRITE  reading a 1 in a reserved field.
16869                           <1>     ; I : INSTRUC -     When set, the page fault was caused by
16870                           <1>     ;     FETCH  an instruction fetch
16871                           <1>     ;
16872                           <1>     ;; x86 (32 bit) VIRTUAL ADDRESS TRANSLATION
16873                           <1>     ;  31               22                  12 11                    0
16874                           <1>     ; +-------------------+-------------------+-----------------------+
16875                           <1>           ; | PAGE DIR. ENTRY # | PAGE TAB. ENTRY # |        OFFSET         |
16876                           <1>           ; +-------------------+-------------------+-----------------------+
16877                           <1>     ;
16878                           <1>
16879                           <1>     ;; CR3 REGISTER (Control Register 3)
16880                           <1>     ;  31                                   12             5 4 3 2   0
16881                           <1>     ; +-----------------------------------------+-------------+---+-----+
16882                           <1>           ; |                                         |          |P|P|     |
16883                           <1>           ; |   PAGE DIRECTORY TABLE BASE ADDRESS     |  reserved      |C|W|rsvrd|
16884                           <1>           ; |                                         |          |D|T|     |
16885                           <1>     ; +-----------------------------------------+-------------+---+-----+
16886                           <1>     ;
16887                           <1>     ;     PWT    - WRITE THROUGH
16888                           <1>     ;     PCD    - CACHE DISABLE
16889                           <1>     ;
16890                           <1>     ;
16891                           <1>     ;; x86 PAGE DIRECTORY ENTRY (4 KByte Page)
16892                           <1>     ;  31                                   12 11  9 8 7 6 5 4 3 2 1 0
16893                           <1>     ; +-----------------------------------------+-----+---+-+-+---+-+-+-+
16894                           <1>           ; |                                         |     | | | | |P|P|U|R| |
16895                           <1>           ; |     PAGE TABLE BASE ADDRESS 31..12      | AVL |G|0|D|A|C|W|/|/|P|
16896                           <1>           ; |                                         |     | | | | |D|T|S|W| |
16897                           <1>     ; +-----------------------------------------+-----+---+-+-+---+-+-+-+
16898                           <1>     ;
16899                           <1>       ;       P      - PRESENT
16900                           <1>       ;       R/W    - READ/WRITE
16901                           <1>       ;       U/S    - USER/SUPERVISOR
16902                           <1>     ;     PWT    - WRITE THROUGH
16903                           <1>     ;     PCD    - CACHE DISABLE
16904                           <1>     ;     A      - ACCESSED
16905                           <1>       ;       D      - DIRTY (IGNORED)
16906                           <1>     ;     PAT    - PAGE ATTRIBUTE TABLE INDEX (CACHE BEHAVIOR)
16907                           <1>     ;     G      - GLOBAL     (IGNORED)
16908                           <1>       ;       AVL    - AVAILABLE FOR SYSTEMS PROGRAMMER USE
16909                           <1>     ;
16910                           <1>     ;
16911                           <1>     ;; x86 PAGE TABLE ENTRY (4 KByte Page)
16912                           <1>     ;  31                                   12 11  9 8 7 6 5 4 3 2 1 0
16913                           <1>     ; +-----------------------------------------+-----+---+-+-+---+-+-+-+
16914                           <1>           ; |                                         |     | |P| | |P|P|U|R| |
16915                           <1>           ; |     PAGE FRAME BASE ADDRESS 31..12      | AVL |G|A|D|A|C|W|/|/|P|
16916                           <1>           ; |                                         |     | |T| | |D|T|S|W| |
16917                           <1>     ; +-----------------------------------------+-----+---+-+-+---+-+-+-+
16918                           <1>     ;
16919                           <1>       ;       P      - PRESENT
16920                           <1>       ;       R/W    - READ/WRITE
16921                           <1>       ;       U/S    - USER/SUPERVISOR
16922                           <1>     ;     PWT    - WRITE THROUGH
16923                           <1>     ;     PCD    - CACHE DISABLE
16924                           <1>     ;     A      - ACCESSED
16925                           <1>       ;       D      - DIRTY
16926                           <1>     ;     PAT    - PAGE ATTRIBUTE TABLE INDEX (CACHE BEHAVIOR)
16927                           <1>     ;     G      - GLOBAL
16928                           <1>       ;       AVL    - AVAILABLE FOR SYSTEMS PROGRAMMER USE
16929                           <1>     ;
```

```
16930                                  <1>       ;
16931                                  <1>       ;; 80386 PAGE TABLE ENTRY (4 KByte Page)
16932                                  <1>       ;  31                                   12 11  9 8 7 6 5 4 3 2 1 0
16933                                  <1>       ; +---------------------------------------+-----+-+-+-+-+---+-+-+-+
16934                                  <1>            ; |                                       |     | | | | | | |U|R| |
16935                                  <1>            ; |     PAGE FRAME BASE ADDRESS 31..12    | AVL |0|0|D|A|0|0|/|/|P|
16936                                  <1>            ; |                                       |     | | | | | | |S|W| |
16937                                  <1>            ; +---------------------------------------+-----+-+-+-+-+---+-+-+-+
16938                                  <1>       ;
16939                                  <1>         ;       P      - PRESENT
16940                                  <1>         ;       R/W    - READ/WRITE
16941                                  <1>         ;       U/S    - USER/SUPERVISOR
16942                                  <1>         ;       D      - DIRTY
16943                                  <1>         ;       AVL    - AVAILABLE FOR SYSTEMS PROGRAMMER USE
16944                                  <1>       ;
16945                                  <1>         ;       NOTE: 0 INDICATES INTEL RESERVED. DO NOT DEFINE.
16946                                  <1>       ;
16947                                  <1>       ;
16948                                  <1>       ;; Invalid Page Table Entry
16949                                  <1>       ; 31                                                    1 0
16950                                  <1>            ; +-------------------------------------------------------------+-+
16951                                  <1>            ; |                                                             | |
16952                                  <1>            ; |                           AVAILABLE                         |0|
16953                                  <1>            ; |                                                             | |
16954                                  <1>            ; +-------------------------------------------------------------+-+
16955                                  <1>       ;
16956                                  <1>
16957 00004E51 53                      <1>       push   ebx
16958 00004E52 52                      <1>       push   edx
16959 00004E53 51                      <1>       push   ecx
16960                                  <1>       ;
16961                                  <1>       ; 21/09/2015 (debugging)
16962 00004E54 FF05[CC030600]          <1>       inc    dword [u.pfcount] ; page fault count for running process
16963 00004E5A FF05[80050600]          <1>       inc    dword [PF_Count] ; total page fault count
16964                                  <1>       ; 28/06/2015
16965                                  <1>       ;mov   edx, [error_code] ; Lower 5 bits are valid
16966 00004E60 8A15[78050600]          <1>       mov    dl, [error_code]
16967                                  <1>       ;
16968 00004E66 F6C201                  <1>       test   dl, 1  ; page fault was caused by a non-present page
16969                                  <1>                     ; sign
16970 00004E69 7422                    <1>       jz     short pfh_alloc_np
16971                                  <1>       ;
16972                                  <1>       ; If it is not a 'write on read only page' type page fault
16973                                  <1>       ; major page fault error with minor reason must be returned without
16974                                  <1>       ; fixing the problem. 'sys_exit with error' will be needed
16975                                  <1>       ; after return here!
16976                                  <1>       ; Page fault will be remedied, by copying page contents
16977                                  <1>       ; to newly allocated page with write permission;
16978                                  <1>       ; sys_fork -> sys_exec -> copy on write, demand paging method is
16979                                  <1>       ; used for working with minimum possible memory usage.
16980                                  <1>       ; sys_fork will duplicate page directory and tables of parent
16981                                  <1>       ; process with 'read only' flag. If the child process attempts to
16982                                  <1>       ; write on these read only pages, page fault will be directed here
16983                                  <1>       ; for allocating a new page with same data/content.
16984                                  <1>       ;
16985                                  <1>       ; IMPORTANT : Retro UNIX 386 v1 (and SINGLIX and TR-DOS)
16986                                  <1>       ; will not force to separate CODE and DATA space
16987                                  <1>       ; in a process/program...
16988                                  <1>       ; CODE segment/section may contain DATA!
16989                                  <1>       ; It is flat, smoth and simplest programming method already as in
16990                                  <1>       ; Retro UNIX 8086 v1 and MS-DOS programs.
16991                                  <1>       ;
16992 00004E6B F6C202                  <1>       test   dl, 2  ; page fault was caused by a page write
16993                                  <1>                     ; sign
16994 00004E6E 0F84AB000000            <1>         jz      pfh_p_err
16995                                  <1>       ; 31/08/2015
16996 00004E74 F6C204                  <1>       test   dl, 4  ; page fault was caused while CPL = 3 (user mode)
16997                                  <1>                     ; sign.  (U+W+P = 4+2+1 = 7)
16998 00004E77 0F84A2000000            <1>         jz   pfh_pv_err
16999                                  <1>       ;
17000                                  <1>       ; make a new page and copy the parent's page content
17001                                  <1>       ; as the child's new page content
17002                                  <1>       ;
17003 00004E7D 0F20D3                  <1>       mov    ebx, cr2 ; CR2 contains the linear address
17004                                  <1>                       ; which has caused to page fault
17005 00004E80 E8A2000000              <1>       call   copy_page
17006 00004E85 0F828D000000            <1>         jc      pfh_im_err ; insufficient memory
17007                                  <1>       ;
17008 00004E8B EB7D                    <1>         jmp     pfh_cpp_ok
17009                                  <1>       ;
17010                                  <1> pfh_alloc_np:
17011 00004E8D E8E7FCFFFF              <1>       call   allocate_page; (allocate a new page)
17012 00004E92 0F8280000000            <1>         jc      pfh_im_err      ; 'insufficient memory' error
17013                                  <1> pfh_chk_cpl:
17014                                  <1>       ; EAX = Physical (base) address of the allocated (new) page
17015                                  <1>             ; (Lower 12 bits are ZERO, because
17016                                  <1>             ;     the address is on a page boundary)
17017 00004E98 80E204                  <1>       and    dl, 4  ; CPL = 3 ?
17018 00004E9B 7505                    <1>       jnz    short pfh_um
17019                                  <1>                     ; Page fault handler for kernel/system mode (CPL=0)
17020 00004E9D 0F20DB                  <1>       mov    ebx, cr3 ; CR3 (Control Register 3) contains physical address
17021                                  <1>                       ; of the current/active page directory
17022                                  <1>                       ; (Always kernel/system mode page directory, here!)
17023                                  <1>                       ; Note: Lower 12 bits are 0. (page boundary)
17024 00004EA0 EB06                    <1>       jmp    short pfh_get_pde
17025                                  <1>       ;
17026                                  <1> pfh_um:                   ; Page fault handler for user/appl. mode (CPL=3)
17027 00004EA2 8B1D[B8030600]          <1>       mov    ebx, [u.pgdir] ; Page directory of current/active process
17028                                  <1>                     ; Physical address of the USER's page directory
17029                                  <1>                     ; Note: Lower 12 bits are 0. (page boundary)
17030                                  <1> pfh_get_pde:
17031 00004EA8 80CA03                  <1>       or     dl, 3  ; USER + WRITE + PRESENT or SYSTEM + WRITE + PRESENT
```

```
17032 00004EAB 0F20D1              <1>       mov    ecx, cr2 ; CR2 contains the virtual address
17033                              <1>                       ; which has been caused to page fault
17034                              <1>                       ;
17035 00004EAE C1E914              <1>       shr    ecx, 20       ; shift 20 bits right
17036 00004EB1 80E1FC              <1>       and    cl, 0FCh ; mask lower 2 bits to get PDE offset
17037                              <1>       ;
17038 00004EB4 01CB                <1>       add    ebx, ecx ; now, EBX points to the relevant page dir entry
17039 00004EB6 8B0B                <1>       mov    ecx, [ebx] ; physical (base) address of the page table
17040 00004EB8 F6C101              <1>       test   cl, 1   ; check bit 0 is set (1) or not (0).
17041 00004EBB 740B                <1>       jz     short pfh_set_pde ; Page directory entry is not valid,
17042                              <1>                                ; set/validate page directory entry
17043 00004EBD 6681E100F0          <1>       and    cx, PDE_A_CLEAR ; 0F000h ; Clear attribute bits
17044 00004EC2 89CB                <1>       mov    ebx, ecx ; Physical address of the page table
17045 00004EC4 89C1                <1>       mov    ecx, eax ; new page address (physical)
17046 00004EC6 EB16                <1>       jmp    short pfh_get_pte
17047                              <1> pfh_set_pde:
17048                              <1>       ;; NOTE: Page directories and page tables never be swapped out!
17049                              <1>       ;;     (So, we know this PDE is empty or invalid)
17050                              <1>       ;
17051 00004EC8 08D0                <1>       or     al, dl  ; lower 3 bits are used as U/S, R/W, P flags
17052 00004ECA 8903                <1>       mov    [ebx], eax ; Let's put the new page directory entry here !
17053 00004ECC 30C0                <1>       xor    al, al  ; clear lower (3..8) bits
17054 00004ECE 89C3                <1>       mov    ebx, eax
17055 00004ED0 E8A4FCFFFF          <1>       call   allocate_page ; (allocate a new page)
17056 00004ED5 7241                <1>       jc     short pfh_im_err   ; 'insufficient memory' error
17057                              <1> pfh_spde_1:
17058                              <1>       ; EAX = Physical (base) address of the allocated (new) page
17059 00004ED7 89C1                <1>       mov    ecx, eax
17060 00004ED9 E815FDFFFF          <1>       call   clear_page ; Clear page content
17061                              <1> pfh_get_pte:
17062 00004EDE 0F20D0              <1>       mov    eax, cr2 ; virtual address
17063                              <1>                       ; which has been caused to page fault
17064 00004EE1 89C7                <1>       mov    edi, eax ; 20/07/2015
17065 00004EE3 C1E80C              <1>       shr    eax, 12       ; shift 12 bit right to get
17066                              <1>                          ; higher 20 bits of the page fault address
17067 00004EE6 25FF030000          <1>       and    eax, 3FFh ; mask PDE# bits, the result is PTE# (0 to 1023)
17068 00004EEB C1E002              <1>       shl    eax, 2 ; shift 2 bits left to get PTE offset
17069 00004EEE 01C3                <1>       add    ebx, eax ; now, EBX points to the relevant page table entry
17070 00004EF0 8B03                <1>       mov    eax, [ebx] ; get previous value of pte
17071                              <1>              ; bit 0 of EAX is always 0 (otherwise we would not be here)
17072 00004EF2 21C0                <1>       and    eax, eax
17073 00004EF4 7410                <1>       jz     short pfh_gpte_1
17074                              <1>       ; 20/07/2015
17075 00004EF6 87D9                <1>       xchg   ebx, ecx ; new page address (physical)
17076 00004EF8 55                  <1>       push   ebp ; 20/07/2015
17077 00004EF9 0F20D5              <1>       mov    ebp, cr2
17078                              <1>              ; ECX = physical address of the page table entry
17079                              <1>              ; EBX = Memory page address (physical!)
17080                              <1>              ; EAX = Swap disk (offset) address
17081                              <1>              ; EBP = virtual address (page fault address)
17082 00004EFC E8B7000000          <1>       call   swap_in
17083 00004F01 5D                  <1>       pop    ebp
17084 00004F02 7210                <1>       jc      short pfh_err_retn
17085 00004F04 87CB                <1>       xchg   ecx, ebx
17086                              <1>              ; EBX = physical address of the page table entry
17087                              <1>              ; ECX = new page
17088                              <1> pfh_gpte_1:
17089 00004F06 08D1                <1>       or     cl, dl ; lower 3 bits are used as U/S, R/W, P flags
17090 00004F08 890B                <1>       mov    [ebx], ecx ; Let's put the new page table entry here !
17091                              <1> pfh_cpp_ok:
17092                              <1>       ; 20/07/2015
17093 00004F0A 0F20D3              <1>       mov    ebx, cr2
17094 00004F0D E8A6020000          <1>       call   add_to_swap_queue
17095                              <1>       ;
17096                              <1>       ; The new PTE (which contains the new page) will be added to
17097                              <1>       ; the swap queue, here.
17098                              <1>       ; (Later, if memory will become insufficient,
17099                              <1>       ; one page will be swapped out which is at the head of
17100                              <1>       ; the swap queue by using FIFO and access check methods.)
17101                              <1>       ;
17102 00004F12 31C0                <1>       xor    eax, eax  ; 0
17103                              <1>       ;
17104                              <1> pfh_err_retn:
17105 00004F14 59                  <1>       pop    ecx
17106 00004F15 5A                  <1>       pop    edx
17107 00004F16 5B                  <1>       pop    ebx
17108 00004F17 C3                  <1>       retn
17109                              <1>
17110                              <1> pfh_im_err:
17111 00004F18 B8E4000000          <1>       mov    eax, ERR_MAJOR_PF + ERR_MINOR_IM ; Error code in AX
17112                              <1>                     ; Major (Primary) Error: Page Fault
17113                              <1>                     ; Minor (Secondary) Error: Insufficient Memory !
17114 00004F1D EBF5                <1>       jmp    short pfh_err_retn
17115                              <1>
17116                              <1>
17117                              <1> pfh_p_err: ; 09/03/2015
17118                              <1> pfh_pv_err:
17119                              <1>       ; Page fault was caused by a protection-violation
17120 00004F1F B8E6000000          <1>       mov    eax, ERR_MAJOR_PF + ERR_MINOR_PV ; Error code in AX
17121                              <1>                     ; Major (Primary) Error: Page Fault
17122                              <1>                     ; Minor (Secondary) Error: Protection violation !
17123 00004F24 F9                  <1>       stc
17124 00004F25 EBED                <1>       jmp    short pfh_err_retn
17125                              <1>
17126                              <1> copy_page:
17127                              <1>       ; 22/09/2015
17128                              <1>       ; 21/09/2015
17129                              <1>       ; 19/09/2015
17130                              <1>       ; 07/09/2015
17131                              <1>       ; 31/08/2015
17132                              <1>       ; 20/07/2015
17133                              <1>       ; 05/05/2015
```

```
17134                                   <1>       ; 03/05/2015
17135                                   <1>       ; 18/04/2015
17136                                   <1>       ; 12/04/2015
17137                                   <1>       ; 30/10/2014
17138                                   <1>       ; 18/10/2014 (Retro UNIX 386 v1 - beginning)
17139                                   <1>       ;
17140                                   <1>       ; INPUT ->
17141                                   <1>       ;     EBX = Virtual (linear) address of source page
17142                                   <1>       ;          (Page fault address)
17143                                   <1>       ; OUTPUT ->
17144                                   <1>       ;     EAX = PHYSICAL (real/flat) ADDRESS OF THE ALLOCATED PAGE
17145                                   <1>       ;     (corresponding PAGE TABLE ENTRY is mapped/set)
17146                                   <1>       ;     EAX = 0 (CF = 1)
17147                                   <1>       ;           if there is not a free page to be allocated
17148                                   <1>       ;     (page content of the source page will be copied
17149                                   <1>       ;     onto the target/new page)
17150                                   <1>       ;
17151                                   <1>       ; Modified Registers -> ecx, ebx (except EAX)
17152                                   <1>       ;
17153 00004F27 56                      <1>       push   esi
17154 00004F28 57                      <1>       push   edi
17155                                   <1>       ;push  ebx
17156                                   <1>       ;push  ecx
17157 00004F29 31F6                    <1>       xor    esi, esi
17158 00004F2B C1EB0C                  <1>       shr    ebx, 12 ; shift 12 bits right to get PDE & PTE numbers
17159 00004F2E 89D9                    <1>       mov    ecx, ebx ; save page fault address (as 12 bit shifted)
17160 00004F30 C1EB08                  <1>       shr    ebx, 8  ; shift 8 bits right and then
17161 00004F33 80E3FC                  <1>       and    bl, 0FCh ; mask lower 2 bits to get PDE offset
17162 00004F36 89DF                    <1>       mov    edi, ebx ; save it for the parent of current process
17163 00004F38 031D[B8030600]          <1>       add    ebx, [u.pgdir] ; EBX points to the relevant page dir entry
17164 00004F3E 8B03                    <1>       mov    eax, [ebx] ; physical (base) address of the page table
17165 00004F40 662500F0                <1>       and    ax, PTE_A_CLEAR ; 0F000h ; clear attribute bits
17166 00004F44 89CB                    <1>       mov    ebx, ecx   ; (restore higher 20 bits of page fault address)
17167 00004F46 81E3FF030000            <1>       and    ebx, 3FFh  ; mask PDE# bits, the result is PTE# (0 to 1023)
17168 00004F4C 66C1E302                <1>       shl    bx, 2     ; shift 2 bits left to get PTE offset
17169 00004F50 01C3                    <1>       add    ebx, eax   ; EBX points to the relevant page table entry
17170                                   <1>       ; 07/09/2015
17171 00004F52 66F7030002              <1>         test    word [ebx], PTE_DUPLICATED ; (Does current process share this
17172                                   <1>                                    ; read only page as a child process?)
17173 00004F57 7509                    <1>       jnz    short cpp_0 ; yes
17174 00004F59 8B0B                    <1>       mov    ecx, [ebx] ; PTE value
17175 00004F5B 6681E100F0              <1>       and    cx, PTE_A_CLEAR ; 0F000h  ; clear page attributes
17176 00004F60 EB32                    <1>       jmp    short cpp_1
17177                                   <1> cpp_0:
17178 00004F62 89FE                    <1>       mov    esi, edi
17179 00004F64 0335[BC030600]          <1>       add    esi, [u.ppgdir] ; the parent's page directory entry
17180 00004F6A 8B06                    <1>       mov    eax, [esi] ; physical (base) address of the page table
17181 00004F6C 662500F0                <1>       and    ax, PTE_A_CLEAR ; 0F000h ; clear attribute bits
17182 00004F70 89CE                    <1>       mov    esi, ecx   ; (restore higher 20 bits of page fault address)
17183 00004F72 81E6FF030000            <1>       and    esi, 3FFh  ; mask PDE# bits, the result is PTE# (0 to 1023)
17184 00004F78 66C1E602                <1>       shl    si, 2     ; shift 2 bits left to get PTE offset
17185 00004F7C 01C6                    <1>       add    esi, eax   ; EDX points to the relevant page table entry
17186 00004F7E 8B0E                    <1>       mov    ecx, [esi] ; PTE value of the parent process
17187                                   <1>       ; 21/09/2015
17188 00004F80 8B03                    <1>       mov    eax, [ebx] ; PTE value of the child process
17189 00004F82 662500F0                <1>       and    ax, PTE_A_CLEAR ; 0F000h ; clear page attributes
17190                                   <1>       ;
17191 00004F86 F6C101                  <1>       test   cl, PTE_A_PRESENT ; is it a present/valid page ?
17192 00004F89 7424                    <1>       jz     short cpp_3 ; the parent's page is not same page
17193                                   <1>       ;
17194 00004F8B 6681E100F0              <1>       and    cx, PTE_A_CLEAR ; 0F000h ; clear page attributes
17195 00004F90 39C8                    <1>       cmp    eax, ecx   ; Same page?
17196 00004F92 751B                    <1>       jne    short cpp_3 ; Parent page and child page are not same
17197                                   <1>                          ; Convert child's page to writable page
17198                                   <1> cpp_1:
17199 00004F94 E8E0FBFFFF              <1>       call   allocate_page
17200 00004F99 721A                    <1>       jc     short cpp_4 ; 'insufficient memory' error
17201 00004F9B 21F6                    <1>       and    esi, esi    ; check ESI is valid or not
17202 00004F9D 7405                    <1>       jz     short cpp_2
17203                                   <1>              ; Convert read only page to writable page
17204                                   <1>              ;(for the parent of the current process)
17205                                   <1>       ;and   word [esi], PTE_A_CLEAR ; 0F000h
17206                                   <1>       ; 22/09/2015
17207 00004F9F 890E                    <1>       mov    [esi], ecx
17208 00004FA1 800E07                  <1>       or     byte [esi], PTE_A_PRESENT + PTE_A_WRITE + PTE_A_USER
17209                                   <1>                                ; 1+2+4 = 7
17210                                   <1> cpp_2:
17211 00004FA4 89C7                    <1>       mov    edi, eax ; new page address of the child process
17212                                   <1>       ; 07/09/2015
17213 00004FA6 89CE                    <1>       mov    esi, ecx ; the page address of the parent process
17214 00004FA8 B900040000              <1>       mov    ecx, PAGE_SIZE / 4
17215 00004FAD F3A5                    <1>       rep    movsd ; 31/08/2015
17216                                   <1> cpp_3:
17217 00004FAF 0C07                    <1>       or     al, PTE_A_PRESENT + PTE_A_WRITE + PTE_A_USER ; 1+2+4 = 7
17218 00004FB1 8903                    <1>       mov    [ebx], eax ; Update PTE
17219 00004FB3 28C0                    <1>       sub    al, al ; clear attributes
17220                                   <1> cpp_4:
17221                                   <1>       ;pop   ecx
17222                                   <1>       ;pop   ebx
17223 00004FB5 5F                      <1>       pop    edi
17224 00004FB6 5E                      <1>       pop    esi
17225 00004FB7 C3                      <1>       retn
17226                                   <1>
17227                                   <1> ;; 28/04/2015
17228                                   <1> ;; 24/10/2014
17229                                   <1> ;; 21/10/2014 (Retro UNIX 386 v1 - beginning)
17230                                   <1> ;; SWAP_PAGE_QUEUE (4096 bytes)
17231                                   <1> ;;
17232                                   <1> ;;   0000   0001   0002   0003   ....   1020   1021   1022   1023
17233                                   <1> ;; +------+------+------+------+-    -+------+------+------+------+
17234                                   <1> ;; |  pg1 |  pg2 |  pg3 |  pg4 | .... |pg1021|pg1022|pg1023|pg1024|
17235                                   <1> ;; +------+------+------+------+-    -+------+------+------+------+
```

```
17236                           <1> ;;
17237                           <1> ;; [swpq_last] = 0 to 4096 (step 4) -> the last position on the queue
17238                           <1> ;;
17239                           <1> ;; Method:
17240                           <1> ;;    Swap page queue is a list of allocated pages with physical
17241                           <1> ;;    addresses (system mode virtual adresses = physical addresses).
17242                           <1> ;;    It is used for 'swap_in' and 'swap_out' procedures.
17243                           <1> ;;    When a new page is being allocated, swap queue is updated
17244                           <1> ;;    by 'swap_queue_shift' procedure, header of the queue (offset 0)
17245                           <1> ;;    is checked for 'accessed' flag. If the 1st page on the queue
17246                           <1> ;;    is 'accessed' or 'read only', it is dropped from the list;
17247                           <1> ;;    other pages from the 2nd to the last (in [swpq_last]) shifted
17248                           <1> ;;    to head then the 2nd page becomes the 1st and '[swpq_last]'
17249                           <1> ;;    offset value becomes it's previous offset value - 4.
17250                           <1> ;;    If the 1st page of the swap page queue is not 'accessed'
17251                           <1> ;;    the queue/list is not shifted.
17252                           <1> ;;    After the queue/list shift, newly allocated page is added
17253                           <1> ;;    to the tail of the queue at the [swpq_count*4] position.
17254                           <1> ;;    But, if [swpq_count] > 1023, the newly allocated page
17255                           <1> ;;    will not be added to the tail of swap page queue.
17256                           <1> ;;
17257                           <1> ;;    During 'swap_out' procedure, swap page queue is checked for
17258                           <1> ;;    the first non-accessed, writable page in the list,
17259                           <1> ;;    from the head to the tail. The list is shifted to left
17260                           <1> ;;    (to the head) till a non-accessed page will be found in the list.
17261                           <1> ;;    Then, this page    is swapped out (to disk) and then it is dropped
17262                           <1> ;;    from the list by a final swap queue shift. [swpq_count] value
17263                           <1> ;;    is changed. If all pages on the queue' are 'accessed',
17264                           <1> ;;    'insufficient memory' error will be returned ('swap_out'
17265                           <1> ;;    procedure will be failed)...
17266                           <1> ;;
17267                           <1> ;;    Note: If the 1st page of the queue is an 'accessed' page,
17268                           <1> ;;    'accessed' flag of the page will be reset (0) and that page
17269                           <1> ;;    (PTE) will be added to the tail of the queue after
17270                           <1> ;;    the check, if [swpq_count] < 1023. If [swpq_count] = 1024
17271                           <1> ;;    the queue will be rotated and the PTE in the head will be
17272                           <1> ;;    added to the tail after resetting 'accessed' bit.
17273                           <1> ;;
17274                           <1> ;;
17275                           <1> ;;
17276                           <1> ;; SWAP DISK/FILE (with 4096 bytes swapped page blocks)
17277                           <1> ;;
17278                           <1> ;;  00000000  00000004  00000008  0000000C   ...   size-8    size-4
17279                           <1> ;; +---------+---------+---------+---------+-- --+---------+---------+
17280                           <1> ;; |descriptr| page(1) | page(2) | page(3) | ... |page(n-1)| page(n) |
17281                           <1> ;; +---------+---------+---------+---------+-- --+---------+---------+
17282                           <1> ;;
17283                           <1> ;; [swpd_next] = the first free block address in swapped page records
17284                           <1> ;;                  for next free block search by 'swap_out' procedure.
17285                           <1> ;; [swpd_size] = swap disk/file size in sectors (512 bytes)
17286                           <1> ;;           NOTE: max. possible swap disk size is 1024 GB
17287                           <1> ;;           (entire swap space must be accessed by using
17288                           <1> ;;           31 bit offset address)
17289                           <1> ;; [swpd_free] = free block (4096 bytes) count in swap disk/file space
17290                           <1> ;; [swpd_start] = absolute/start address of the swap disk/file
17291                           <1> ;;            0 for file, or beginning sector of the swap partition
17292                           <1> ;; [swp_drv] = logical drive description table addr. of swap disk/file
17293                           <1> ;;
17294                           <1> ;;
17295                           <1> ;; Method:
17296                           <1> ;;    When the memory (ram) becomes insufficient, page allocation
17297                           <1> ;;    procedure swaps out a page from memory to the swap disk
17298                           <1> ;;    (partition) or swap file to get a new free page at the memory.
17299                           <1> ;;    Swapping out is performed by using swap page queue.
17300                           <1> ;;
17301                           <1> ;;    Allocation block size of swap disk/file is equal to page size
17302                           <1> ;;    (4096 bytes). Swapping address (in sectors) is recorded
17303                           <1> ;;    into relevant page file entry as 31 bit physical (logical)
17304                           <1> ;;    offset address as 1 bit shifted to left for present flag (0).
17305                           <1> ;;    Swapped page address is between 1 and swap disk/file size - 4.
17306                           <1> ;;    Absolute physical (logical) address of the swapped page is
17307                           <1> ;;    calculated by adding offset value to the swap partition's
17308                           <1> ;;    start address. If the swap device (disk) is a virtual disk
17309                           <1> ;;    or it is a file, start address of the swap disk/volume is 0,
17310                           <1> ;;    and offset value is equal to absolute (physical or logical)
17311                           <1> ;;    address/position. (It has not to be ZERO if the swap partition
17312                           <1> ;;    is in a partitioned virtual hard disk.)
17313                           <1> ;;
17314                           <1> ;;    Note: Swap addresses are always specified/declared in sectors,
17315                           <1> ;;    not in bytes or     in blocks/zones/clusters (4096 bytes) as unit.
17316                           <1> ;;
17317                           <1> ;;    Swap disk/file allocation is mapped via 'Swap Allocation Table'
17318                           <1> ;;    at memory as similar to 'Memory Allocation Table'.
17319                           <1> ;;
17320                           <1> ;;    Every bit of Swap Allocation Table repsesents one swap block
17321                           <1> ;;    (equal to page size) respectively. Bit 0 of the S.A.T. byte 0
17322                           <1> ;;    is reserved for swap disk/file block 0 as descriptor block
17323                           <1> ;;    (also for compatibility with PTE). If bit value is ZERO,
17324                           <1> ;;    it means relevant (respective) block is in use, and,
17325                           <1> ;;    of course, if bit value is 1, it means relevant (respective)
17326                           <1> ;;      swap disk/file block is free.
17327                           <1> ;;    For example: bit 1 of the byte 128 repsesents block 1025
17328                           <1> ;;    (128*8+1) or sector (offset) 8200 on the swap disk or
17329                           <1> ;;    byte (offset/position) 4198400 in the swap file.
17330                           <1> ;;    4GB swap space is represented via 128KB Swap Allocation Table.
17331                           <1> ;;    Initial layout of Swap Allocation Table is as follows:
17332                           <1> ;;    ------------------------------------------------------------
17333                           <1> ;;    01111111111111111111111 .... 11111111111111111111111111111
17334                           <1> ;;    ------------------------------------------------------------
17335                           <1> ;;    (0 is reserved block, 1s represent free blocks respectively.)
17336                           <1> ;;    (Note: Allocation cell/unit of the table is bit, not byte)
17337                           <1> ;;
```

```
17338                                  <1> ;;    ..........................................................
17339                                  <1> ;;
17340                                  <1> ;;    'swap_out' procedure checks 'free_swap_blocks' count at first,
17341                                  <1> ;;    then it searches Swap Allocation Table if free count is not
17342                                  <1> ;;    zero. From begining the [swpd_next] dword value, the first bit
17343                                  <1> ;;    position with value of 1 on the table is converted to swap
17344                                  <1> ;;    disk/file offset address, in sectors (not 4096 bytes block).
17345                                  <1> ;;    'ldrv_write' procedure is called with ldrv (logical drive
17346                                  <1> ;;    number of physical swap disk or virtual swap disk)
17347                                  <1> ;;    number, sector offset (not absolute sector -LBA- number),
17348                                  <1> ;;    and sector count (8, 512*8 = 4096) and buffer adress
17349                                  <1> ;;    (memory page). That will be a direct disk write procedure.
17350                                  <1> ;;    (for preventing late memory allocation, significant waiting).
17351                                  <1> ;;    If disk write procedure returns with error or free count of
17352                                  <1> ;;    swap blocks is ZERO, 'swap_out' procedure will return with
17353                                  <1> ;;    'insufficient memory error' (cf=1).
17354                                  <1> ;;
17355                                  <1> ;;    (Note: Even if free swap disk/file blocks was not zero,
17356                                  <1> ;;    any disk write error will not be fixed by 'swap_out' procedure,
17357                                  <1> ;;    in other words, 'swap_out' will not check the table for other
17358                                  <1> ;;    free blocks after a disk write error. It will return to
17359                                  <1> ;;    the caller with error (CF=1) which means swapping is failed.
17360                                  <1> ;;
17361                                  <1> ;;    After writing the page on to swap disk/file address/sector,
17362                                  <1> ;;    'swap_out' procesure returns with that swap (offset) sector
17363                                  <1> ;;    address (cf=0).
17364                                  <1> ;;
17365                                  <1> ;;    ..........................................................
17366                                  <1> ;;
17367                                  <1> ;;    'swap_in' procedure loads addressed (relevant) swap disk or
17368                                  <1> ;;    file sectors at specified memory page. Then page allocation
17369                                  <1> ;;    procedure updates relevant page table entry with 'present'
17370                                  <1> ;;    attribute. If swap disk or file reading fails there is nothing
17371                                  <1> ;;    to do, except to terminate the process which is the owner of
17372                                  <1> ;;    the swapped page.
17373                                  <1> ;;
17374                                  <1> ;;    'swap_in' procedure sets the relevant/respective bit value
17375                                  <1> ;;    in the Swap Allocation Table (as free block). 'swap_in' also
17376                                  <1> ;;    updates [swpd_first] pointer if it is required.
17377                                  <1> ;;
17378                                  <1> ;;    ..........................................................
17379                                  <1> ;;
17380                                  <1> ;;    Note: If [swap_enabled] value is ZERO, that means there is not
17381                                  <1> ;;    a swap disk or swap file in use... 'swap_in' and 'swap_out'
17382                                  <1> ;;    procedures ans 'swap page que' procedures will not be active...
17383                                  <1> ;;    'Insufficient memory' error will be returned by 'swap_out'
17384                                  <1> ;;    and 'general protection fault' will be returned by 'swap_in'
17385                                  <1> ;;    procedure, if it is called mistakenly (a wrong value in a PTE).
17386                                  <1> ;;
17387                                  <1>
17388                                  <1> swap_in:
17389                                  <1>       ; 31/08/2015
17390                                  <1>       ; 20/07/2015
17391                                  <1>       ; 28/04/2015
17392                                  <1>       ; 18/04/2015
17393                                  <1>       ; 24/10/2014 (Retro UNIX 386 v1 - beginning)
17394                                  <1>       ;
17395                                  <1>       ; INPUT ->
17396                                  <1>       ;     EBX = PHYSICAL (real/flat) ADDRESS OF THE MEMORY PAGE
17397                                  <1>       ;     EBP = VIRTUAL (LINEAR) ADDRESS (page fault address)
17398                                  <1>       ;     EAX = Offset Address for the swapped page on the
17399                                  <1>       ;           swap disk or in the swap file.
17400                                  <1>       ;
17401                                  <1>       ; OUTPUT ->
17402                                  <1>       ;     EAX = 0 if loading at memory has been successful
17403                                  <1>       ;
17404                                  <1>       ;     CF = 1 -> swap disk reading error (disk/file not present
17405                                  <1>       ;              or sector not present or drive not ready
17406                                  <1>       ;          EAX = Error code
17407                                  <1>       ;          [u.error] = EAX
17408                                  <1>       ;                  = The last error code for the process
17409                                  <1>       ;                    (will be reset after returning to user)
17410                                  <1>       ;
17411                                  <1>       ; Modified Registers -> EAX
17412                                  <1>       ;
17413                                  <1>
17414 00004FB8 833D[62050600]00       <1>         cmp     dword [swp_drv], 0
17415 00004FBF 7646                   <1>       jna   short swpin_dnp_err
17416                                  <1>
17417 00004FC1 3B05[66050600]         <1>       cmp   eax, [swpd_size]
17418 00004FC7 734A                   <1>       jnb   short swpin_snp_err
17419                                  <1>
17420 00004FC9 56                     <1>       push  esi
17421 00004FCA 53                     <1>       push  ebx
17422 00004FCB 51                     <1>       push  ecx
17423 00004FCC 8B35[62050600]         <1>       mov   esi, [swp_drv]
17424 00004FD2 B908000000             <1>       mov   ecx, PAGE_SIZE / LOGIC_SECT_SIZE  ; 8 !
17425                                  <1>             ; Note: Even if corresponding physical disk's sector
17426                                  <1>             ; size different than 512 bytes, logical disk sector
17427                                  <1>             ; size is 512 bytes and disk reading procedure
17428                                  <1>             ; will be performed for reading 4096 bytes
17429                                  <1>             ; (2*2048, 8*512).
17430                                  <1>       ; ESI = Logical disk description table address
17431                                  <1>       ; EBX = Memory page (buffer) address (physical!)
17432                                  <1>       ; EAX = Sector adress (offset address, logical sector number)
17433                                  <1>       ; ECX = Sector count ; 8 sectors
17434 00004FD7 50                     <1>       push  eax
17435 00004FD8 E8AF020000             <1>       call  logical_disk_read
17436 00004FDD 58                     <1>       pop   eax
17437 00004FDE 730C                   <1>       jnc   short swpin_read_ok
17438                                  <1>       ;
17439 00004FE0 B828000000             <1>       mov   eax, SWP_DISK_READ_ERR ; drive not ready or read error
```

```
17440 00004FE5 A3[C8030600]        <1>       mov    [u.error], eax
17441 00004FEA EB17                <1>       jmp    short swpin_retn
17442                              <1>       ;
17443                              <1> swpin_read_ok:
17444                              <1>       ; EAX = Offset address (logical sector number)
17445 00004FEC E80D020000          <1>       call   unlink_swap_block  ; Deallocate swap block
17446                              <1>       ;
17447                              <1>       ; EBX = Memory page (buffer) address (physical!)
17448                              <1>       ; 20/07/2015
17449 00004FF1 89EB                <1>       mov    ebx, ebp ; virtual address (page fault address)
17450 00004FF3 6681E300F0          <1>         and     bx, ~PAGE_OFF ; ~0FFFh ; reset bits, 0 to 11
17451 00004FF8 8A1D[B3030600]      <1>       mov    bl, [u.uno] ; current process number
17452                              <1>       ; EBX = Virtual (Linear) address & process number combination
17453 00004FFE E8DB000000          <1>       call   swap_queue_shift
17454                              <1>       ; eax = 0 ; 10/06/2016 (if ebx input > 0, eax output = 0)
17455                              <1>       ;sub   eax, eax  ; 0 ; Error Code = 0  (no error)
17456                              <1>       ; zf = 1
17457                              <1> swpin_retn:
17458 00005003 59                  <1>       pop    ecx
17459 00005004 5B                  <1>       pop    ebx
17460 00005005 5E                  <1>       pop    esi
17461 00005006 C3                  <1>       retn
17462                              <1>
17463                              <1> swpin_dnp_err:
17464 00005007 B829000000          <1>       mov    eax, SWP_DISK_NOT_PRESENT_ERR
17465                              <1> swpin_err_retn:
17466 0000500C A3[C8030600]        <1>       mov    [u.error], eax
17467 00005011 F9                  <1>       stc
17468 00005012 C3                  <1>       retn
17469                              <1>
17470                              <1> swpin_snp_err:
17471 00005013 B82A000000          <1>       mov    eax, SWP_SECTOR_NOT_PRESENT_ERR
17472 00005018 EBF2                <1>       jmp    short swpin_err_retn
17473                              <1>
17474                              <1> swap_out:
17475                              <1>       ; 10/06/2016
17476                              <1>       ; 07/06/2016
17477                              <1>         ; 23/05/2016
17478                              <1>       ; 19/05/2016 - TRDOS 386 (TRDOS v2.0)
17479                              <1>       ; 24/10/2014 - 31/08/2015 (Retro UNIX 386 v1)
17480                              <1>       ;
17481                              <1>       ; INPUT ->
17482                              <1>       ;      none
17483                              <1>       ;
17484                              <1>       ; OUTPUT ->
17485                              <1>       ;      EAX = Physical page address (which is swapped out
17486                              <1>       ;            for allocating a new page)
17487                              <1>       ;      CF = 1 -> swap disk writing error (disk/file not present
17488                              <1>       ;               or sector not present or drive not ready
17489                              <1>       ;           EAX = Error code
17490                              <1>       ;           [u.error] = EAX
17491                              <1>       ;                   = The last error code for the process
17492                              <1>       ;                     (will be reset after returning to user)
17493                              <1>       ;
17494                              <1>       ; Modified Registers -> none (except EAX)
17495                              <1>       ;
17496 0000501A 66833D[60050600]01  <1>       cmp    word [swpq_count], 1
17497 00005022 0F82AF000000        <1>         jc      swpout_im_err ; 'insufficient memory'
17498                              <1>
17499                              <1>         ;cmp    dword [swp_drv], 1
17500                              <1>       ;jc    short swpout_dnp_err ; 'swap disk/file not present'
17501                              <1>
17502 00005028 833D[6A050600]01    <1>         cmp     dword [swpd_free], 1
17503 0000502F 0F828F000000        <1>         jc      swpout_nfspc_err ; 'no free space on swap disk'
17504                              <1>
17505 00005035 53                  <1>       push   ebx ; *
17506                              <1> swpout_1:
17507                              <1>       ; 10/06/2016
17508 00005036 31DB                <1>       xor    ebx, ebx ; shift the queue and return a PTE value
17509 00005038 E8A1000000          <1>       call   swap_queue_shift
17510 0000503D 21C0                <1>       and    eax, eax     ; 0 = empty queue (improper entries)
17511 0000503F 0F848A000000        <1>         jz      swpout_npts_err        ; There is not any proper PTE
17512                              <1>                                ; pointer in the swap queue
17513                              <1>       ; EAX = PTE value of the page
17514                              <1>       ; EBX = PTE address of the page
17515 00005045 662500F0            <1>       and    ax, PTE_A_CLEAR ; 0F000h ; clear attribute bits
17516                              <1>       ;
17517                              <1>       ; 07/06/2016
17518                              <1>       ; 19/05/2016
17519                              <1>       ; check this page is in timer events or not
17520                              <1>
17521                              <1> swpout_timer_page_0:
17522 00005049 52                  <1>       push   edx ; **
17523                              <1>
17524                              <1>       ; 07/06/2016
17525 0000504A 803D[5F5B0100]00    <1>       cmp    byte [timer_events], 0
17526 00005051 762F                <1>       jna    short swpout_2
17527                              <1>       ;
17528 00005053 8A15[5F5B0100]      <1>       mov    dl, [timer_events]
17529                              <1>
17530 00005059 51                  <1>       push   ecx ; ***
17531 0000505A 53                  <1>       push   ebx ; ****
17532 0000505B BB[60040600]        <1>       mov    ebx, timer_set ; beginning address of timer event
17533                              <1>                             ; structures
17534                              <1> swpout_timer_page_1:
17535 00005060 8A0B                <1>       mov    cl, [ebx]
17536 00005062 08C9                <1>       or     cl, cl ; 0 = free, >0 = process number
17537 00005064 7415                <1>       jz     short swpout_timer_page_3
17538 00005066 8B4B0C              <1>       mov    ecx, [ebx+12] ; response (signal return) address
17539 00005069 6681E100F0          <1>       and    cx, PTE_A_CLEAR ; clear offset part (right 12 bits)
17540                              <1>                              ; of the response byte address, to
17541                              <1>                              ; get beginning of the page address)
```

```
17542 0000506E 39C8              <1>       cmp    eax, ecx
17543 00005070 7505              <1>       jne    short swpout_timer_page_2 ; not same page
17544                            <1>
17545                            <1>       ; !same page!
17546                            <1>       ;
17547                            <1>       ; NOTE: // 19/05/2016 // - TRDOS 386 feature only ! -
17548                            <1>       ; This page will be used by the kernel to put timer event
17549                            <1>       ; response (signal return) byte at the requested address;
17550                            <1>       ; in order to prevent a possible wrong write (while
17551                            <1>       ; this page is swapped out) on physical memory,
17552                            <1>       ; we must protect this page against to be swapped out!
17553                            <1>       ;
17554 00005072 5B                <1>       pop    ebx ; ****
17555 00005073 59                <1>       pop    ecx ; ***
17556 00005074 5A                <1>       pop    edx ; **
17557 00005075 EBBF              <1>       jmp    short swpout_1      ; do not swap out this page !
17558                            <1>
17559                            <1> swpout_timer_page_2:
17560                            <1>       ; 07/06/2016
17561 00005077 FECA              <1>       dec    dl
17562 00005079 7405              <1>       jz     short swpout_timer_page_4
17563                            <1> swpout_timer_page_3:
17564                            <1>       ;cmp   ebx, timer_set + 240 ; last timer event (15*16)
17565                            <1>       ;jnb   short swpout_timer_page_4
17566 0000507B 83C310            <1>       add    ebx, 16
17567 0000507E EBE0              <1>       jmp    short swpout_timer_page_1
17568                            <1>
17569                            <1> swpout_timer_page_4:
17570 00005080 5B                <1>       pop    ebx ; ****
17571 00005081 59                <1>       pop    ecx ; ***
17572                            <1> swpout_2:
17573 00005082 89DA              <1>       mov    edx, ebx            ; Page table entry address
17574 00005084 89C3              <1>       mov    ebx, eax            ; Buffer (Page) Address
17575                            <1>       ;
17576 00005086 E8A6010000        <1>       call   link_swap_block
17577 0000508B 7304              <1>       jnc    short swpout_3             ; It may not be needed here
17578                            <1>                                   ; because [swpd_free] value
17579                            <1>                                   ; was checked at the beginging.
17580 0000508D 5A                <1>       pop    edx ; **
17581 0000508E 5B                <1>       pop    ebx ; *
17582 0000508F EB33              <1>       jmp    short swpout_nfspc_err
17583                            <1> swpout_3:
17584 00005091 A900000080        <1>       test   eax, 80000000h ; test bit 31 (this may not be needed!)
17585 00005096 752C              <1>       jnz    short swpout_nfspc_err  ; 10/06/2016 (bit 31 = 1 !)
17586                            <1>       ;
17587 00005098 56                <1>       push   esi ; **
17588 00005099 51                <1>       push   ecx ; ***
17589 0000509A 50                <1>       push   eax ; sector address ; (31 bit !, bit 31 = 0)
17590 0000509B 8B35[62050600]    <1>       mov    esi, [swp_drv]
17591 000050A1 B908000000        <1>       mov    ecx, PAGE_SIZE / LOGIC_SECT_SIZE  ; 8 !
17592                            <1>              ; Note: Even if corresponding physical disk's sector
17593                            <1>              ; size different than 512 bytes, logical disk sector
17594                            <1>              ; size is 512 bytes and disk writing procedure
17595                            <1>              ; will be performed for writing 4096 bytes
17596                            <1>              ; (2*2048, 8*512).
17597                            <1>       ; ESI = Logical disk description table address
17598                            <1>       ; EBX = Buffer (Page) address
17599                            <1>       ; EAX = Sector adress (offset address, logical sector number)
17600                            <1>       ; ECX = Sector count ; 8 sectors
17601                            <1>       ; edx = PTE address
17602 000050A6 E8E2010000        <1>       call   logical_disk_write
17603                            <1>       ; edx = PTE address
17604 000050AB 59                <1>       pop    ecx ; sector address
17605 000050AC 730C              <1>       jnc    short swpout_write_ok
17606                            <1>       ;
17607                            <1>       ;; call      unlink_swap_block ; this block must be left as 'in use'
17608                            <1> swpout_dw_err:
17609 000050AE B82C000000        <1>       mov    eax, SWP_DISK_WRITE_ERR ; drive not ready or write error
17610 000050B3 A3[C8030600]      <1>       mov    [u.error], eax
17611 000050B8 EB06              <1>       jmp    short swpout_retn
17612                            <1>       ;
17613                            <1> swpout_write_ok:
17614                            <1>       ; EBX = Buffer (page) address
17615                            <1>       ; EDX = Page Table Entry address
17616                            <1>       ; ECX = Swap disk sector (file block) address (31 bit)
17617 000050BA D1E1              <1>       shl    ecx, 1  ; 31 bit sector address from bit 1 to bit 31
17618 000050BC 890A              <1>       mov    [edx], ecx
17619                            <1>              ; bit 0 = 0 (swapped page)
17620 000050BE 89D8              <1>       mov    eax, ebx
17621                            <1> swpout_retn:
17622 000050C0 59                <1>       pop    ecx ; ***
17623 000050C1 5E                <1>       pop    esi ; **
17624 000050C2 5B                <1>       pop    ebx ; *
17625 000050C3 C3                <1>       retn
17626                            <1>
17627                            <1> ;swpout_dnp_err:
17628                            <1> ;     mov    eax, SWP_DISK_NOT_PRESENT_ERR ; disk not present
17629                            <1> ;     jmp    short swpout_err_retn
17630                            <1> swpout_nfspc_err:
17631 000050C4 B82B000000        <1>       mov    eax, SWP_NO_FREE_SPACE_ERR ; no free space
17632                            <1> swpout_err_retn:
17633 000050C9 A3[C8030600]      <1>       mov    [u.error], eax
17634                            <1>       ;stc
17635 000050CE C3                <1>       retn
17636                            <1> swpout_npts_err:
17637 000050CF B82D000000        <1>       mov    eax, SWP_NO_PAGE_TO_SWAP_ERR
17638 000050D4 5B                <1>       pop    ebx
17639 000050D5 EBF2              <1>       jmp    short swpout_err_retn
17640                            <1> swpout_im_err:
17641 000050D7 B804000000        <1>       mov    eax, ERR_MINOR_IM ; insufficient (out of) memory
17642 000050DC EBEB              <1>       jmp    short swpout_err_retn
17643                            <1>
```

```
                                <1> swap_queue_shift:
                                <1>       ; 26/03/2017
                                <1>       ; 10/06/2016
                                <1>       ; 09/06/2016 - TRDOS 386 (TRDOS v2.0)
                                <1>       ; 23/10/2014 - 20/07/2015 (Retro UNIX 386 v1)
                                <1>       ;
                                <1>       ; INPUT ->
                                <1>       ;     EBX = Virtual (linear) address (bit 12 to 31)
                                <1>       ;           and process number combination (bit 0 to 11)
                                <1>       ;     EBX = 0 -> shift/drop from the head (offset 0)
                                <1>       ;
                                <1>       ; OUTPUT ->
                                <1>       ;     If EBX input > 0
                                <1>       ;        the queue will be shifted 4 bytes (dword),
                                <1>       ;        from the tail to the head, up to entry offset
                                <1>       ;        which points to EBX input value or nothing
                                <1>       ;        to do if EBX value is not found on the queue.
                                <1>       ;        (The entry -with EBX value- will be removed
                                <1>       ;        from the queue if it is found.)
                                <1>       ;
                                <1>       ;        EAX = 0
                                <1>       ;
                                <1>       ;     If EBX input = 0
                                <1>       ;        the queue will be shifted 4 bytes (dword),
                                <1>       ;        from the tail to the head, if the PTE address
                                <1>       ;        which is pointed in head of the queue is marked
                                <1>       ;        as "accessed" or it is marked as "non present".
                                <1>       ;        (If "accessed" flag of the PTE -which is pointed
                                <1>       ;        in the head- is set -to 1-, it will be reset
                                <1>       ;        -to 0- and then, the queue will be rotated
                                <1>       ;        -without dropping pointer of the PTE from
                                <1>       ;        the queue- for 4 bytes on head to tail direction.
                                <1>       ;        Pointer in the head will be moved into the tail,
                                <1>       ;        other PTEs will be shifted on head direction.)
                                <1>       ;
                                <1>       ;        Swap queue will be shifted up to the first
                                <1>       ;        'present' or 'non accessed' page will be found
                                <1>       ;        (as pointed) on the queue head (then it will be
                                <1>       ;          removed/dropped from the queue).
                                <1>       ;
                                <1>       ;        EAX (> 0) = PTE value of the page which is
                                <1>       ;             (it's pointer -virtual address-) dropped
                                <1>       ;             (removed) from swap queue.
                                <1>       ;        EBX = PTE address of the page (if EAX > 0)
                                <1>       ;              which is (it's pointer -virtual address-)
                                <1>       ;             dropped (removed) from swap queue.
                                <1>       ;
                                <1>       ;        EAX = 0 -> empty swap queue !
                                <1>       ;
                                <1>       ; Modified Registers -> EAX, EBX
                                <1>       ;
000050DE 0FB705[60050600]       <1>       movzx   eax, word [swpq_count]  ; Max. 1024
000050E5 6621C0                 <1>       and    ax, ax
000050E8 7431                   <1>       jz     short swpqs_retn
000050EA 57                     <1>       push   edi
000050EB 56                     <1>       push   esi
000050EC 51                     <1>       push   ecx
000050ED BE00E00800             <1>       mov    esi, swap_queue
000050F2 89C1                   <1>       mov    ecx, eax
000050F4 09DB                   <1>       or     ebx, ebx
000050F6 7424                   <1>       jz     short swpqs_7
                                <1> swpqs_1:
000050F8 AD                     <1>       lodsd
000050F9 39D8                   <1>       cmp    eax, ebx
000050FB 7406                   <1>       je     short swpqs_2
000050FD E2F9                   <1>       loop   swpqs_1
                                <1>       ; 10/06/2016
000050FF 29C0                   <1>       sub    eax, eax
00005101 EB15                   <1>       jmp    short swpqs_6
                                <1> swpqs_2:
00005103 89F7                   <1>       mov    edi, esi
00005105 83EF04                 <1>       sub    edi, 4
                                <1> swpqs_3:
00005108 66FF0D[60050600]       <1>       dec    word [swpq_count]
0000510F 7403                   <1>       jz     short swpqs_5
                                <1> swpqs_4:
00005111 49                     <1>       dec    ecx
00005112 F3A5                   <1>       rep    movsd  ; shift up (to the head)
                                <1> swpqs_5:
00005114 31C0                   <1>       xor    eax, eax
00005116 8907                   <1>       mov    [edi], eax
                                <1> swpqs_6:
00005118 59                     <1>       pop    ecx
00005119 5E                     <1>       pop    esi
0000511A 5F                     <1>       pop    edi
                                <1> swpqs_retn:
0000511B C3                     <1>       retn
                                <1> swpqs_7:
0000511C 89F7                   <1>       mov    edi, esi ; head
0000511E AD                     <1>       lodsd
                                <1>       ; 20/07/2015
0000511F 89C3                   <1>       mov    ebx, eax
00005121 81E300F0FFFF           <1>       and    ebx, ~PAGE_OFF ; ~0FFFh
                                <1>                     ; ebx = virtual address (at page boundary)
00005127 25FF0F0000             <1>       and    eax, PAGE_OFF ; 0FFFh
                                <1>                     ; ax = process number (1 to 4095)
0000512C 3A05[B3030600]         <1>       cmp    al, [u.uno]
                                <1>             ; Max. 16 (nproc) processes for Retro UNIX 386 v1
00005132 7507                   <1>       jne    short swpqs_8
00005134 A1[B8030600]           <1>       mov    eax, [u.pgdir]
00005139 EB28                   <1>       jmp    short swpqs_9
                                <1> swpqs_8:
```

```
17746                                  <1>       ; 09/06/2016
17747 0000513B 80B8[AF000600]00        <1>       cmp    byte [eax+p.stat-1], 0
17748 00005142 76C4                    <1>       jna    short swpqs_3     ; free (or terminated) process
17749 00005144 80B8[AF000600]02        <1>       cmp    byte [eax+p.stat-1], 2 ; waiting
17750 0000514B 77BB                    <1>       ja     short swpqs_3         ; zombie (3) or undefined ?
17751                                  <1>
17752                                  <1>       ;shl   ax, 2
17753 0000514D C0E002                  <1>       shl    al, 2
17754 00005150 8B80[BC000600]          <1>       mov    eax, [eax+p.upage-4]
17755 00005156 09C0                    <1>       or     eax, eax
17756 00005158 74AE                    <1>       jz     short swpqs_3 ; invalid upage
17757 0000515A 83C05C                  <1>       add    eax, u.pgdir - user
17758                                  <1>                     ; u.pgdir value for the process
17759                                  <1>                     ; is in [eax]
17760 0000515D 8B00                    <1>       mov    eax, [eax]
17761 0000515F 21C0                    <1>       and    eax, eax
17762 00005161 74A5                    <1>       jz     short swpqs_3 ; invalid page directory
17763                                  <1> swpqs_9:
17764 00005163 52                      <1>       push   edx
17765                                  <1>       ; eax = page directory
17766                                  <1>       ; ebx = virtual address
17767 00005164 E82BFBFFFF              <1>       call   get_pte
17768 00005169 89D3                    <1>       mov    ebx, edx     ; PTE address
17769 0000516B 5A                      <1>       pop    edx
17770                                  <1>       ; 10/06/2016
17771 0000516C 723A                    <1>       jc     short swpqs_13 ; empty PDE
17772                                  <1>       ; EAX = PTE value
17773 0000516E A801                    <1>       test   al, PTE_A_PRESENT ; bit 0 = 1
17774 00005170 7436                    <1>       jz     short swpqs_13  ; Drop non-present page
17775                                  <1>                              ; from the queue (head)
17776 00005172 A802                    <1>       test   al, PTE_A_WRITE     ; bit 1 = 0 (read only)
17777 00005174 7432                    <1>       jz     short swpqs_13  ; Drop read only page
17778                                  <1>                              ; from the queue (head)
17779                                  <1>       ;test  al, PTE_A_ACCESS ; bit 5 = 1 (Accessed)
17780                                  <1>       ;jnz   short swpqs_11  ; present
17781                                  <1>                              ; accessed page
17782 00005176 0FBAF005                <1>         btr     eax, PTE_A_ACCESS_BIT ; reset 'accessed' bit
17783 0000517A 7210                    <1>       jc     short swpqs_11  ; accessed page
17784                                  <1>
17785 0000517C 49                      <1>       dec    ecx
17786 0000517D 66890D[60050600]        <1>       mov    [swpq_count], cx
17787 00005184 7402                    <1>         jz      short swpqs_10
17788                                  <1>             ; esi = head + 4
17789                                  <1>             ; edi = head
17790 00005186 F3A5                    <1>       rep    movsd   ; n = 1 to k-1, [n - 1] = [n]
17791                                  <1> swpqs_10:
17792 00005188 890F                    <1>       mov    [edi], ecx ; 0
17793 0000518A EB8C                    <1>       jmp    short swpqs_6 ; 26/03/2017
17794                                  <1>
17795                                  <1> swpqs_11:
17796 0000518C 8903                    <1>       mov    [ebx], eax     ; save changed attribute
17797                                  <1>       ; Rotation (head -> tail)
17798 0000518E 49                      <1>       dec    ecx     ; entry count -> last entry number
17799 0000518F 74F7                    <1>       jz     short swpqs_10
17800                                  <1>             ; esi = head + 4
17801                                  <1>             ; edi = head
17802 00005191 8B07                    <1>       mov    eax, [edi] ; 20/07/2015
17803 00005193 F3A5                    <1>       rep    movsd   ; n = 1 to k-1, [n - 1] = [n]
17804 00005195 8907                    <1>       mov    [edi], eax ; head -> tail ; [k] = [1]
17805                                  <1>
17806 00005197 668B0D[60050600]        <1>       mov    cx, [swpq_count]
17807                                  <1>
17808                                  <1> swpqs_12:
17809 0000519E BE00E00800              <1>       mov    esi, swap_queue ; head
17810 000051A3 E974FFFFFF              <1>         jmp     swpqs_7
17811                                  <1>
17812                                  <1> swpqs_13:
17813 000051A8 49                      <1>       dec    ecx
17814 000051A9 66890D[60050600]        <1>       mov    [swpq_count], cx
17815 000051B0 0F845EFFFFFF            <1>         jz      swpqs_5
17816 000051B6 EBE6                    <1>       jmp    short swpqs_12
17817                                  <1>
17818                                  <1> add_to_swap_queue:
17819                                  <1> ; temporary - 16/09/2015
17820 000051B8 C3                      <1> retn
17821                                  <1>       ; 20/02/2017
17822                                  <1>       ; 20/07/2015
17823                                  <1>       ; 24/10/2014 (Retro UNIX 386 v1 - beginning)
17824                                  <1>       ;
17825                                  <1>       ; Adds new page to swap queue
17826                                  <1>       ; (page directories and page tables must not be added
17827                                  <1>       ; to swap queue)
17828                                  <1>       ;
17829                                  <1>       ; INPUT ->
17830                                  <1>       ;     EBX = Linear (Virtual) addr for current process
17831                                  <1>       ;     [u.uno]
17832                                  <1>       ;     20/02/2017
17833                                  <1>       ;     (Linear address = CORE + user's virtual address)
17834                                  <1>       ;
17835                                  <1>       ; OUTPUT ->
17836                                  <1>       ;     EAX = [swpq_count]
17837                                  <1>       ;           (after the PTE has been added)
17838                                  <1>       ;     EAX = 0 -> Swap queue is full, (1024 entries)
17839                                  <1>       ;           the PTE could not be added.
17840                                  <1>       ;
17841                                  <1>       ; Modified Registers -> EAX
17842                                  <1>       ;
17843 000051B9 53                      <1>       push   ebx
17844 000051BA 6681E300F0              <1>         and     bx, ~PAGE_OFF ; ~0FFFh ; reset bits, 0 to 11
17845 000051BF 8A1D[B3030600]          <1>       mov    bl, [u.uno] ; current process number
17846 000051C5 E814FFFFFF              <1>       call   swap_queue_shift ; drop from the queue if
17847                                  <1>                               ; it is already on the queue
```

```
17848                                  <1>             ; then add it to the tail of the queue
17849 000051CA 0FB705[60050600]        <1>       movzx  eax, word [swpq_count]
17850 000051D1 663D0004                <1>       cmp    ax, 1024
17851 000051D5 7205                    <1>       jb     short atsq_1
17852 000051D7 6629C0                  <1>       sub    ax, ax
17853 000051DA 5B                      <1>       pop    ebx
17854 000051DB C3                      <1>       retn
17855                                  <1> atsq_1:
17856 000051DC 56                      <1>       push   esi
17857 000051DD BE00E00800              <1>       mov    esi, swap_queue
17858 000051E2 6621C0                  <1>       and    ax, ax
17859 000051E5 740A                    <1>       jz     short atsq_2
17860 000051E7 66C1E002                <1>       shl    ax, 2  ; convert to offset
17861 000051EB 01C6                    <1>       add    esi, eax
17862 000051ED 66C1E802                <1>       shr    ax, 2
17863                                  <1> atsq_2:
17864 000051F1 6640                    <1>       inc    ax
17865 000051F3 891E                    <1>       mov    [esi], ebx ; Virtual address + [u.uno] combination
17866 000051F5 66A3[60050600]          <1>       mov    [swpq_count], ax
17867 000051FB 5E                      <1>       pop    esi
17868 000051FC 5B                      <1>       pop    ebx
17869 000051FD C3                      <1>       retn
17870                                  <1>
17871                                  <1> unlink_swap_block:
17872                                  <1>       ; 15/09/2015
17873                                  <1>       ; 30/04/2015
17874                                  <1>       ; 18/04/2015
17875                                  <1>       ; 24/10/2014 (Retro UNIX 386 v1 - beginning)
17876                                  <1>       ;
17877                                  <1>       ; INPUT ->
17878                                  <1>       ;      EAX = swap disk/file offset address
17879                                  <1>       ;            (bit 1 to bit 31)
17880                                  <1>       ; OUTPUT ->
17881                                  <1>       ;      [swpd_free] is increased
17882                                  <1>       ;      (corresponding SWAP DISK ALLOC. TABLE bit is SET)
17883                                  <1>       ;
17884                                  <1>       ; Modified Registers -> EAX
17885                                  <1>       ;
17886 000051FE 53                      <1>       push   ebx
17887 000051FF 52                      <1>       push   edx
17888                                  <1>       ;
17889 00005200 C1E804                  <1>       shr    eax, SECTOR_SHIFT+1  ;3+1 ; shift sector address to
17890                                  <1>                                   ; 3 bits right
17891                                  <1>                                   ; to get swap block/page number
17892 00005203 89C2                    <1>       mov    edx, eax
17893                                  <1>       ; 15/09/2015
17894 00005205 C1EA03                  <1>       shr    edx, 3            ; to get offset to S.A.T.
17895                                  <1>                                   ; (1 allocation bit = 1 page)
17896                                  <1>                                   ; (1 allocation bytes = 8 pages)
17897 00005208 80E2FC                  <1>       and    dl, 0FCh          ; clear lower 2 bits
17898                                  <1>                                   ; (to get 32 bit position)
17899                                  <1>       ;
17900 0000520B BB00000D00              <1>       mov    ebx, swap_alloc_table ; Swap Allocation Table address
17901 00005210 01D3                    <1>       add    ebx, edx
17902 00005212 83E01F                  <1>       and    eax, 1Fh          ; lower 5 bits only
17903                                  <1>                                   ; (allocation bit position)
17904 00005215 3B05[6E050600]          <1>       cmp    eax, [swpd_next]     ; is the new free block addr. lower
17905                                  <1>                                   ; than the address in 'swpd_next' ?
17906                                  <1>                                   ; (next/first free block value)
17907 0000521B 7305                    <1>       jnb    short uswpbl_1          ; no
17908 0000521D A3[6E050600]            <1>       mov    [swpd_next], eax     ; yes
17909                                  <1> uswpbl_1:
17910 00005222 0FAB03                  <1>       bts    [ebx], eax        ; unlink/release/deallocate block
17911                                  <1>                                   ; set relevant bit to 1.
17912                                  <1>                                   ; set CF to the previous bit value
17913 00005225 F5                      <1>       cmc                      ; complement carry flag
17914 00005226 7206                    <1>       jc     short uswpbl_2          ; do not increase swfd_free count
17915                                  <1>                                   ; if the block is already deallocated
17916                                  <1>                                   ; before.
17917 00005228 FF05[6A050600]          <1>         inc     dword [swpd_free]
17918                                  <1> uswpbl_2:
17919 0000522E 5A                      <1>       pop    edx
17920 0000522F 5B                      <1>       pop    ebx
17921 00005230 C3                      <1>       retn
17922                                  <1>
17923                                  <1> link_swap_block:
17924                                  <1>       ; 01/07/2015
17925                                  <1>       ; 18/04/2015
17926                                  <1>       ; 24/10/2014 (Retro UNIX 386 v1 - beginning)
17927                                  <1>       ;
17928                                  <1>       ; INPUT -> none
17929                                  <1>       ;
17930                                  <1>       ; OUTPUT ->
17931                                  <1>       ;      EAX = OFFSET ADDRESS OF THE ALLOCATED BLOCK (4096 bytes)
17932                                  <1>       ;            in sectors (corresponding
17933                                  <1>       ;            SWAP DISK ALLOCATION TABLE bit is RESET)
17934                                  <1>       ;
17935                                  <1>       ;      CF = 1 and EAX = 0
17936                                  <1>       ;                if there is not a free block to be allocated
17937                                  <1>       ;
17938                                  <1>       ; Modified Registers -> none (except EAX)
17939                                  <1>       ;
17940                                  <1>
17941                                  <1>       ;mov   eax, [swpd_free]
17942                                  <1>       ;and   eax, eax
17943                                  <1>       ;jz    short out_of_swpspc
17944                                  <1>       ;
17945 00005231 53                      <1>       push   ebx
17946 00005232 51                      <1>       push   ecx
17947                                  <1>       ;
17948 00005233 BB00000D00              <1>       mov    ebx, swap_alloc_table ; Swap Allocation Table offset
17949 00005238 89D9                    <1>       mov    ecx, ebx
```

```
17950 0000523A 031D[6E050600]      <1>       add    ebx, [swpd_next] ; Free block searching starts from here
17951                              <1>                               ; next_free_swap_block >> 5
17952 00005240 030D[72050600]      <1>       add    ecx, [swpd_last] ; Free block searching ends here
17953                              <1>                               ; (total_swap_blocks - 1) >> 5
17954                              <1> lswbl_scan:
17955 00005246 39CB                <1>       cmp    ebx, ecx
17956 00005248 770A                <1>       ja     short lswbl_notfound
17957                              <1>       ;
17958 0000524A 0FBC03              <1>       bsf    eax, [ebx] ; Scans source operand for first bit set (1).
17959                              <1>                         ; Clears ZF if a bit is found set (1) and
17960                              <1>                         ; loads the destination with an index to
17961                              <1>                         ; first set bit. (0 -> 31)
17962                              <1>                         ; Sets ZF to 1 if no bits are found set.
17963                              <1>       ; 01/07/2015
17964 0000524D 751C                <1>       jnz    short lswbl_found ; ZF = 0 -> a free block has been found
17965                              <1>                          ;
17966                              <1>                          ; NOTE:  a Swap Disk Allocation Table bit
17967                              <1>                          ;       with value of 1 means
17968                              <1>                          ;       the corresponding page is free
17969                              <1>                          ;       (Retro UNIX 386 v1 feaure only!)
17970 0000524F 83C304              <1>       add    ebx, 4
17971                              <1>                          ; We return back for searching next page block
17972                              <1>                          ; NOTE: [swpd_free] is not ZERO; so,
17973                              <1>                          ;      we always will find at least 1 free block here.
17974 00005252 EBF2                <1>       jmp           short lswbl_scan
17975                              <1>       ;
17976                              <1> lswbl_notfound:
17977 00005254 81E900000D00        <1>       sub    ecx, swap_alloc_table
17978 0000525A 890D[6E050600]      <1>       mov    [swpd_next], ecx ; next/first free page = last page
17979                              <1>                               ; (unlink_swap_block procedure will change it)
17980 00005260 31C0                <1>       xor    eax, eax
17981 00005262 A3[6A050600]        <1>       mov    [swpd_free], eax
17982 00005267 F9                  <1>       stc
17983                              <1> lswbl_ok:
17984 00005268 59                  <1>       pop    ecx
17985 00005269 5B                  <1>       pop    ebx
17986 0000526A C3                  <1>       retn
17987                              <1>       ;
17988                              <1> ;out_of_swpspc:
17989                              <1> ;     stc
17990                              <1> ;     retn
17991                              <1>
17992                              <1> lswbl_found:
17993 0000526B 89D9                <1>       mov    ecx, ebx
17994 0000526D 81E900000D00        <1>       sub    ecx, swap_alloc_table
17995 00005273 890D[6E050600]      <1>       mov    [swpd_next], ecx ; Set first free block searching start
17996                              <1>                               ; address/offset (to the next)
17997 00005279 FF0D[6A050600]      <1>         dec     dword [swpd_free] ; 1 block has been allocated (X = X-1)
17998                              <1>       ;
17999 0000527F 0FB303              <1>       btr    [ebx], eax     ; The destination bit indexed by the source value
18000                              <1>                               ; is copied into the Carry Flag and then cleared
18001                              <1>                               ; in the destination.
18002                              <1>                               ;
18003                              <1>                               ; Reset the bit which is corresponding to the
18004                              <1>                               ; (just) allocated block.
18005 00005282 C1E105              <1>       shl    ecx, 5         ; (block offset * 32) + block index
18006 00005285 01C8                <1>       add    eax, ecx       ; = block number
18007 00005287 C1E003              <1>       shl    eax, SECTOR_SHIFT ; 3, sector (offset) address of the block
18008                              <1>                               ; 1 block =  8 sectors
18009                              <1>       ;
18010                              <1>       ; EAX = offset address of swap disk/file sector (beginning of the block)
18011                              <1>       ;
18012                              <1>       ; NOTE: The relevant page table entry will be updated
18013                              <1>       ;       according to this EAX value...
18014                              <1>       ;
18015 0000528A EBDC                <1>       jmp    short lswbl_ok
18016                              <1>
18017                              <1> logical_disk_read:
18018                              <1>       ; 20/07/2015
18019                              <1>       ; 09/03/2015 (temporary code here)
18020                              <1>       ;
18021                              <1>       ; INPUT ->
18022                              <1>       ;      ESI = Logical disk description table address
18023                              <1>       ;      EBX = Memory page (buffer) address (physical!)
18024                              <1>       ;      EAX = Sector adress (offset address, logical sector number)
18025                              <1>       ;      ECX = Sector count
18026                              <1>       ;
18027                              <1>       ;
18028 0000528C C3                  <1>       retn
18029                              <1>
18030                              <1> logical_disk_write:
18031                              <1>       ; 20/07/2015
18032                              <1>       ; 09/03/2015 (temporary code here)
18033                              <1>       ;
18034                              <1>       ; INPUT ->
18035                              <1>       ;      ESI = Logical disk description table address
18036                              <1>       ;      EBX = Memory page (buffer) address (physical!)
18037                              <1>       ;      EAX = Sector adress (offset address, logical sector number)
18038                              <1>       ;      ECX = Sector count
18039                              <1>       ;
18040 0000528D C3                  <1>       retn
18041                              <1>
18042                              <1> get_physical_addr:
18043                              <1>       ; 26/03/2017
18044                              <1>       ; 20/02/2017
18045                              <1>       ; 27/05/2016 - TRDOS 386 (TRDOS v2.0)
18046                              <1>       ; 18/10/2015
18047                              <1>       ; 29/07/2015
18048                              <1>       ; 20/07/2015
18049                              <1>       ; 04/06/2015
18050                              <1>       ; 20/05/2015
18051                              <1>       ; 28/04/2015
```

```
18052                                  <1>       ; 18/04/2015
18053                                  <1>       ; Get physical address
18054                                  <1>       ;     (allocates a new page for user if it is not present)
18055                                  <1>       ;
18056                                  <1>       ; (This subroutine is needed for mapping user's virtual
18057                                  <1>       ; (buffer) address to physical address (of the buffer).)
18058                                  <1>       ; ('sys write', 'sys read' system calls...)
18059                                  <1>       ;
18060                                  <1>       ; INPUT ->
18061                                  <1>       ;     EBX = virtual address
18062                                  <1>       ;     u.pgdir = page directory (physical) address
18063                                  <1>       ;
18064                                  <1>       ; OUTPUT ->
18065                                  <1>       ;     EAX = physical address
18066                                  <1>       ;     EBX = linear address
18067                                  <1>       ;     EDX = physical address of the page frame
18068                                  <1>       ;           (with attribute bits)
18069                                  <1>       ;     ECX = byte count within the page frame
18070                                  <1>       ;
18071                                  <1>       ; Modified Registers -> EAX, EBX, ECX, EDX
18072                                  <1>       ;
18073 0000528E 81C300004000            <1>       add    ebx, CORE ; 18/10/2015
18074                                  <1> get_physical_addr_x: ; 27/05/2016
18075 00005294 A1[B8030600]            <1>       mov    eax, [u.pgdir]
18076 00005299 E8F6F9FFFF              <1>       call   get_pte
18077                                  <1>              ; EDX = Page table entry address (if CF=0)
18078                                  <1>                ;       Page directory entry address (if CF=1)
18079                                  <1>              ;       (Bit 0 value is 0 if PT is not present)
18080                                  <1>              ; EAX = Page table entry value (page address)
18081                                  <1>              ;     CF = 1 -> PDE not present or invalid ?
18082 0000529E 731C                    <1>       jnc    short gpa_1
18083                                  <1>       ;
18084 000052A0 E8D4F8FFFF              <1>       call   allocate_page
18085 000052A5 7248                    <1>       jc     short gpa_im_err  ; 'insufficient memory' error
18086                                  <1> gpa_0:
18087 000052A7 E847F9FFFF              <1>       call   clear_page
18088                                  <1>       ; EAX = Physical (base) address of the allocated (new) page
18089 000052AC 0C07                    <1>       or     al, PDE_A_PRESENT + PDE_A_WRITE + PDE_A_USER ; 4+2+1 = 7
18090                                  <1>                          ; lower 3 bits are used as U/S, R/W, P flags
18091                                  <1>                          ; (user, writable, present page)
18092 000052AE 8902                    <1>       mov    [edx], eax ; Let's put the new page directory entry here !
18093 000052B0 A1[B8030600]            <1>       mov    eax, [u.pgdir]
18094 000052B5 E8DAF9FFFF              <1>       call   get_pte
18095 000052BA 7233                    <1>       jc     short gpa_im_err ; 'insufficient memory' error
18096                                  <1> gpa_1:
18097                                  <1>       ; EAX = PTE value, EDX = PTE address
18098 000052BC A801                    <1>       test   al, PTE_A_PRESENT
18099 000052BE 751F                    <1>       jnz    short gpa_3 ; 26/03/2017
18100 000052C0 09C0                    <1>       or     eax, eax
18101 000052C2 7456                    <1>       jz     short gpa_7  ; Allocate a new page
18102                                  <1>       ; 20/07/2015
18103 000052C4 55                      <1>       push   ebp
18104 000052C5 89DD                    <1>       mov    ebp, ebx ; virtual (linear) address
18105                                  <1>       ; reload swapped page
18106 000052C7 E878000000              <1>       call   reload_page ; 28/04/2015
18107 000052CC 5D                      <1>       pop    ebp
18108 000052CD 724A                    <1>       jc     short gpa_retn
18109                                  <1> gpa_2:
18110                                  <1>       ; 26/03/2017
18111                                  <1>       ; 20/02/2017
18112                                  <1>       ; If a page will contain a Signal Response Byte
18113                                  <1>       ; it must not be swapped out, because
18114                                  <1>       ; timer service or irq callback service
18115                                  <1>       ; will write a signal return/response byte
18116                                  <1>       ; directly by using physical address of Signal
18117                                  <1>       ; Response Byte.(Even if process is not running,
18118                                  <1>       ; or it is running with swapped out pages.)
18119                                  <1>       ;
18120                                  <1>       ; 'no_page_swap' will be set by 'systimer' or
18121                                  <1>       ; 'syscalbac' sistem functions/calls. (*)
18122                                  <1>       ;
18123 000052CF 803D[9E600100]00         <1>       cmp    byte [no_page_swap], 0
18124 000052D6 761D                    <1>       jna    short gpa_4 ; this page can be swapped out
18125                                  <1>       ; this page must not be swapped out
18126                                  <1>       ; but 'no_page_swap' must be reset here
18127                                  <1>       ; imediately for other callers (*)
18128                                  <1>       ; (otherwise, swap queue would not be long enough)
18129 000052D8 E84B000000              <1>       call   gpa_8 ; 26/03/2017
18130 000052DD EB1D                    <1>       jmp    short gpa_5
18131                                  <1> gpa_3:
18132                                  <1>       ; 26/03/2017
18133 000052DF 803D[9E600100]00         <1>       cmp    byte [no_page_swap], 0
18134 000052E6 7618                    <1>       jna    short gpa_6 ; this page can be swapped out
18135 000052E8 E83B000000              <1>       call   gpa_8
18136 000052ED EB11                    <1>       jmp    short gpa_6
18137                                  <1>
18138                                  <1> gpa_im_err:
18139 000052EF B804000000              <1>       mov    eax, ERR_MINOR_IM ; Insufficient memory (minor) error!
18140                                  <1>                                ; Major error = 0 (No protection fault)
18141 000052F4 C3                      <1>       retn
18142                                  <1> gpa_4:
18143                                  <1>       ; 20/07/2015
18144                                  <1>       ; 20/05/2015
18145                                  <1>       ; add this page to swap queue
18146 000052F5 50                      <1>       push   eax
18147                                  <1>       ; EBX = Linear (CORE+virtual) address ; 20/02/2017
18148 000052F6 E8BDFEFFFF              <1>       call   add_to_swap_queue
18149 000052FB 58                      <1>       pop    eax
18150                                  <1> gpa_5:
18151                                  <1>              ; PTE address in EDX
18152                                  <1>              ; virtual address in EBX
18153                                  <1>       ; EAX = memory page address
```

```
18154 000052FC 0C07               <1>       or    al, PTE_A_PRESENT + PTE_A_USER + PTE_A_WRITE
18155                             <1>                             ; present flag, bit 0 = 1
18156                             <1>                             ; user flag, bit 2 = 1
18157                             <1>                             ; writable flag, bit 1 = 1
18158 000052FE 8902               <1>       mov   [edx], eax  ; Update PTE value
18159                             <1> gpa_6:
18160                             <1>       ; 18/10/2015
18161 00005300 89D9               <1>       mov   ecx, ebx
18162 00005302 81E1FF0F0000       <1>       and   ecx, PAGE_OFF
18163 00005308 89C2               <1>       mov   edx, eax
18164 0000530A 662500F0           <1>       and   ax, PTE_A_CLEAR
18165 0000530E 01C8               <1>       add   eax, ecx
18166 00005310 F7D9               <1>       neg   ecx ; 1 -> -1 (0FFFFFFFFh), 4095 (0FFFh) -> -4095
18167 00005312 81C100100000       <1>       add   ecx, PAGE_SIZE
18168 00005318 F8                 <1>       clc
18169                             <1> gpa_retn:
18170 00005319 C3                 <1>       retn
18171                             <1> gpa_7:
18172 0000531A E85AF8FFFF         <1>       call  allocate_page
18173 0000531F 72CE               <1>       jc    short gpa_im_err ; 'insufficient memory' error
18174 00005321 E8CDF8FFFF         <1>       call  clear_page
18175 00005326 EBA7               <1>       jmp   short gpa_2
18176                             <1>
18177                             <1> gpa_8: ; 26/03/2017
18178 00005328 C605[9E600100]00   <1>       mov   byte [no_page_swap], 0
18179 0000532F 53                 <1>       push  ebx
18180 00005330 50                 <1>       push  eax ; 26/03/2017
18181 00005331 6681E300F0         <1>         and     bx, ~PAGE_OFF ; ~0FFFh ; reset bits, 0 to 11
18182 00005336 8A1D[B3030600]     <1>       mov   bl, [u.uno] ; current process number
18183 0000533C E89DFDFFFF         <1>       call  swap_queue_shift ; drop from the queue if
18184                             <1>                              ; it is already on the queue
18185 00005341 58                 <1>       pop   eax ; 26/03/2017
18186 00005342 5B                 <1>       pop   ebx
18187 00005343 C3                 <1>       retn
18188                             <1>
18189                             <1> reload_page:
18190                             <1>       ; 20/07/2015
18191                             <1>       ; 28/04/2015 (Retro UNIX 386 v1 - beginning)
18192                             <1>       ;
18193                             <1>       ; Reload (Restore) swapped page at memory
18194                             <1>       ;
18195                             <1>       ; INPUT ->
18196                             <1>       ;     EBP = Virtual (linear) memory address
18197                             <1>       ;     EAX = PTE value (swap disk sector address)
18198                             <1>       ;     (Swap disk sector address = bit 1 to bit 31 of EAX)
18199                             <1>       ; OUTPUT ->
18200                             <1>       ;     EAX = PHYSICAL (real/flat) ADDRESS OF RELOADED PAGE
18201                             <1>       ;
18202                             <1>       ;     CF = 1 and EAX = error code
18203                             <1>       ;
18204                             <1>       ; Modified Registers -> none (except EAX)
18205                             <1>       ;
18206 00005344 D1E8               <1>       shr   eax, 1   ; Convert PTE value to swap disk address
18207 00005346 53                 <1>       push  ebx      ;
18208 00005347 89C3               <1>       mov   ebx, eax ; Swap disk (offset) address
18209 00005349 E82BF8FFFF         <1>       call  allocate_page
18210 0000534E 720C               <1>       jc    short rlp_im_err
18211 00005350 93                 <1>       xchg  eax, ebx
18212                             <1>       ; EBX = Physical memory (page) address
18213                             <1>       ; EAX = Swap disk (offset) address
18214                             <1>       ; EBP = Virtual (linear) memory address
18215 00005351 E862FCFFFF         <1>       call  swap_in
18216 00005356 720B               <1>       jc    short rlp_swp_err  ; (swap disk/file read error)
18217 00005358 89D8               <1>       mov   eax, ebx
18218                             <1> rlp_retn:
18219 0000535A 5B                 <1>       pop   ebx
18220 0000535B C3                 <1>       retn
18221                             <1>
18222                             <1> rlp_im_err:
18223 0000535C B804000000         <1>       mov   eax, ERR_MINOR_IM ; Insufficient memory (minor) error!
18224                             <1>                               ; Major error = 0 (No protection fault)
18225 00005361 EBF7               <1>       jmp   short rlp_retn
18226                             <1>
18227                             <1> rlp_swp_err:
18228 00005363 B828000000         <1>       mov   eax, SWP_DISK_READ_ERR ; Swap disk read error !
18229 00005368 EBF0               <1>       jmp   short rlp_retn
18230                             <1>
18231                             <1>
18232                             <1> copy_page_dir:
18233                             <1>       ; 19/09/2015
18234                             <1>       ; temporary - 07/09/2015
18235                             <1>       ; 07/09/2015 (Retro UNIX 386 v1 - beginning)
18236                             <1>       ;
18237                             <1>       ; INPUT ->
18238                             <1>       ;     [u.pgdir] = PHYSICAL (real/flat) ADDRESS of the parent's
18239                             <1>       ;                 page directory.
18240                             <1>       ; OUTPUT ->
18241                             <1>       ;     EAX =  PHYSICAL (real/flat) ADDRESS of the child's
18242                             <1>       ;            page directory.
18243                             <1>       ;     (New page directory with new page table entries.)
18244                             <1>       ;     (New page tables with read only copies of the parent's
18245                             <1>       ;     pages.)
18246                             <1>       ;     EAX = 0 -> Error (CF = 1)
18247                             <1>       ;
18248                             <1>       ; Modified Registers -> none (except EAX)
18249                             <1>       ;
18250 0000536A E80AF8FFFF         <1>       call  allocate_page
18251 0000536F 723E               <1>       jc    short cpd_err
18252                             <1>       ;
18253 00005371 55                 <1>       push  ebp ; 20/07/2015
18254 00005372 56                 <1>       push  esi
18255 00005373 57                 <1>       push  edi
```

```
18256 00005374 53                  <1>       push   ebx
18257 00005375 51                  <1>       push   ecx
18258 00005376 8B35[B8030600]      <1>       mov    esi, [u.pgdir]
18259 0000537C 89C7                <1>       mov    edi, eax
18260 0000537E 50                  <1>       push   eax ; save child's page directory address
18261                              <1>       ; copy PDE 0 from the parent's page dir to the child's page dir
18262                              <1>       ; (use same system space for all user page tables)
18263 0000537F A5                  <1>       movsd
18264 00005380 BD00004000          <1>       mov    ebp, 1024*4096 ; pass the 1st 4MB (system space)
18265 00005385 B9FF030000          <1>       mov    ecx, (PAGE_SIZE / 4) - 1 ; 1023
18266                              <1> cpd_0:
18267 0000538A AD                  <1>       lodsd
18268                              <1>       ;or    eax, eax
18269                              <1>         ;jnz     short cpd_1
18270 0000538B A801                <1>       test   al, PDE_A_PRESENT ;  bit 0 =  1
18271 0000538D 7508                <1>       jnz    short cpd_1
18272                              <1>       ; (virtual address at the end of the page table)
18273 0000538F 81C500004000        <1>       add    ebp, 1024*4096 ; page size * PTE count
18274 00005395 EB0F                <1>       jmp    short cpd_2
18275                              <1> cpd_1:
18276 00005397 662500F0            <1>       and    ax, PDE_A_CLEAR ; 0F000h ; clear attribute bits
18277 0000539B 89C3                <1>       mov    ebx, eax
18278                              <1>       ; EBX = Parent's page table address
18279 0000539D E81F000000          <1>       call   copy_page_table
18280 000053A2 720C                <1>       jc     short cpd_p_err
18281                              <1>       ; EAX = Child's page table address
18282 000053A4 0C07                <1>       or     al, PDE_A_PRESENT + PDE_A_WRITE + PDE_A_USER
18283                              <1>                       ; set bit 0, bit 1 and bit 2 to 1
18284                              <1>                       ; (present, writable, user)
18285                              <1> cpd_2:
18286 000053A6 AB                  <1>       stosd
18287 000053A7 E2E1                <1>       loop   cpd_0
18288                              <1>       ;
18289 000053A9 58                  <1>       pop    eax  ; restore child's page directory address
18290                              <1> cpd_3:
18291 000053AA 59                  <1>       pop    ecx
18292 000053AB 5B                  <1>       pop    ebx
18293 000053AC 5F                  <1>       pop    edi
18294 000053AD 5E                  <1>       pop    esi
18295 000053AE 5D                  <1>       pop    ebp
18296                              <1> cpd_err:
18297 000053AF C3                  <1>       retn
18298                              <1> cpd_p_err:
18299                              <1>       ; release the allocated pages missing (recover free space)
18300 000053B0 58                  <1>       pop    eax  ; the new page directory address (physical)
18301 000053B1 8B1D[B8030600]      <1>       mov    ebx, [u.pgdir] ; parent's page directory address
18302 000053B7 E8F6F8FFFF          <1>       call   deallocate_page_dir
18303 000053BC 29C0                <1>       sub    eax, eax ; 0
18304 000053BE F9                  <1>       stc
18305 000053BF EBE9                <1>       jmp    short cpd_3
18306                              <1>
18307                              <1> copy_page_table:
18308                              <1>       ; 19/09/2015
18309                              <1>       ; temporary - 07/09/2015
18310                              <1>       ; 07/09/2015 (Retro UNIX 386 v1 - beginning)
18311                              <1>       ;
18312                              <1>       ; INPUT ->
18313                              <1>       ;      EBX = PHYSICAL (real/flat) ADDRESS of the parent's page table.
18314                              <1>       ;      EBP = page table entry index (from 'copy_page_dir')
18315                              <1>       ; OUTPUT ->
18316                              <1>       ;      EAX = PHYSICAL (real/flat) ADDRESS of the child's page table.
18317                              <1>       ;      EBP = (recent) page table index (for 'add_to_swap_queue')
18318                              <1>       ;      CF = 1 -> error
18319                              <1>       ;
18320                              <1>       ; Modified Registers -> EBP (except EAX)
18321                              <1>       ;
18322 000053C1 E8B3F7FFFF          <1>       call   allocate_page
18323 000053C6 725A                <1>       jc     short cpt_err
18324                              <1>       ;
18325 000053C8 50                  <1>       push   eax ; *
18326                              <1>       ;push  ebx
18327 000053C9 56                  <1>       push   esi
18328 000053CA 57                  <1>       push   edi
18329 000053CB 52                  <1>       push   edx
18330 000053CC 51                  <1>       push   ecx
18331                              <1>       ;
18332 000053CD 89DE                <1>       mov    esi, ebx
18333 000053CF 89C7                <1>       mov    edi, eax
18334 000053D1 89C2                <1>       mov    edx, eax
18335 000053D3 81C200100000        <1>       add    edx, PAGE_SIZE
18336                              <1> cpt_0:
18337 000053D9 AD                  <1>       lodsd
18338 000053DA A801                <1>       test   al, PTE_A_PRESENT ;  bit 0 =  1
18339 000053DC 750B                <1>       jnz    short cpt_1
18340 000053DE 21C0                <1>       and    eax, eax
18341 000053E0 7430                <1>       jz     short cpt_2
18342                              <1>       ; ebp = virtual (linear) address of the memory page
18343 000053E2 E85DFFFFFF          <1>       call   reload_page ; 28/04/2015
18344 000053E7 7234                <1>       jc     short cpt_p_err
18345                              <1> cpt_1:
18346 000053E9 662500F0            <1>       and    ax, PTE_A_CLEAR ; 0F000h ; clear attribute bits
18347 000053ED 89C1                <1>       mov    ecx, eax
18348                              <1>       ; Allocate a new page for the child process
18349 000053EF E885F7FFFF          <1>       call   allocate_page
18350 000053F4 7227                <1>       jc     short cpt_p_err
18351 000053F6 57                  <1>       push   edi
18352 000053F7 56                  <1>       push   esi
18353 000053F8 89CE                <1>       mov    esi, ecx
18354 000053FA 89C7                <1>       mov    edi, eax
18355 000053FC B900040000          <1>       mov    ecx, PAGE_SIZE/4
18356 00005401 F3A5                <1>       rep    movsd  ; copy page (4096 bytes)
18357 00005403 5E                  <1>       pop    esi
```

```
18358 00005404 5F                <1>       pop    edi
18359                            <1>       ;
18360 00005405 53                <1>       push   ebx
18361 00005406 50                <1>       push   eax
18362 00005407 89EB              <1>       mov    ebx, ebp
18363                            <1>       ; ebx = virtual address of the memory page
18364 00005409 E8AAFDFFFF        <1>       call   add_to_swap_queue
18365 0000540E 58                <1>       pop    eax
18366 0000540F 5B                <1>       pop    ebx
18367                            <1>       ;
18368                            <1>       ;or    ax, PTE_A_USER+PTE_A_PRESENT
18369 00005410 0C07              <1>       or     al, PTE_A_USER+PTE_A_WRITE+PTE_A_PRESENT
18370                            <1> cpt_2:
18371 00005412 AB                <1>       stosd  ; EDI points to child's PTE
18372                            <1>       ;
18373 00005413 81C500100000      <1>       add    ebp, 4096 ; 20/07/2015 (next page)
18374                            <1>       ;
18375 00005419 39D7              <1>       cmp    edi, edx
18376 0000541B 72BC              <1>       jb     short cpt_0
18377                            <1> cpt_p_err:
18378 0000541D 59                <1>       pop    ecx
18379 0000541E 5A                <1>       pop    edx
18380 0000541F 5F                <1>       pop    edi
18381 00005420 5E                <1>       pop    esi
18382                            <1>       ;pop   ebx
18383 00005421 58                <1>       pop    eax ; *
18384                            <1> cpt_err:
18385 00005422 C3                <1>       retn
18386                            <1>
18387                            <1> allocate_memory_block:
18388                            <1>       ; 01/05/2017
18389                            <1>       ; 28/04/2017
18390                            <1>       ; 25/04/2017
18391                            <1>       ; 01/04/2016, 02/04/2016, 03/04/2016
18392                            <1>       ; 13/03/2016, 14/03/2016
18393                            <1>       ; 12/03/2016 (TRDOS 386 = TRDOS v2.0)
18394                            <1>       ; Allocating contiguous memory pages (in the kernel's memory space)
18395                            <1>       ;
18396                            <1>       ; INPUT ->
18397                            <1>       ;     EAX = Beginning address (physical)
18398                            <1>       ;     EAX = 0 -> Allocate memory block from the first proper aperture
18399                            <1>       ;     ECX = Number of bytes to be allocated
18400                            <1>       ;
18401                            <1>       ; OUTPUT ->
18402                            <1>       ;     1) cf = 0 -> successful
18403                            <1>       ;     EAX = Beginning (physical) address of the allocated memory block
18404                            <1>       ;     ECX = Number of allocated bytes (rounded up to page borders)
18405                            <1>       ;     2) cf = 1 -> unsuccessful
18406                            <1>       ;      2.1) If EAX > 0 ->
18407                            <1>       ;           (Number of requested pages is more than # of free pages
18408                            <1>       ;            but contiguous free pages -the aperture- is not enough!)
18409                            <1>       ;           EAX = Beginning address of available aperture
18410                            <1>       ;                (one of all aperture with max. aperture size/length)
18411                            <1>       ;           ECX = Size of available aperture (memory block) in bytes
18412                            <1>       ;      2.2) If EAX = 0 -> Out of memory error
18413                            <1>       ;                  (number of free pages is less than requested number)
18414                            <1>       ;           ECX = Total number of free bytes (free pages * 4096)
18415                            <1>       ;                (It is not number of contiguous free bytes)
18416                            <1>       ;
18417                            <1>       ; (Modified Registers -> EAX, ECX)
18418                            <1>       ;
18419                            <1>       ; PURPOSE: Loading a file at memory for copying or running etc.
18420                            <1>       ; If this procedure returns with cf is set, ECX contains maximum
18421                            <1>       ; available space and EAX contains the beginning address of it.
18422                            <1>       ; If EAX has zero, ECX contains total number of free bytes.
18423                            <1>       ; If requested block has been successfully allocated (by rounding up to
18424                            <1>       ; the last page border), it must be deallocated later by using
18425                            <1>       ; 'deallocate_memory_block' procedure.
18426                            <1>
18427 00005423 52                <1>       push   edx ; *
18428 00005424 BAFF0F0000        <1>       mov    edx, PAGE_SIZE - 1   ; 4095
18429 00005429 01D0              <1>       add    eax, edx
18430 0000542B 01D1              <1>       add    ecx, edx
18431 0000542D C1E90C            <1>       shr    ecx, PAGE_SHIFT          ; 12
18432                            <1>
18433                            <1>       ; ECX = number of contiguous pages to be allocated
18434 00005430 8B15[D04D0100]    <1>       mov    edx, [free_pages]
18435                            <1>       ; 01/05/2017
18436                            <1>       ;or    ecx, ecx
18437                            <1>       ;jz    short amb3
18438                            <1>       ; If ECX=0, set cf to 1 and return with max. available mem block size
18439                            <1>
18440 00005436 39D1              <1>       cmp    ecx, edx
18441 00005438 7760              <1>       ja     short amb_3
18442                            <1>
18443 0000543A C1E80C            <1>       shr    eax, PAGE_SHIFT      ; 12
18444                            <1>
18445 0000543D 89C2              <1>       mov    edx, eax          ; page number
18446 0000543F C1EA03            <1>       shr    edx, 3            ; to get offset to M.A.T.
18447                            <1>                                ; (1 allocation bit = 1 page)
18448                            <1>                                ; (1 allocation bytes = 8 pages)
18449 00005442 80E2FC            <1>       and    dl, 0FCh          ; clear lower 2 bits
18450                            <1>                                ; (to get 32 bit position)
18451 00005445 53                <1>       push   ebx ; **
18452                            <1> amb_0:
18453 00005446 890D[885A0100]    <1>       mov    [mem_ipg_count], ecx ; initial (reset) value of page count
18454 0000544C 890D[8C5A0100]    <1>       mov    [mem_pg_count], ecx
18455 00005452 31C9              <1>       xor    ecx, ecx ; 0
18456 00005454 890D[905A0100]    <1>       mov    [mem_aperture], ecx ; 0
18457 0000545A 890D[945A0100]    <1>       mov    [mem_max_aperture], ecx ; 0
18458                            <1>
18459 00005460 BB00001000        <1>       mov    ebx, MEM_ALLOC_TBL   ; Memory Allocation Table address.
```

```
18460 00005465 3B15[D44D0100]      <1>       cmp    edx, [next_page]     ; Is the beginning page address lower
18461                              <1>                                   ; than the address in 'next_page' ?
18462                              <1>                                   ; (the first/next free page of user space)
18463 0000546B 7208                <1>       jb     short amb_1
18464 0000546D 3B15[D84D0100]      <1>       cmp    edx, [last_page]     ; is the beginning page address higher
18465                              <1>                                   ; than the address in 'last_page' ?
18466                              <1>                                   ; (end of the memory)
18467 00005473 7606                <1>       jna    short amb_2          ; no
18468                              <1> amb_1:
18469 00005475 8B15[D44D0100]      <1>       mov    edx, [next_page]     ; M.A.T. offset (1 M.A.T. byte = 8 pages)
18470                              <1> amb_2:
18471 0000547B 01D3                <1>       add    ebx, edx
18472                              <1>
18473                              <1>       ; 28/04/2017
18474                              <1>       ;xor   ecx, ecx
18475 0000547D 0FBC0B              <1>       bsf    ecx, [ebx]           ; 0 to 31
18476 00005480 89D0                <1>       mov    eax, edx
18477 00005482 C1E003              <1>       shl    eax, 3               ; *8
18478 00005485 01C8                <1>       add    eax, ecx             ; beginning page number
18479                              <1>
18480 00005487 A3[985A0100]        <1>       mov    [mem_pg_pos], eax    ; beginning page no (for curr. mem. aperture)
18481 0000548C A3[9C5A0100]        <1>       mov    [mem_max_pg_pos], eax ; beginning page no for max. mem. aperture
18482                              <1>
18483 00005491 83E01F              <1>       and    eax, 1Fh             ; lower 5 bits only (0 to 31)
18484                              <1>                                   ; (allocation bit position)
18485 00005494 750E                <1>       jnz    short amb_4          ; 0
18486 00005496 B120                <1>       mov    cl, 32
18487 00005498 EB4B                <1>       jmp    short amb_10
18488                              <1>
18489                              <1> amb_3:       ; out_of_memory
18490 0000549A 31C0                <1>       xor    eax, eax ; 0
18491 0000549C 89D1                <1>       mov    ecx, edx ; free pages
18492 0000549E C1E10C              <1>       shl    ecx, PAGE_SHIFT
18493 000054A1 5A                  <1>       pop    edx ; *
18494 000054A2 F9                  <1>       stc
18495 000054A3 C3                  <1>       retn
18496                              <1> amb_4:
18497 000054A4 8B13                <1>       mov    edx, [ebx]
18498 000054A6 88C1                <1>       mov    cl, al ; 1 to 31
18499 000054A8 D3EA                <1>       shr    edx, cl
18500 000054AA 89D0                <1>       mov    eax, edx
18501                              <1> amb_5:
18502 000054AC D1E8                <1>       shr    eax, 1 ; (***)
18503 000054AE 7317                <1>       jnc    short amb_7
18504 000054B0 FF05[905A0100]      <1>       inc    dword [mem_aperture]
18505 000054B6 FF0D[8C5A0100]      <1>       dec    dword [mem_pg_count]
18506 000054BC 7470                <1>       jz     short amb_15
18507                              <1> amb_6:
18508                              <1>       ; 28/04/2017
18509 000054BE FEC1                <1>       inc    cl
18510 000054C0 80F920              <1>       cmp    cl, 32
18511 000054C3 730D                <1>       jnb    short amb_9
18512 000054C5 EBE5                <1>       jmp    short amb_5
18513                              <1> amb_7:
18514 000054C7 50                  <1>       push   eax ; (***) allocation bits (in shifted status)
18515 000054C8 E81B010000          <1>       call   amb_26 ; set maximum memory aperture (free memory block size)
18516 000054CD 58                  <1>       pop    eax ; (***)
18517 000054CE EBEE                <1>       jmp    short amb_6
18518                              <1> amb_8:
18519                              <1>       ; 28/04/2017
18520 000054D0 B120                <1>       mov    cl, 32
18521                              <1> amb_9:
18522 000054D2 89DA                <1>       mov    edx, ebx
18523 000054D4 81EA00001000        <1>       sub    edx, MEM_ALLOC_TBL
18524 000054DA 3B15[D84D0100]      <1>       cmp    edx, [last_page]
18525 000054E0 7336                <1>       jnb    short amb_14 ; contiguous pages not enough
18526 000054E2 83C304              <1>       add    ebx, 4
18527                              <1> amb_10:
18528 000054E5 8B03                <1>       mov    eax, [ebx]
18529 000054E7 21C0                <1>       and    eax, eax
18530 000054E9 7408                <1>         jz     short amb_11 ; there is not a free page bit in this alloc dword
18531 000054EB 40                  <1>       inc    eax ; 0FFFFFFFFh -> 0
18532 000054EC 740C                <1>       jz     short amb_12 ; all of bits are set (32 free pages)
18533 000054EE 48                  <1>       dec    eax
18534 000054EF 28C9                <1>       sub    cl, cl ; 0
18535 000054F1 EBB9                <1>       jmp    short amb_5
18536                              <1> amb_11:
18537 000054F3 E8F0000000          <1>       call   amb_26 ; set maximum memory aperture (free memory block size)
18538 000054F8 EBD8                <1>       jmp    short amb_9
18539                              <1> amb_12:
18540 000054FA 390D[8C5A0100]      <1>       cmp    [mem_pg_count], ecx ; 32
18541 00005500 7306                <1>       jnb    short amb_13
18542 00005502 8B0D[8C5A0100]      <1>       mov    ecx, [mem_pg_count]
18543                              <1> amb_13:
18544 00005508 010D[905A0100]      <1>       add    [mem_aperture], ecx
18545 0000550E 290D[8C5A0100]      <1>       sub    [mem_pg_count], ecx
18546 00005514 7618                <1>       jna    short amb_15
18547 00005516 EBBA                <1>       jmp    short amb_9 ; 01/05/2017
18548                              <1> amb_14:
18549 00005518 E8CB000000          <1>       call   amb_26 ; 28/04/2017
18550 0000551D A1[9C5A0100]        <1>       mov    eax, [mem_max_pg_pos] ; begin address of max. mem aperture
18551 00005522 8B0D[945A0100]      <1>       mov    ecx, [mem_max_aperture] ; max. (largest) memory aperture
18552 00005528 F9                  <1>       stc
18553 00005529 E9AF000000          <1>         jmp     amb_25
18554                              <1>
18555                              <1> amb_15: ; OK !
18556 0000552E A1[985A0100]        <1>       mov    eax, [mem_pg_pos]    ; Beginning address as page number
18557 00005533 8B0D[905A0100]      <1>       mov    ecx, [mem_aperture]  ; Free contiguous page count (>=1)
18558                              <1> amb_16:
18559                              <1>       ; allocate contiguous memory pages (via memory allocation table bits)
18560 00005539 89C2                <1>       mov    edx, eax
18561                              <1>       ; 25/04/2017
```

```
18562 0000553B C1EA03              <1>       shr    edx, 3        ; 8 pages in one allocation byte
18563 0000553E 80E2FC              <1>       and    dl, 0FCh      ; clear lower 2 bits
18564                              <1>                            ; (for dword/32bit positioning)
18565                              <1>
18566 00005541 BB00001000          <1>       mov    ebx, MEM_ALLOC_TBL
18567 00005546 01D3                <1>       add    ebx, edx
18568 00005548 83E01F              <1>       and    eax, 1Fh ; 31
18569                              <1>       ; 03/04/2016
18570 0000554B BA20000000          <1>       mov    edx, 32
18571 00005550 28C2                <1>       sub    dl, al
18572 00005552 39CA                <1>       cmp    edx, ecx      ; ecx >= 1
18573 00005554 7602                <1>       jna    short amb_17
18574 00005556 89CA                <1>       mov    edx, ecx
18575                              <1> amb_17:
18576 00005558 29D1                <1>       sub    ecx, edx
18577 0000555A 51                  <1>       push   ecx ; ***
18578 0000555B 89D1                <1>       mov    ecx, edx
18579                              <1> amb_18:
18580 0000555D 0FB303              <1>       btr    [ebx], eax    ; The destination bit indexed by the source value
18581                              <1>                            ; is copied into the Carry Flag and then cleared
18582                              <1>                            ; in the destination.
18583 00005560 FF0D[D04D0100]      <1>       dec     dword [free_pages] ; 1 page has been allocated (X = X-1)
18584 00005566 49                  <1>       dec    ecx
18585 00005567 7404                <1>       jz     short amb_19
18586 00005569 FEC0                <1>       inc    al
18587 0000556B EBF0                <1>       jmp    short amb_18
18588                              <1> amb_19:
18589 0000556D 59                  <1>       pop    ecx ; ***
18590 0000556E 21C9                <1>       and    ecx, ecx ; 0 ?
18591 00005570 741E                <1>       jz     short amb_22
18592                              <1>       ; 01/04/2016
18593 00005572 B020                <1>       mov    al, 32
18594                              <1> amb_20:
18595 00005574 83C304              <1>       add    ebx, 4
18596 00005577 39C1                <1>       cmp    ecx, eax ; 32
18597 00005579 7305                <1>       jnb    short amb_21
18598                              <1>       ; ECX < 32
18599 0000557B 28C0                <1>       sub    al, al ; 0
18600 0000557D 50                  <1>       push   eax ; 0 ***
18601 0000557E EBDD                <1>       jmp    short amb_18
18602                              <1> amb_21:
18603 00005580 2905[D04D0100]      <1>       sub    [free_pages], eax   ; [free_pages] = [free_pages] - 32
18604 00005586 C703000000000       <1>       mov    dword [ebx], 0         ; reset 32 bits
18605 0000558C 29C1                <1>       sub    ecx, eax ; 32
18606 0000558E 75E4                <1>       jnz    short amb_20
18607                              <1> amb_22:
18608 00005590 A1[985A0100]        <1>       mov    eax, [mem_pg_pos]   ; Beginning address as page number
18609 00005595 8B0D[905A0100]      <1>       mov    ecx, [mem_aperture] ; Free contiguous page count
18610                              <1>       ; [next_page] update
18611 0000559B 89C2                <1>       mov    edx, eax
18612                              <1>       ; 03/04/2016
18613 0000559D C1EA03              <1>       shr    edx, 3            ; to get offset to M.A.T.
18614                              <1>                                ; (1 allocation bit = 1 page)
18615                              <1>                                ; (1 allocation bytes = 8 pages)
18616 000055A0 80E2FC              <1>       and    dl, 0FCh          ; clear lower 2 bits
18617                              <1>                                ; (to get 32 bit position)
18618 000055A3 3B15[D44D0100]      <1>       cmp    edx, [next_page] ; first free page pointer offset
18619 000055A9 7732                <1>       ja     short amb_25
18620 000055AB BB00001000          <1>       mov    ebx, MEM_ALLOC_TBL
18621 000055B0 833C1300            <1>       cmp    dword [ebx+edx], 0
18622 000055B4 7721                <1>       ja     short amb_24
18623 000055B6 89C2                <1>       mov    edx, eax
18624 000055B8 01CA                <1>       add    edx, ecx
18625 000055BA C1EA03              <1>       shr    edx, 3
18626 000055BD 80E2FC              <1>       and    dl, 0FCh
18627                              <1> amb_23:
18628 000055C0 833C1300            <1>       cmp    dword [ebx+edx], 0
18629 000055C4 7711                <1>       ja     short amb_24
18630 000055C6 83C204              <1>       add    edx, 4
18631 000055C9 3B15[D84D0100]      <1>       cmp    edx, [last_page]    ; last page pointer offset
18632 000055CF 76EF                <1>       jna    short amb_23
18633 000055D1 8B15[DC4D0100]      <1>       mov    edx, [first_page]   ; (for) beginning of user's space
18634                              <1> amb_24:
18635 000055D7 8915[D44D0100]      <1>       mov    [next_page], edx
18636                              <1> amb_25:
18637 000055DD 9C                  <1>       pushf
18638 000055DE C1E00C              <1>       shl    eax, PAGE_SHIFT         ; convert to phy. address in bytes
18639 000055E1 C1E10C              <1>       shl    ecx, PAGE_SHIFT         ; convert to byte counts
18640 000055E4 9D                  <1>       popf
18641 000055E5 5B                  <1>       pop    ebx ; **
18642 000055E6 5A                  <1>       pop    edx ; *
18643 000055E7 C3                  <1>       retn
18644                              <1>
18645                              <1> amb_26:     ; set maximum free memory aperture (free memory block size)
18646 000055E8 89DA                <1>       mov    edx, ebx ; current address
18647 000055EA 81EA00001000        <1>       sub    edx, MEM_ALLOC_TBL ; MAT beginning address
18648                              <1>       ; 02/04/2016
18649 000055F0 C1E203              <1>       shl    edx, 3 ; MAT byte offset * 8 = page number base
18650 000055F3 01CA                <1>       add    edx, ecx ; current page number (ecx = 0 to 32)
18651                              <1>       ;
18652 000055F5 A1[905A0100]        <1>       mov    eax, [mem_aperture]
18653 000055FA 21C0                <1>       and    eax, eax
18654 000055FC 7421                <1>         jz      short amb_27
18655 000055FE C705[905A0100]0000- <1>         mov     dword [mem_aperture], 0
18656 00005606 0000                <1>
18657 00005608 3B05[945A0100]      <1>       cmp    eax, [mem_max_aperture]
18658 0000560E 760F                <1>       jna    short amb_27
18659 00005610 A3[945A0100]        <1>       mov    [mem_max_aperture], eax
18660                              <1>       ; 25/04/2017
18661 00005615 A1[985A0100]        <1>       mov    eax, [mem_pg_pos]
18662                              <1>       ; EAX = Beginning page number of the max. aperture
18663 0000561A A3[9C5A0100]        <1>       mov    [mem_max_pg_pos], eax
```

```
18664                                  <1> amb_27:
18665 0000561F 8915[985A0100]           <1>       mov    [mem_pg_pos], edx ; current page
18666                                  <1>
18667 00005625 A1[885A0100]             <1>       mov    eax, [mem_ipg_count] ; initial (reset) value of page count
18668 0000562A A3[8C5A0100]             <1>       mov    [mem_pg_count], eax
18669                                  <1>
18670 0000562F C3                       <1>       retn
18671                                  <1>
18672                                  <1> deallocate_memory_block:
18673                                  <1>       ; 03/04/2016
18674                                  <1>       ; 14/03/2016 (TRDOS 386 = TRDOS v2.0)
18675                                  <1>       ; Deallocating contiguous memory pages (in the kernel's memory space)
18676                                  <1>       ;
18677                                  <1>       ; INPUT ->
18678                                  <1>       ;     EAX = Beginning address (physical)
18679                                  <1>       ;     ECX = Number of bytes to be deallocated
18680                                  <1>       ;
18681                                  <1>       ; OUTPUT ->
18682                                  <1>       ;     Memory Allocation Table bits will be updated
18683                                  <1>       ;     [free_pages] will be changed (increased)
18684                                  <1>       ;
18685                                  <1>       ; (Modified Registers -> EAX, ECX)
18686                                  <1>       ;
18687                                  <1>       ; PURPOSE: Unloading/Freeing a file -or an allocated memory block-
18688                                  <1>       ; at memory after copying, running, saving, reading, writing etc.
18689                                  <1>       ;
18690                                  <1>
18691 00005630 52                       <1>       push   edx ; *
18692 00005631 53                       <1>       push   ebx ; **
18693                                  <1>
18694 00005632 C1E80C                   <1>       shr    eax, PAGE_SHIFT         ; 12
18695 00005635 C1E90C                   <1>       shr    ecx, PAGE_SHIFT         ; 12
18696                                  <1>
18697                                  <1>       ; EAX = Beginning page number
18698                                  <1>       ; ECX = Number of contiguous pages to be deallocated
18699                                  <1> damb_0:
18700                                  <1>       ; deallocate contiguous memory pages (via memory allocation table bits)
18701 00005638 89C2                     <1>       mov    edx, eax
18702 0000563A C1EA03                   <1>       shr    edx, 3            ; to get offset to M.A.T.
18703                                  <1>                                  ; (1 allocation bit = 1 page)
18704                                  <1>                                  ; (1 allocation bytes = 8 pages)
18705 0000563D 80E2FC                   <1>       and    dl, 0FCh          ; clear lower 2 bits
18706                                  <1>                                  ; (to get 32 bit position)
18707 00005640 3B15[D44D0100]           <1>       cmp    edx, [next_page] ; next free page
18708 00005646 7306                     <1>       jnb    short damb_1
18709 00005648 8915[D44D0100]           <1>       mov    [next_page], edx
18710                                  <1> damb_1:
18711 0000564E BB00001000               <1>       mov    ebx, MEM_ALLOC_TBL
18712 00005653 01D3                     <1>       add    ebx, edx
18713 00005655 83E01F                   <1>       and    eax, 1Fh ; 31
18714                                  <1>
18715                                  <1>       ; 03/04/2016
18716 00005658 BA20000000               <1>       mov    edx, 32
18717 0000565D 28C2                     <1>       sub    dl, al
18718 0000565F 39CA                     <1>       cmp    edx, ecx
18719 00005661 7602                     <1>       jna    short damb_2
18720 00005663 89CA                     <1>       mov    edx, ecx
18721                                  <1> damb_2:
18722 00005665 29D1                     <1>       sub    ecx, edx
18723 00005667 51                       <1>       push   ecx ; ***
18724 00005668 89D1                     <1>       mov    ecx, edx
18725                                  <1> damb_3:
18726 0000566A 0FAB03                   <1>       bts    [ebx], eax        ; unlink/release/deallocate page
18727                                  <1>                                  ; set relevant bit to 1.
18728                                  <1>                                  ; set CF to the previous bit value
18729 0000566D FF05[D04D0100]           <1>       inc     dword [free_pages]   ; 1 page has been deallocated (X = X+1)
18730 00005673 49                       <1>       dec    ecx
18731 00005674 7404                     <1>       jz     short damb_4
18732 00005676 FEC0                     <1>       inc    al
18733 00005678 EBF0                     <1>       jmp    short damb_3
18734                                  <1> damb_4:
18735 0000567A 59                       <1>       pop    ecx ; ***
18736 0000567B 21C9                     <1>       and    ecx, ecx ; 0 ?
18737 0000567D 741E                     <1>       jz     short damb_7
18738                                  <1>       ; 03/04/2016
18739 0000567F B020                     <1>       mov    al, 32
18740                                  <1> damb_5:
18741 00005681 83C304                   <1>       add    ebx, 4
18742 00005684 39C1                     <1>       cmp    ecx, eax ; 32
18743 00005686 7305                     <1>       jnb    short damb_6
18744                                  <1>       ; ECX < 32
18745 00005688 28C0                     <1>       sub    al, al ; 0
18746 0000568A 50                       <1>       push   eax ; 0 ***
18747 0000568B EBDD                     <1>       jmp    short damb_3
18748                                  <1> damb_6:
18749 0000568D 0105[D04D0100]           <1>       add    [free_pages], eax ; [free_pages] = [free_pages] + 32
18750 00005693 C703FFFFFFFF             <1>       mov    dword [ebx], 0FFFFFFFFh ; set 32 bits
18751 00005699 29C1                     <1>       sub    ecx, eax ; 32
18752 0000569B 75E4                     <1>       jnz    short damb_5
18753                                  <1> damb_7:
18754 0000569D 5B                       <1>       pop    ebx ; **
18755 0000569E 5A                       <1>       pop    edx ; *
18756 0000569F C3                       <1>       retn
18757                                  <1>
18758                                  <1> direct_memory_access:
18759                                  <1>       ; 22/07/2017
18760                                  <1>       ; 12/05/2017
18761                                  <1>       ; 16/07/2016
18762                                  <1>       ; 12/07/2016 (TRDOS 386 = TRDOS v2.0)
18763                                  <1>       ; This processure will be called to map
18764                                  <1>       ; user's (ring 3) page tables to access phsical
18765                                  <1>       ; (flat/linear) memory addresses, directly (without
```

```
18766                                  <1>       ;  kernel's data transfer functions).
18767                                  <1>       ;
18768                                  <1>       ; Purpose: Video memory access and shared memory access.
18769                                  <1>       ;
18770                                  <1>       ; INPUT ->
18771                                  <1>       ;     EAX = Beginning address (physical).
18772                                  <1>       ;     EBX = User's buffer address ; 12/05/2017
18773                                  <1>       ;     ECX = Number of contiguous pages to be mapped.
18774                                  <1>       ; OUTPUT ->
18775                                  <1>       ;     User's page directory and pages tables
18776                                  <1>       ;     will be updated.
18777                                  <1>       ;
18778                                  <1>       ;     If an old page table entry has valid page address,
18779                                  <1>       ;     that page will be deallocated just before PTE will
18780                                  <1>       ;     be changed for direct (1 to 1) memory page access.
18781                                  <1>       ;
18782                                  <1>       ;     If old PTE value points to a swapped page,
18783                                  <1>         ;       that page (block) will be unlinked on swap disk.
18784                                  <1>       ;
18785                                  <1>       ;     Newly allocated pages (except page tables) will not
18786                                  <1>       ;     be applied to Memory Allocation Table.
18787                                  <1>       ;     AVL bit 1 (PTE bit 10) of page table entry will be
18788                                  <1>       ;     used to indicate shared (direct) memory page; then,
18789                                  <1>       ;     this page will not be deallocated later during
18790                                  <1>       ;     process termination. (Memory Allocation Table and
18791                                  <1>       ;     free memory count will not be affected.
18792                                  <1>       ;     (Except deallocating page table's itself.)
18793                                  <1>       ;
18794                                  <1>       ;     CF = 1 -> error (EAX = error code)
18795                                  <1>       ;     CF = 0 -> success (EAX = beginning address)
18796                                  <1>       ;
18797                                  <1>       ;; (Modified Registers -> none)
18798                                  <1>       ; Modified registers: ebp, edx, ecx, ebx, esi, edi
18799                                  <1>       ;
18800                                  <1>
18801                                  <1>       ;push  ebp
18802                                  <1>       ;push  ebx
18803                                  <1>       ;push  ecx
18804                                  <1>       ;push  edx
18805 000056A0 662500F0               <1>       and    ax, PTE_A_CLEAR ; clear page offset
18806 000056A4 50                     <1>       push   eax
18807                                  <1>       ;and   ecx, ecx ; page count
18808                                  <1>       ;jz    dmem_acc_7  ; 'insufficient memory' error
18809 000056A5 89C5                   <1>       mov    ebp, eax
18810 000056A7 81C300004000           <1>       add    ebx, CORE ; 12/05/2017
18811                                  <1> dmem_acc_0:
18812 000056AD 891D[88650100]         <1>       mov    [base_addr], ebx ; 12/05/2017
18813 000056B3 A1[B8030600]           <1>       mov    eax, [u.pgdir] ; page dir address (physical)
18814 000056B8 E8D7F5FFFF             <1>       call   get_pte
18815                                  <1>              ; EDX = Page table entry address (if CF=0)
18816                                  <1>                ;       Page directory entry address (if CF=1)
18817                                  <1>              ;       (Bit 0 value is 0 if PT is not present)
18818                                  <1>              ; EAX = Page table entry value (page address)
18819                                  <1>              ;     CF = 1 -> PDE not present or invalid ?
18820 000056BD 7324                   <1>       jnc    short dmem_acc_1
18821                                  <1>       ;
18822 000056BF E8B5F4FFFF             <1>       call   allocate_page
18823 000056C4 0F82AB000000           <1>         jc     dmem_acc_7  ; 'insufficient memory' error
18824                                  <1>       ;
18825 000056CA E824F5FFFF             <1>       call   clear_page
18826                                  <1>       ; EAX = Physical (base) address of the allocated (new) page
18827 000056CF 0C07                   <1>       or     al, PDE_A_PRESENT + PDE_A_WRITE + PDE_A_USER ; 4+2+1 = 7
18828                                  <1>                       ; lower 3 bits are used as U/S, R/W, P flags
18829                                  <1>                       ; (user, writable, present page)
18830 000056D1 8902                   <1>       mov    [edx], eax ; Let's put the new page directory entry here !
18831 000056D3 A1[B8030600]           <1>       mov    eax, [u.pgdir]
18832 000056D8 E8B7F5FFFF             <1>       call   get_pte
18833 000056DD 0F8292000000           <1>         jc     dmem_acc_7 ; 'insufficient memory' error
18834                                  <1> dmem_acc_1:
18835                                  <1>       ; EAX = PTE value, EDX = PTE address
18836 000056E3 A801                   <1>       test   al, PTE_A_PRESENT
18837 000056E5 750D                   <1>       jnz    short dmem_acc_2
18838 000056E7 09C0                   <1>       or     eax, eax
18839 000056E9 7468                   <1>       jz     short dmem_acc_6   ; Change PTE
18840 000056EB D1E8                   <1>       shr    eax, 1       ; swap disk block (8 sectors) address
18841                                  <1>       ; unlink swap disk block
18842 000056ED E80CFBFFFF             <1>       call   unlink_swap_block
18843 000056F2 EB5F                   <1>       jmp    short dmem_acc_6
18844                                  <1>
18845                                  <1> dmem_acc_2:
18846 000056F4 A802                   <1>       test   al, PTE_A_WRITE   ; bit 1, writable (r/w) flag
18847                                  <1>                                ; (must be 1)
18848 000056F6 7550                   <1>       jnz    short dmem_acc_4
18849                                  <1>       ; Read only -duplicated- page (belongs to a parent or a child)
18850 000056F8 66A90002               <1>         test    ax, PTE_DUPLICATED ; Was this page duplicated
18851                                  <1>                                  ; as child's page ?
18852 000056FC 7455                   <1>       jz     short dmem_acc_5 ; Change PTE but don't deallocate the page!
18853                                  <1>
18854                                  <1>       ;push  edi
18855                                  <1>       ;push  esi
18856                                  <1>
18857 000056FE 51                     <1>       push   ecx
18858                                  <1>       ;push  ebx
18859 000056FF 8B1D[BC030600]         <1>       mov    ebx, [u.ppgdir] ; parent's page dir address (physical)
18860                                  <1>
18861                                  <1>       ; check the parent's PTE value is read only & same page or not..
18862 00005705 89EF                   <1>       mov    edi, ebp
18863 00005707 C1EF16                 <1>       shr    edi, PAGE_D_SHIFT ; 22
18864                                  <1>       ; EDI = page directory entry index (0-1023)
18865 0000570A 89EE                   <1>       mov    esi, ebp
18866 0000570C C1EE0C                 <1>       shr    esi, PAGE_SHIFT ; 12
18867 0000570F 81E6FF030000           <1>       and    esi, PTE_MASK
```

```
18868                                   <1>       ; ESI = page table entry index (0-1023)
18869                                   <1>
18870 00005715 66C1E702                 <1>       shl    di, 2 ; * 4
18871 00005719 01FB                     <1>       add    ebx, edi ; PDE offset (for the parent)
18872 0000571B 8B0F                     <1>       mov    ecx, [edi]
18873 0000571D F6C101                   <1>       test   cl, PDE_A_PRESENT ; present (valid) or not ?
18874 00005720 7425                     <1>       jz     short dmem_acc_3    ; parent process does not use this page
18875 00005722 6681E100F0               <1>       and    cx, PDE_A_CLEAR ; 0F000h ; Clear attribute bits
18876 00005727 66C1E602                 <1>       shl    si, 2 ; *4
18877 0000572B 01CE                     <1>       add    esi, ecx ; PTE offset (for the parent)
18878 0000572D 8B1E                     <1>       mov    ebx, [esi]
18879 0000572F F6C301                   <1>       test   bl, PTE_A_PRESENT ; present or not ?
18880 00005732 7413                     <1>       jz     short dmem_acc_3    ; parent process does not use this page
18881 00005734 662500F0                 <1>       and    ax, PTE_A_CLEAR ; 0F000h ; Clear attribute bits
18882 00005738 6681E300F0               <1>       and    bx, PTE_A_CLEAR ; 0F000h ; Clear attribute bits
18883 0000573D 39D8                     <1>       cmp    eax, ebx     ; parent's and child's pages are same ?
18884 0000573F 7506                     <1>       jne    short dmem_acc_3    ; not same page
18885                                   <1>                              ; deallocate the child's page
18886 00005741 800E02                   <1>         or      byte [esi], PTE_A_WRITE ; convert to writable page (parent)
18887                                   <1>       ;pop   ebx
18888 00005744 59                       <1>       pop    ecx
18889 00005745 EB0C                     <1>       jmp    short dmem_acc_5
18890                                   <1> dmem_acc_3:
18891                                   <1>       ;pop   ebx
18892 00005747 59                       <1>       pop    ecx
18893                                   <1> dmem_acc_4:
18894 00005748 66A90004                 <1>       test   ax, PTE_SHARED ; shared or direct memory access indicator
18895 0000574C 7505                     <1>       jnz    short dmem_acc_5   ; AVL bit 1 = 1, do not deallocate this page!
18896                                   <1>       ;
18897                                   <1>       ;and   ax, PTE_A_CLEAR ; 0F000h ; clear lower 12 (attribute) bits
18898 0000574E E804F6FFFF               <1>       call   deallocate_page
18899                                   <1> dmem_acc_5:
18900                                   <1>       ;pop   esi
18901                                   <1>       ;pop   edi
18902                                   <1> dmem_acc_6:
18903 00005753 89E8                     <1>       mov    eax, ebp ; physical page (offset=0) address
18904                                   <1>       ; EAX = memory page address
18905                                   <1>       ; EDX = PTE entry address (physical)
18906 00005755 660D0704                 <1>       or     ax, PTE_A_PRESENT+PTE_A_USER+PTE_A_WRITE+PTE_SHARED
18907                                   <1>                     ; present flag, bit 0 = 1
18908                                   <1>                     ; user flag, bit 2 = 1
18909                                   <1>                     ; writable flag, bit 1 = 1
18910                                   <1>                     ; direct memory access flag, bit 10 = 1
18911                                   <1>                     ; (This page must not be deallocated!)
18912 00005759 8902                     <1>       mov    [edx], eax  ; Update PTE value
18913 0000575B 49                       <1>       dec    ecx ; remain count of contiguous pages
18914 0000575C 741E                     <1>       jz     short dmem_acc_8
18915 0000575E 81C500100000             <1>       add    ebp, PAGE_SIZE ; next physical page address
18916                                   <1>       ; 22/07/2017
18917                                   <1>       ;mov   eax, ebp
18918                                   <1>       ; 12/05/2017
18919 00005764 8B1D[88650100]           <1>       mov    ebx, [base_addr] ; linear address (virtual+CORE)
18920 0000576A 81C300100000             <1>       add    ebx, PAGE_SIZE      ; next linear address
18921 00005770 E938FFFFFF               <1>         jmp     dmem_acc_0
18922                                   <1> dmem_acc_7:  ; ERROR !
18923 00005775 C7042404000000           <1>       mov    dword [esp], ERR_MINOR_IM
18924                                   <1>              ; Insufficient memory (minor) error!
18925                                   <1>              ; Major error = 0 (No protection fault)
18926                                   <1>       ; cf = 1
18927                                   <1> dmem_acc_8:
18928 0000577C 58                       <1>       pop    eax
18929                                   <1>       ;pop   edx
18930                                   <1>       ;pop   ecx
18931                                   <1>       ;pop   ebx
18932                                   <1>       ;pop   ebp
18933 0000577D C3                       <1>       retn
18934                                   <1>
18935                                   <1> deallocate_user_pages:
18936                                   <1>       ; 20/05/2017
18937                                   <1>       ; 15/05/2017
18938                                   <1>       ; 20/02/2017
18939                                   <1>       ; 19/02/2017 (TRDOS 386 = TRDOS v2.0)
18940                                   <1>       ;
18941                                   <1>       ; Deallocate virtually contiguous user pages (memory block)
18942                                   <1>       ; (caller: 'sysdalloc' system call)
18943                                   <1>       ;
18944                                   <1>       ; INPUT ->
18945                                   <1>       ;     EBX = VIRTUAL ADDRESS (beginning address)
18946                                   <1>       ;     ECX = byte count
18947                                   <1>       ;     [u.pgdir] = user's page directory
18948                                   <1>       ;     [u.ppdir] = parent's page directory
18949                                   <1>       ;
18950                                   <1>       ; OUTPUT ->
18951                                   <1>       ;    If CF = 0
18952                                   <1>       ;      EAX = Deallocated memory bytes
18953                                   <1>       ;        (Even if shared or read only pages will not be
18954                                   <1>       ;         deallocated on M.A.T., this byte count will be
18955                                   <1>       ;         returned as virtually deallocated bytes; in fact
18956                                   <1>       ;         virtually deallocated user pages * 4096.)
18957                                   <1>       ;     EBX = Virtual address (as rounded up)
18958                                   <1>       ;    If CF = 1
18959                                   <1>       ;      EAX = 0 (there is not any deallocated pages)
18960                                   <1>       ;
18961                                   <1>       ; Note: Empty page tables will not be deallocated!!!
18962                                   <1>       ;     (they will be deallocated at process termination stage)
18963                                   <1>       ;
18964                                   <1>       ; Modified Registers -> EAX, EDX, ESI, EDI, EBX, ECX, EBP
18965                                   <1>       ;
18966 0000577E 89DE                     <1>       mov    esi, ebx
18967 00005780 89F7                     <1>       mov    edi, esi
18968 00005782 01CF                     <1>       add    edi, ecx
18969 00005784 81C6FF0F0000             <1>       add    esi, PAGE_SIZE - 1  ; 4095 (round up)
```

```
18970 0000578A C1EE0C              <1>       shr    esi, PAGE_SHIFT
18971 0000578D C1EF0C              <1>       shr    edi, PAGE_SHIFT
18972 00005790 89F8                <1>       mov    eax, edi ; end page
18973 00005792 29F0                <1>       sub    eax, esi ; end page - start page
18974 00005794 0F86D5000000        <1>       jna    da_u_pd_err  ; < 1
18975 0000579A 89F3                <1>       mov    ebx, esi
18976 0000579C C1E30C              <1>       shl    ebx, PAGE_SHIFT ; virtual address (as rounded up)
18977 0000579F 53                  <1>       push   ebx ; *
18978 000057A0 89C1                <1>       mov    ecx, eax ; page count
18979 000057A2 C1E00C              <1>       shl    eax, PAGE_SHIFT ; byte count as adjusted
18980 000057A5 50                  <1>       push   eax ; **
18981 000057A6 8B1D[B8030600]      <1>       mov    ebx, [u.pgdir] ; physical addr of user's page dir
18982 000057AC 81C600040000        <1>       add    esi, CORE/PAGE_SIZE
18983 000057B2 89F7                <1>       mov    edi, esi
18984 000057B4 81E7FF030000        <1>       and    edi, PTE_MASK ; PTE entry in the page table
18985 000057BA 57                  <1>       push   edi ; *** ; PTE index (of page directory)
18986 000057BB C1EE0A              <1>       shr    esi, PAGE_D_SHIFT - PAGE_SHIFT ; 22-12=10
18987 000057BE 89F2                <1>       mov    edx, esi
18988                              <1>       ; EDX = PDE index
18989 000057C0 C1E602              <1>       shl    esi, 2 ; convert PDE index to dword offset
18990 000057C3 01DE                <1>       add    esi, ebx ; add page directory address
18991                              <1> da_u_pd_1:
18992 000057C5 AD                  <1>       lodsd
18993                              <1>       ;
18994 000057C6 89F5                <1>       mov    ebp, esi ; 20/02/2017
18995                              <1>       ; EBP = next PDE address
18996                              <1>       ;
18997 000057C8 A801                <1>       test   al, PDE_A_PRESENT ; bit 0, present flag (must be 1)
18998 000057CA 0F8494000000        <1>       jz     da_u_pd_3 ; 20/05/2017
18999 000057D0 662500F0            <1>       and    ax, PDE_A_CLEAR ; 0F000h ; clear lower 12 (attribute) bits
19000                              <1>       ; EAX = PHYSICAL (flat) ADDRESS OF THE PAGE TABLE
19001 000057D4 8B3C24              <1>       mov    edi, [esp] ; ***
19002                              <1>       ; EDI = PTE index (of complete page directory)
19003                              <1>       ;and   edi, PTE_MASK ; PTE entry in the page table
19004 000057D7 C1E702              <1>       shl    edi, 2 ; convert PTE index to dword offset
19005 000057DA 89FE                <1>       mov    esi, edi ; PTE offset in page table (0-4092)
19006 000057DC 01C6                <1>       add    esi, eax ; now, esi points to requested PTE
19007                              <1> da_u_pt_0:
19008 000057DE AD                  <1>       lodsd
19009 000057DF A801                <1>       test   al, PTE_A_PRESENT ; bit 0, present flag (must be 1)
19010 000057E1 743F                <1>       jz     short da_u_pt_1
19011                              <1>       ;
19012 000057E3 A802                <1>       test   al, PTE_A_WRITE   ; bit 1, writable (r/w) flag
19013                              <1>                                ; (must be 1)
19014 000057E5 7549                <1>       jnz    short da_u_pt_3
19015                              <1>       ; Read only -duplicated- page (belongs to a parent or a child)
19016 000057E7 66A90002            <1>         test    ax, PTE_DUPLICATED ; Was this page duplicated
19017                              <1>                                    ; as child's page ?
19018 000057EB 744E                <1>       jz     short da_u_pt_4 ; Clear PTE but don't deallocate the page!
19019                              <1>       ;
19020                              <1>       ; check the parent's PTE value is read only & same page or not..
19021                              <1>       ; EDX = page directory entry index (0-1023)
19022 000057ED 52                  <1>       push   edx ; ****
19023                              <1>       ; EDI = page table entry offset (0-4092)
19024 000057EE 8B1D[BC030600]      <1>       mov    ebx, [u.ppgdir] ; page directory of the parent process
19025 000057F4 66C1E202            <1>       shl    dx, 2 ; *4
19026 000057F8 01D3                <1>       add    ebx, edx ; PDE address (for the parent)
19027 000057FA 8B13                <1>       mov    edx, [ebx] ; page table address
19028 000057FC F6C201              <1>       test   dl, PDE_A_PRESENT ; present (valid) or not ?
19029 000057FF 742E                <1>       jz     short da_u_pt_2     ; parent process does not use this page
19030 00005801 6681E200F0          <1>       and    dx, PDE_A_CLEAR ; 0F000h ; Clear attribute bits
19031                              <1>       ; EDI = page table entry offset (0-4092)
19032 00005806 01D7                <1>       add    edi, edx     ; PTE address (for the parent)
19033 00005808 8B1F                <1>       mov    ebx, [edi]
19034 0000580A F6C301              <1>       test   bl, PTE_A_PRESENT ; present or not ?
19035 0000580D 7420                <1>       jz     short da_u_pt_2     ; parent process does not use this page
19036 0000580F 662500F0            <1>       and    ax, PTE_A_CLEAR ; 0F000h ; Clear attribute bits
19037 00005813 6681E300F0          <1>       and    bx, PTE_A_CLEAR ; 0F000h ; Clear attribute bits
19038 00005818 39D8                <1>       cmp    eax, ebx     ; parent's and child's pages are same ?
19039 0000581A 7513                <1>       jne    short da_u_pt_2     ; not same page
19040                              <1>                                ; deallocate the child's page
19041 0000581C 800F02              <1>         or      byte [edi], PTE_A_WRITE ; convert to writable page (parent)
19042 0000581F 5A                  <1>       pop    edx ; ****
19043 00005820 EB19                <1>       jmp    short da_u_pt_4
19044                              <1> da_u_pt_1:
19045 00005822 09C0                <1>       or     eax, eax     ; swapped page ?
19046 00005824 741C                <1>       jz     short da_u_pt_5     ; no
19047                              <1>                                ; yes
19048 00005826 D1E8                <1>       shr    eax, 1
19049 00005828 E8D1F9FFFF          <1>       call   unlink_swap_block ; Deallocate swapped page block
19050                              <1>                                ; on the swap disk (or in file)
19051 0000582D EB13                <1>       jmp    short da_u_pt_5
19052                              <1> da_u_pt_2:
19053 0000582F 5A                  <1>       pop    edx ; ****
19054                              <1> da_u_pt_3:
19055 00005830 66A90004            <1>       test   ax, PTE_SHARED      ; shared or direct memory access indicator
19056 00005834 7505                <1>       jnz    short da_u_pt_4     ; AVL bit 1 = 1, do not deallocate this page!
19057                              <1>       ;
19058                              <1>       ;and   ax, PTE_A_CLEAR ; 0F000h ; clear lower 12 (attribute) bits
19059 00005836 E81CF5FFFF          <1>       call   deallocate_page ; set the mem allocation bit of this page
19060                              <1> da_u_pt_4:
19061 0000583B C746FC00000000      <1>       mov    dword [esi-4], 0 ; clear/reset PTE (child, dupl. as parent)
19062                              <1> da_u_pt_5:
19063                              <1>       ; 20/05/2017
19064 00005842 58                  <1>       pop    eax ; *** PTE index (of page directory)
19065 00005843 49                  <1>       dec    ecx ; remain page count
19066 00005844 7426                <1>       jz     short da_u_pd_4
19067 00005846 40                  <1>       inc    eax ; next PTE
19068 00005847 6625FF03            <1>       and    ax, PTE_MASK ; PTE entry index in the page table
19069 0000584B 50                  <1>       push   eax ; *** (save again)
19070                              <1>       ;mov   edi, eax
19071                              <1>       ;and   di, PTE_MASK
```

```
19072                                  <1>       ;cmp   edi, PAGE_SIZE / 4 ; 1024
19073                                  <1>       ;jnb   short da_u_pd_2
19074 0000584C 89C7                    <1>       mov    edi, eax
19075 0000584E C1E702                  <1>       shl    edi, 2 ; convert index to dword offset
19076                                  <1>       ;test  ax, PTE_MASK ; 3FFh
19077 00005851 09C0                    <1>       or     eax, eax
19078 00005853 7589                    <1>       jnz    short da_u_pt_0 ; 1-1023
19079                                  <1> da_u_pd_2:
19080 00005855 42                      <1>       inc    edx
19081                                  <1>       ; 20/05/2017
19082 00005856 6681E2FF03              <1>       and    dx, PTE_MASK  ; 3FFh
19083 0000585B 740F                    <1>       jz     short da_u_pd_4  ; 0 (1024)
19084                                  <1>       ;cmp   edx, 1024
19085                                  <1>       ;jnb   short da_u_pd_4
19086 0000585D 89EE                    <1>       mov    esi, ebp ; 20/02/2017
19087 0000585F E961FFFFFF              <1>       jmp    da_u_pd_1
19088                                  <1> da_u_pd_3:
19089                                  <1>       ; 15/05/2017 (empty page directory entry)
19090 00005864 81E900040000            <1>       sub    ecx, 1024
19091 0000586A 77E9                    <1>       ja     short da_u_pd_2 ; 20/05/2017
19092                                  <1> da_u_pd_4:
19093 0000586C 58                      <1>       pop    eax ; **
19094 0000586D 5B                      <1>       pop    ebx ; *
19095 0000586E C3                      <1>       retn
19096                                  <1>
19097                                  <1> da_u_pd_err:
19098 0000586F 31C0                    <1>       xor    eax, eax
19099 00005871 F9                      <1>       stc
19100 00005872 C3                      <1>       retn
19101                                  <1>
19102                                  <1> allocate_user_pages:
19103                                  <1>       ; 20/05/2017
19104                                  <1>       ; 01/05/2017, 02/05/2017, 15/05/2017
19105                                  <1>       ; 04/03/2017
19106                                  <1>       ; 20/02/2017 (TRDOS 386 = TRDOS v2.0)
19107                                  <1>       ;
19108                                  <1>       ; Allocate physically contiguous user pages (memory block)
19109                                  <1>       ; (caller: 'sysalloc' system call)
19110                                  <1>       ;
19111                                  <1>       ; Note: This procedure does not alloc a page's itself
19112                                  <1>       ;     (page bit) on Memory Allocation Table.
19113                                  <1>       ;     (allocate_memory_block is needed before this proc)
19114                                  <1>       ;
19115                                  <1>       ; INPUT ->
19116                                  <1>       ;     EAX = PHYSICAL ADDRESS (beginning address)
19117                                  <1>       ;     EBX = VIRTUAL ADDRESS (beginning address)
19118                                  <1>       ;     ECX = byte count (>=4096)
19119                                  <1>       ;     [u.pgdir] = user's page directory
19120                                  <1>       ;
19121                                  <1>       ;     Note: All addresses are (must be) already adjusted
19122                                  <1>       ;     to page      borders, otherwise, lower 12bits of addresses
19123                                  <1>       ;     and byte count would be truncated.
19124                                  <1>       ;
19125                                  <1>       ; OUTPUT ->
19126                                  <1>       ;     none
19127                                  <1>       ;
19128                                  <1>       ;     CF = 1 -> insufficient memory error
19129                                  <1>       ;
19130                                  <1>       ; Note: All pages will be allocated in physical page order
19131                                  <1>       ;     from the beginning page address.
19132                                  <1>       ;     * A new page table will be added to the page dir
19133                                  <1>       ;       when the requested PDE is invalid.
19134                                  <1>       ;     * Those pages will not be added to swap queue
19135                                  <1>       ;       because main purpose of this allocation is to
19136                                  <1>       ;       set a direct memory access (DMA controller) buffer.
19137                                  <1>       ;      (Swapping out a page in a DMA buffer would be wrong!)
19138                                  <1>       ;     * Previous content of page tables (PTEs) would be
19139                                  <1>       ;       (should be) deallocated before entering this
19140                                  <1>       ;       procedure. So, new page table entries (PTEs)
19141                                  <1>       ;       directly will be written without checking
19142                                  <1>       ;       their previous content.
19143                                  <1>       ;     * Only solution to increase free memory by removing
19144                                  <1>       ;       that non-swappable memory block is to terminate
19145                                  <1>       ;       the process or to wait until the process will
19146                                  <1>       ;       deallocate that memory block as itself. ('sysdalloc')
19147                                  <1>       ;       (No problem, if the process does not grab all of
19148                                  <1>       ;       -very big amount of- free memory by using
19149                                  <1>       ;       'sysalloc' system call!?)
19150                                  <1>       ;       (Even if the process has grabbed all of free memory,
19151                                  <1>       ;       no problem if the process is not running in
19152                                  <1>       ;       multitasking mode. No problem in multitasking
19153                                  <1>       ;       mode if there is not another process which is running
19154                                  <1>       ;       or waiting or sleeping for an event as it's pages
19155                                  <1>       ;       are swapped-out. But a new process can not start to
19156                                  <1>       ;       run if all of free memory has beeen allocated
19157                                  <1>       ;       by running processes. Deallocation -'sysdalloc'-
19158                                  <1>       ;       or terminate a running process is needed
19159                                  <1>       ;       in order to run a new process.)
19160                                  <1>       ;
19161                                  <1>       ; Modified Registers -> EAX, EDX, ESI, EDI, EBX, ECX, EBP
19162                                  <1>       ;
19163                                  <1>
19164                                  <1>       ; 01/05/2017
19165 00005873 662500F0                <1>       and    ax, ~PAGE_OFF
19166 00005877 6681E300F0              <1>       and    bx, ~PAGE_OFF
19167                                  <1>       ; 02/05/2017
19168 0000587C BD00F0FFFF              <1>       mov    ebp, 0FFFFF000h ; 4 Giga Bytes - 4096 Bytes (for Stack)
19169 00005881 C1E90C                  <1>       shr    ecx, PAGE_SHIFT ; page count
19170 00005884 83F901                  <1>       cmp    ecx, 1
19171 00005887 7251                    <1>       jb     short a_u_im_retn
19172 00005889 89C2                    <1>       mov    edx, eax
19173 0000588B 01CA                    <1>       add    edx, ecx
```

```
19174 0000588D 724B              <1>       jc    short a_u_im_retn
19175 0000588F 39D5              <1>       cmp   ebp, edx
19176 00005891 7247              <1>       jb    short a_u_im_retn
19177 00005893 89DA              <1>       mov   edx, ebx
19178 00005895 81C200004000      <1>       add   edx, CORE
19179 0000589B 723D              <1>       jc    short a_u_im_retn
19180 0000589D 01CA              <1>       add   edx, ecx
19181 0000589F 7239              <1>       jc    short a_u_im_retn
19182 000058A1 39D5              <1>       cmp   ebp, edx
19183 000058A3 7235              <1>       jb    short a_u_im_retn
19184                            <1>       ;
19185 000058A5 89C5              <1>       mov   ebp, eax ; physical address
19186 000058A7 89DE              <1>       mov   esi, ebx
19187 000058A9 81C600004000      <1>       add   esi, CORE ; start of user's memory (4M)
19188 000058AF C1EE0C            <1>       shr   esi, PAGE_SHIFT ; higher 20 bits of the linear address
19189                            <1>       ;shr   ecx, PAGE_SHIFT ; page count
19190 000058B2 8B1D[B8030600]    <1>       mov   ebx, [u.pgdir] ; physical addr of user's page dir
19191 000058B8 89F7              <1>       mov   edi, esi
19192 000058BA 81E7FF030000      <1>       and   edi, PTE_MASK ; PTE entry index in the page table
19193 000058C0 57                <1>       push  edi  ; * ; PTE index (in page directory)
19194 000058C1 C1EE0A            <1>       shr   esi, PAGE_D_SHIFT - PAGE_SHIFT ; 22-12=10
19195 000058C4 89F2              <1>       mov   edx, esi
19196                            <1>       ; EDX = PDE index
19197 000058C6 C1E602            <1>       shl   esi, 2 ; convert PDE index to dword offset
19198 000058C9 01DE              <1>       add   esi, ebx ; add page directory address
19199                            <1> a_u_pd_0:
19200 000058CB AD                <1>       lodsd
19201                            <1>       ;
19202 000058CC 89F3              <1>       mov   ebx, esi ; next PDE address
19203                            <1>       ;
19204 000058CE A801              <1>       test  al, PDE_A_PRESENT ; bit 0, present flag (must be 1)
19205 000058D0 7513              <1>       jnz   short a_u_pd_2
19206                            <1>       ;
19207                            <1>       ; empty PDE (it does not point to valid page table address)
19208 000058D2 E8A2F2FFFF        <1>       call  allocate_page  ; (allocate a new page table)
19209 000058D7 7302              <1>       jnc   short a_u_pd_1 ; OK... now, we have a new page table.
19210                            <1>       ; cf = 1
19211                            <1>       ; There is not a free memory page to allocate a new page table !!!
19212 000058D9 5E                <1>       pop   esi ; *
19213                            <1> a_u_im_retn:
19214 000058DA C3                <1>       retn  ; return to 'sysalloc' with 'insufficient memory' error
19215                            <1>       ;
19216                            <1> a_u_pd_1: ; clear the new page table content
19217                            <1>       ; EAX = Physical (base) address of the new page table
19218 000058DB E813F3FFFF        <1>       call  clear_page ; Clear page content
19219                            <1>       ;
19220 000058E0 0C07              <1>       or    al, PDE_A_PRESENT + PDE_A_WRITE + PDE_A_USER
19221                            <1>             ; set bit 0, bit 1 and bit 2 to 1
19222                            <1>             ; (present, writable, user)
19223 000058E2 8946FC            <1>       mov   [esi-4], eax
19224                            <1> a_u_pd_2:
19225 000058E5 662500F0          <1>       and   ax, PDE_A_CLEAR ; 0F000h ; clear lower 12 (attribute) bits
19226                            <1>       ; EAX = PHYSICAL (flat) ADDRESS OF THE PAGE TABLE
19227 000058E9 8B3C24            <1>       mov   edi, [esp] ; *
19228                            <1>       ; EDI = PTE index (of page directory)
19229                            <1>       ;and  edi, PTE_MASK ; PTE entry index in the page table
19230                            <1>       ; EBX = next PDE address
19231 000058EC 89FE              <1>       mov   esi, edi ; PTE index in page table (0-1023)
19232 000058EE C1E702            <1>       shl   edi, 2 ; convert PTE index to dword offset
19233 000058F1 01C7              <1>       add   edi, eax ; now, edi points to requested PTE
19234                            <1> a_u_pt_0:
19235                            <1>       ; 02/05/2017
19236 000058F3 8B07              <1>       mov   eax, [edi]
19237                            <1>       ;
19238 000058F5 A801              <1>       test  al, PTE_A_PRESENT ; bit 0, present flag (must be 1)
19239 000058F7 7445              <1>       jz    short a_u_pt_1
19240                            <1>       ;
19241 000058F9 A802              <1>       test  al, PTE_A_WRITE   ; bit 1, writable (r/w) flag
19242                            <1>                               ; (must be 1)
19243 000058FB 7550              <1>       jnz   short a_u_pt_3
19244                            <1>       ; Read only -duplicated- page (belongs to a parent or a child)
19245 000058FD 66A90002          <1>         test    ax, PTE_DUPLICATED ; Was this page duplicated
19246                            <1>                                    ; as child's page ?
19247 00005901 7455              <1>       jz    short a_u_pt_4      ; Clear PTE but don't deallocate the page!
19248                            <1>       ;
19249                            <1>       ; check the parent's PTE value is read only & same page or not..
19250                            <1>       ; EDX = page directory entry index (0-1023)
19251 00005903 52                <1>       push  edx ; **
19252 00005904 53                <1>       push  ebx ; ***
19253                            <1>       ; ESI = page table entry index (0-1023)
19254                            <1>       ;push  esi ; **** ; 20/05/2017
19255 00005905 8B1D[BC030600]    <1>       mov   ebx, [u.ppgdir] ; page directory of the parent process
19256 0000590B 66C1E202          <1>       shl   dx, 2 ; *4
19257 0000590F 01D3              <1>       add   ebx, edx ; PTE address,0 (for the parent)
19258 00005911 8B13              <1>       mov   edx, [ebx] ; page table address
19259 00005913 F6C201            <1>       test  dl, PDE_A_PRESENT ; present (valid) or not ?
19260 00005916 7433              <1>       jz    short a_u_pt_2      ; parent process does not use this page
19261 00005918 6681E200F0        <1>       and   dx, PDE_A_CLEAR ; 0F000h ; Clear attribute bits
19262 0000591D 66C1E602          <1>       shl   si, 2 ; *4
19263                            <1>       ; ESI = page table entry offset (0-4092)
19264 00005921 01D6              <1>       add   esi, edx     ; PTE address (for the parent)
19265 00005923 8B1E              <1>       mov   ebx, [esi]
19266 00005925 F6C301            <1>       test  bl, PTE_A_PRESENT ; present or not ?
19267 00005928 7421              <1>       jz    short a_u_pt_2      ; parent process does not use this page
19268 0000592A 662500F0          <1>       and   ax, PTE_A_CLEAR ; 0F000h ; Clear attribute bits
19269 0000592E 6681E300F0        <1>       and   bx, PTE_A_CLEAR ; 0F000h ; Clear attribute bits
19270 00005933 39D8              <1>       cmp   eax, ebx     ; parent's and child's pages are same ?
19271 00005935 7514              <1>       jne   short a_u_pt_2      ; not same page
19272                            <1>                                 ; deallocate the child's page
19273 00005937 800E02            <1>         or      byte [esi], PTE_A_WRITE ; convert to writable page (parent)
19274                            <1>       ;pop  esi ; **** ; 20/05/2017
19275 0000593A 5B                <1>       pop   ebx ; ***
```

```
19276 0000593B 5A                <1>       pop    edx ; **
19277 0000593C EB1A              <1>       jmp    short a_u_pt_4
19278                            <1> a_u_pt_1:
19279 0000593E 09C0              <1>       or     eax, eax     ; swapped page ?
19280 00005940 7416              <1>       jz     short a_u_pt_4      ; no
19281                            <1>                                  ; yes
19282 00005942 D1E8              <1>       shr    eax, 1
19283 00005944 E8B5F8FFFF        <1>       call   unlink_swap_block ; Deallocate swapped page block
19284                            <1>                                  ; on the swap disk (or in file)
19285 00005949 EB0D              <1>       jmp    short a_u_pt_4
19286                            <1> a_u_pt_2:
19287                            <1>       ;pop   esi ; **** ; 20/05/2017
19288 0000594B 5B                <1>       pop    ebx ; ***
19289 0000594C 5A                <1>       pop    edx ; **
19290                            <1> a_u_pt_3:
19291 0000594D 66A90004          <1>       test   ax, PTE_SHARED      ; shared or direct memory access indicator
19292 00005951 7505              <1>       jnz    short a_u_pt_4      ; AVL bit 1 = 1, do not deallocate this page!
19293                            <1>       ;
19294                            <1>       ;and   ax, PTE_A_CLEAR ; 0F000h ; clear lower 12 (attribute) bits
19295 00005953 E8FFF3FFFF        <1>       call   deallocate_page ; set the mem allocation bit of this page
19296                            <1>       ;
19297                            <1> a_u_pt_4:
19298 00005958 89E8              <1>       mov    eax, ebp ; physical address
19299 0000595A 0C07              <1>       or     al, PTE_A_PRESENT + PTE_A_WRITE + PTE_A_USER ; 04/03/2017
19300 0000595C AB                <1>       stosd
19301 0000595D 5E                <1>       pop    esi ; * ; 20/05/2017
19302 0000595E 49                <1>       dec    ecx ; remain page count
19303 0000595F 7417              <1>       jz     short a_u_pd_5
19304 00005961 81C500100000      <1>       add    ebp, PAGE_SIZE
19305 00005967 46                <1>       inc    esi ; next PTE (index)
19306                            <1>       ; 20/05/2017
19307                            <1>       ;cmp   esi, PAGE_SIZE/4 ; 1024
19308                            <1>       ;jb    short a_u_pt_0
19309 00005968 6681E6FF03        <1>       and    si, PTE_MASK ; 3FFh (0 to 1023)
19310 0000596D 56                <1>       push   esi ; *
19311 0000596E 7583              <1>       jnz    short a_u_pt_0 ; > 0 (<1024)
19312                            <1> a_u_pd_3:
19313 00005970 42                <1>       inc    edx
19314                            <1> ;     cmp    edx, 1024
19315                            <1> ;     jnb    short a_u_pd_4 ; 02/05/2017 (error!, ecx > 0)
19316 00005971 89DE              <1>       mov    esi, ebx ; the next PDE address
19317 00005973 E953FFFFFF        <1>       jmp    a_u_pd_0
19318                            <1> a_u_pd_4:
19319                            <1>       ; 02/05/2017
19320                            <1> ;     stc
19321                            <1> a_u_pd_5:
19322                            <1>       ; 20/05/2017
19323                            <1>       ;pop   edi ; *
19324 00005978 C3                <1>       retn
19325                            <1>
19326                            <1>
19327                            <1> ; /// End Of MEMORY MANAGEMENT FUNCTIONS ///
19328                            <1>
19329                            <1> ;; Data:
19330                            <1>
19331                            <1> ; 09/03/2015
19332                            <1> ;swpq_count: dw 0 ; count of pages on the swap que
19333                            <1> ;swp_drv:    dd 0 ; logical drive description table address of the swap drive/disk
19334                            <1> ;swpd_size:  dd 0 ; size of swap drive/disk (volume) in sectors (512 bytes).

19335                            <1> ;swpd_free:  dd 0 ; free page blocks (4096 bytes) on swap disk/drive (logical)
19336                            <1> ;swpd_next:  dd 0 ; next free page block
19337                            <1> ;swpd_last:  dd 0 ; last swap page block
19338                                %include 'timer.s'   ; 17/01/2015
19339                            <1> ; ****************************************************************************
19340                            <1> ; TRDOS386.ASM (TRDOS 386 Kernel) - v2.0.0 - timer.s
19341                            <1> ; ----------------------------------------------------------------------------
19342                            <1> ; Last Update: 15/01/2017
19343                            <1> ; ----------------------------------------------------------------------------
19344                            <1> ; Beginning: 17/01/2016
19345                            <1> ; ----------------------------------------------------------------------------
19346                            <1> ; Assembler: NASM version 2.11 (trdos386.s)
19347                            <1> ; ----------------------------------------------------------------------------
19348                            <1> ; Turkish Rational DOS
19349                            <1> ; Operating System Project v2.0 by ERDOGAN TAN (Beginning: 04/01/2016)
19350                            <1> ;
19351                            <1> ; Derived from 'Retro UNIX 386 Kernel - v0.2.1.0' source code by Erdogan Tan
19352                            <1> ;
19353                            <1> ; Derived from 'IBM PC-AT' BIOS source code (1985)
19354                            <1> ; ****************************************************************************
19355                            <1>
19356                            <1> ; TRDOS 386  (TRDOS v2.0) Kernel - TIMER & REAL TIME CLOCK (BIOS) FUNCTIONS
19357                            <1>
19358                            <1> ; IBM PC-AT BIOS Source Code ('BIOS2.ASM')
19359                            <1> ; TITLE BIOS2 ---- 06/10/85 BIOS INTERRUPT ROUTINES
19360                            <1>
19361                            <1> ;
19362                            <1> ; ///////// TIMER (& REAL TIME CLOCK) FUNCTIONS //////////////
19363                            <1>
19364                            <1> int1Ah:
19365                            <1>       ; 29/01/2016
19366                            <1>       ; 17/01/2016 (TRDOS 386 = TRDOS v2.0)
19367 00005979 9C                <1>       pushfd
19368 0000597A 0E                <1>       push   cs
19369 0000597B E801000000        <1>       call   TIME_OF_DAY_1
19370 00005980 C3                <1>       retn
19371                            <1>
19372                            <1> ;--- INT  1A H -- (TIME OF DAY) ------------------------------------------------
19373                            <1> ;       THIS BIOS ROUTINE ALLOWS THE CLOCKS TO BE SET OR READ                :
19374                            <1> ;                                                                            :
19375                            <1> ; PARAMETERS:                                                                :
19376                            <1> ;     (AH) = 00H  READ THE CURRENT SETTING AND RETURN WITH,                  :
```

```
19377                                  <1> ;                   (CX) = HIGH PORTION OF COUNT                    :
19378                                  <1> ;                   (DX) = LOW PORTION OF COUNT                     :
19379                                  <1> ;                   (AL) = 0 TIMER HAS NOT PASSED 24 HOURS SINCE LAST READ :
19380                                  <1> ;                          1 IF ON ANOTHER DAY. (RESET TO ZERO AFTER READ) :
19381                                  <1> ;                                                                  :
19382                                  <1> ;     (AH) = 01H  SET THE CURRENT CLOCK USING,                                  :
19383                                  <1> ;                 (CX) = HIGH PORTION OF COUNT                     :
19384                                  <1> ;                 (DX) = LOW PORTION OF COUNT.                     :
19385                                  <1> ;                                                                  :
19386                                  <1> ;               NOTE: COUNTS OCCUR AT THE RATE OF 1193180/65536 COUNTS/SECOND :
19387                                  <1> ;                           (OR ABOUT 18.2 PER SECOND -- SEE EQUATES)      :
19388                                  <1> ;                                                                  :
19389                                  <1> ;     (AH) = 02H  READ THE REAL TIME CLOCK AND RETURN WITH,                     :
19390                                  <1> ;                   (CH) = HOURS IN BCD (00-23)                    :
19391                                  <1> ;                   (CL) = MINUTES IN BCD (00-59)                           :
19392                                  <1> ;                   (DH) = SECONDS IN BCD (00-59)                           :
19393                                  <1> ;                   (DL) = DAYLIGHT SAVINGS ENABLE (00-01)                  :
19394                                  <1> ;                                                                  :
19395                                  <1> ;     (AH) = 03H  SET THE REAL TIME CLOCK USING,                    :
19396                                  <1> ;                  (CH) = HOURS IN BCD (00-23)                     :
19397                                  <1> ;                  (CL) = MINUTES IN BCD (00-59)                   :
19398                                  <1> ;                  (DH) = SECONDS IN BCD (00-59)                   :
19399                                  <1> ;                  (DL) = 01 IF DAYLIGHT SAVINGS ENABLE OPTION, ELSE 00.   :
19400                                  <1> ;                                                                  :
19401                                  <1> ;             NOTE: (DL) = 00 IF DAYLIGHT SAVINGS TIME ENABLE IS NOT ENABLED. :
19402                                  <1> ;                   (DL) = 01 ENABLES TWO SPECIAL UPDATES THE LAST SUNDAY IN  :
19403                                  <1> ;                APRIL   (1:59:59 --> 3:00:00 AM) AND THE LAST SUNDAY IN      :
19404                                  <1> ;                   OCTOBER (1:59:59 --> 1:00:00 AM) THE FIRST TIME.           :
19405                                  <1> ;                                                                  :
19406                                  <1> ;     (AH) = 04H  READ THE DATE FROM THE REAL TIME CLOCK AND RETURN WITH,     :
19407                                  <1> ;                   (CH) = CENTURY IN BCD (19 OR 20)                        :
19408                                  <1> ;                   (CL) = YEAR IN BCD (00-99)                     :
19409                                  <1> ;                    (DH) = MONTH IN BCD (01-12)                    :
19410                                  <1> ;                    (DL) = DAY IN BCD (01-31).                     :
19411                                  <1> ;                                                                  :
19412                                  <1> ;     (AH) = 05H  SET THE DATE INTO THE REAL TIME CLOCK USING,                :
19413                                  <1> ;                   (CH) = CENTURY IN BCD (19 OR 20)                        :
19414                                  <1> ;                   (CL) = YEAR IN BCD (00-99)                     :
19415                                  <1> ;                   (DH) = MONTH IN BCD (01-12)                    :
19416                                  <1> ;                   (DL) = DAY IN BCD (01-31).                     :
19417                                  <1> ;                                                                  :
19418                                  <1> ;     (AH) = 06H  SET THE ALARM TO INTERRUPT AT SPECIFIED TIME,               :
19419                                  <1> ;                   (CH) = HOURS IN BCD (00-23 (OR FFH))           :
19420                                  <1> ;                   (CL) = MINUTES IN BCD (00-59 (OR FFH))                  :
19421                                  <1> ;                   (DH) = SECONDS IN BCD (00-59 (OR FFH))                  :
19422                                  <1> ;                                                                  :
19423                                  <1> ;     (AH) = 07H  RESET THE ALARM INTERRUPT FUNCTION.                         :
19424                                  <1> ;                                                                  :
19425                                  <1> ; NOTES: FOR ALL RETURNS CY= 0 FOR SUCCESSFUL OPERATION.           :
19426                                  <1> ;        FOR (AH)= 2, 4, 6 - CARRY FLAG SET IF REAL TIME CLOCK NOT OPERATING. :
19427                                  <1> ;        FOR (AH)= 6 - CARRY FLAG SET IF ALARM ALREADY ENABLED.    :
19428                                  <1> ;        FOR THE ALARM FUNCTION (AH = 6) THE USER MUST SUPPLY A ROUTINE AND   :
19429                                  <1> ;         INTERCEPT THE CORRECT ADDRESS IN THE VECTOR TABLE FOR INTERRUPT 4AH.      :
19430                                  <1> ;         USE 0FFH FOR ANY "DO NOT CARE" POSITION FOR INTERVAL INTERRUPTS.    :
19431                                  <1> ;        INTERRUPTS ARE DISABLED DURING DATA MODIFICATION.          :
19432                                  <1> ;        AH & AL ARE RETURNED MODIFIED AND NOT DEFINED EXCEPT WHERE INDICATED.      :
19433                                  <1> ;--------------------------------------------------------------------------
19434                                  <1>
19435                                  <1> ; 15/01/2017
19436                                  <1> ; 14/01/2017
19437                                  <1> ; 07/01/2017
19438                                  <1> ; 02/01/2017
19439                                  <1> ; 29/05/2016
19440                                  <1> ; 29/01/2016
19441                                  <1> ; 17/01/2016 (TRDOS 386 = TRDOS v2.0)
19442                                  <1>
19443                                  <1> ; 29/05/2016
19444                                  <1> ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
19445                                  <1> int35h:  ; Date/Time functions
19446                                  <1>
19447                                  <1> TIME_OF_DAY_1:
19448                                  <1>      ;sti                    ; INTERRUPTS BACK ON
19449                                  <1>      ; 29/05/2016
19450 00005981 80642408FE             <1>      and   byte [esp+8], 11111110b   ; clear carry bit of eflags register
19451                                  <1>      ;
19452 00005986 80FC08                 <1>      cmp   ah, (RTC_TBE-RTC_TB)/4    ; CHECK IF COMMAND IN VALID RANGE (0-7)
19453 00005989 F5                     <1>      cmc                     ; COMPLEMENT CARRY FOR ERROR EXIT
19454                                  <1>      ; (*) jc short TIME_9           ; EXIT WITH CARRY = 1 IF NOT VALID
19455 0000598A 721A                   <1>      jc    short _TIME_9 ; 29/05/2016
19456                                  <1>
19457 0000598C 1E                     <1>      push  ds
19458 0000598D 56                     <1>      push  esi
19459 0000598E 66BE1000               <1>      mov   si, KDATA          ; kernel data segment
19460 00005992 8EDE                   <1>      mov   ds, si
19461                                  <1>
19462                                  <1>      ;;15/01/2017
19463                                  <1>      ; 14/01/2017
19464                                  <1>      ; 02/01/2017
19465                                  <1>      ;;mov  byte [intflg], 35h  ; date & time interrupt
19466                                  <1>      ;sti
19467                                  <1>      ;
19468 00005994 C0E402                 <1>      shl   ah, 2             ; convert function to dword offset
19469 00005997 0FB6F4                 <1>      movzx  esi, ah                ; PLACE INTO ADDRESSING REGISTER
19470                                  <1>      ;cli                    ; NO INTERRUPTS DURING TIME FUNCTIONS
19471 0000599A FF96[AC590000]         <1>      call  [esi+RTC_TB]      ; VECTOR TO FUNCTION REQUESTED WITH CY=0
19472                                  <1>                              ; RETURN WITH CARRY FLAG SET FOR RESULT
19473                                  <1>      ;sti                    ; INTERRUPTS BACK ON
19474 000059A0 B400                   <1>      mov   ah, 0             ; CLEAR (AH) TO ZERO
19475 000059A2 5E                     <1>      pop   esi               ; RECOVER USERS REGISTER
19476 000059A3 1F                     <1>      pop   ds                ; RECOVER USERS SEGMENT SELECTOR
19477                                  <1>
19478                                  <1>      ;;15/01/2017
```

```
19479                                  <1>       ; 02/01/2017
19480                                  <1>       ;;mov  byte [ss:intflg], 0 ; 07/01/2017
19481                                  <1>
19482                                  <1> ;TIME_9:
19483                                  <1>                                     ; RETURN WITH CY= 0 IF NO ERROR
19484                                  <1>       ; (*) 29/05/2016
19485                                  <1>       ; (*) retf 4 ; skip eflags on stack
19486 000059A4 7305                   <1>       jnc    short _TIME_10
19487                                  <1> _TIME_9:
19488                                  <1>       ; 29/05/2016 -set carry flag on stack-
19489                                  <1>       ; [esp] = EIP
19490                                  <1>       ; [esp+4] = CS
19491                                  <1>       ; [esp+8] = E-FLAGS
19492 000059A6 804C240801             <1>       or     byte [esp+8], 1      ; set carry bit of eflags register
19493                                  <1>       ; [esp+12] = ESP (user)
19494                                  <1>       ; [esp+16] = SS (User)
19495                                  <1> _TIME_10:
19496 000059AB CF                     <1>       iretd
19497                                  <1>
19498                                  <1>       ; (*) 29/05/2016 - 'ref 4' intruction causes to stack fault
19499                                  <1>       ; (OUTER-PRIVILEGE-LEVEL)
19500                                  <1>       ; INTEL 80386 PROGRAMMER'S REFERENCE MANUAL 1986
19501                                  <1>       ; // RETF instruction:
19502                                  <1>       ;
19503                                  <1>       ; IF OperandMode=32 THEN
19504                                  <1>       ;    Load CS:EIP from stack;
19505                                  <1>       ;    Set CS RPL to CPL;
19506                                  <1>       ;    Increment eSP by 8 plus the immediate offset if it exists;
19507                                  <1>       ;    Load SS:eSP from stack;
19508                                  <1>       ; ELSE (* OperandMode=16 *)
19509                                  <1>       ;    Load CS:IP from stack;
19510                                  <1>       ;    Set CS RPL to CPL;
19511                                  <1>       ;    Increment eSP by 4 plus the immediate offset if it exists;
19512                                  <1>       ;    Load SS:eSP from stack;
19513                                  <1>       ; FI;
19514                                  <1>       ;
19515                                  <1>       ; //
19516                                  <1>                                     ; ROUTINE VECTOR TABLE (AH)=
19517                                  <1> RTC_TB:
19518 000059AC [CC590000]             <1>       dd     RTC_00             ; 0 = READ CURRENT CLOCK COUNT
19519 000059B0 [DF590000]             <1>       dd     RTC_10             ; 1 = SET CLOCK COUNT
19520 000059B4 [ED590000]             <1>       dd     RTC_20             ; 2 = READ THE REAL TIME CLOCK TIME
19521 000059B8 [1C5A0000]             <1>       dd     RTC_30             ; 3 = SET REAL TIME CLOCK TIME
19522 000059BC [5E5A0000]             <1>       dd     RTC_40             ; 4 = READ THE REAL TIME CLOCK DATE
19523 000059C0 [8B5A0000]             <1>       dd     RTC_50             ; 5 = SET REAL TIME CLOCK DATE
19524 000059C4 [D85A0000]             <1>       dd     RTC_60             ; 6 = SET THE REAL TIME CLOCK ALARM
19525 000059C8 [2B5B0000]             <1>       dd     RTC_70             ; 7 = RESET ALARM
19526                                  <1>
19527                                  <1> RTC_TBE     equ    $
19528                                  <1>
19529                                  <1> RTC_00:                            ; READ TIME COUNT
19530 000059CC A0[4C4E0100]           <1>       mov    al, [TIMER_OFL]           ; GET THE OVERFLOW FLAG
19531 000059D1 C605[4C4E0100]00       <1>       mov    byte [TIMER_OFL], 0 ; AND THEN RESET THE OVERFLOW FLAG
19532 000059D8 8B0D[484E0100]         <1>         mov     ecx, [TIMER_LH]         ; GET COUNT OF TIME
19533 000059DE C3                     <1>       retn
19534                                  <1>
19535                                  <1> RTC_10:                            ; SET TIME COUNT
19536 000059DF 890D[484E0100]         <1>         mov     [TIMER_LH], ecx         ; SET TIME COUNT
19537 000059E5 C605[4C4E0100]00       <1>       mov    byte [TIMER_OFL], 0 ; RESET OVERFLOW FLAG
19538 000059EC C3                     <1>       retn                       ; RETURN WITH NO CARRY
19539                                  <1>
19540                                  <1> RTC_20:                            ; GET RTC TIME
19541 000059ED E8EB010000             <1>       call   UPD_IPR                ; CHECK FOR UPDATE IN PROCESS
19542 000059F2 7227                   <1>       jc     short RTC_29        ; EXIT IF ERROR (CY= 1)
19543                                  <1>
19544 000059F4 B000                   <1>       mov    al, CMOS_SECONDS    ; SET ADDRESS OF SECONDS
19545 000059F6 E8FD010000             <1>       call   CMOS_READ           ; GET SECONDS
19546 000059FB 88C6                   <1>       mov    dh, al              ; SAVE
19547 000059FD B00B                   <1>       mov    al, CMOS_REG_B           ; ADDRESS ALARM REGISTER
19548 000059FF E8F4010000             <1>       call   CMOS_READ           ; READ CURRENT VALUE OF DSE BIT
19549 00005A04 2401                   <1>       and    al, 00000001b       ; MASK FOR VALID DSE BIT
19550 00005A06 88C2                   <1>       mov    dl, al              ; SET [DL] TO ZERO FOR NO DSE BIT
19551 00005A08 B002                   <1>       mov    al, CMOS_MINUTES    ; SET ADDRESS OF MINUTES
19552 00005A0A E8E9010000             <1>       call   CMOS_READ           ; GET MINUTES
19553 00005A0F 88C1                   <1>       mov    cl, al              ; SAVE
19554 00005A11 B004                   <1>         mov     al, CMOS_HOURS          ; SET ADDRESS OF HOURS
19555 00005A13 E8E0010000             <1>       call   CMOS_READ           ; GET HOURS
19556 00005A18 88C5                   <1>       mov    ch, al              ; SAVE
19557 00005A1A F8                     <1>       clc                        ; SET CY= 0
19558                                  <1> RTC_29:
19559 00005A1B C3                     <1>       retn                       ; RETURN WITH RESULT IN CARRY FLAG
19560                                  <1>
19561                                  <1> RTC_30:                            ; SET RTC TIME
19562 00005A1C E8BC010000             <1>       call   UPD_IPR                ; CHECK FOR UPDATE IN PROCESS
19563 00005A21 7305                   <1>       jnc    short RTC_35        ; GO AROUND IF CLOCK OPERATING
19564 00005A23 E817010000             <1>       call   RTC_STA                ; ELSE TRY INITIALIZING CLOCK
19565                                  <1> RTC_35:
19566 00005A28 88F4                   <1>       mov    ah, dh              ; GET TIME BYTE - SECONDS
19567 00005A2A B000                   <1>       mov    al, CMOS_SECONDS    ; ADDRESS SECONDS
19568 00005A2C E8E0010000             <1>       call   CMOS_WRITE          ; UPDATE SECONDS
19569 00005A31 88CC                   <1>       mov    ah, cl              ; GET TIME BYTE - MINUTES
19570 00005A33 B002                   <1>       mov    al, CMOS_MINUTES    ; ADDRESS MINUTES
19571 00005A35 E8D7010000             <1>       call   CMOS_WRITE          ; UPDATE MINUTES
19572 00005A3A 88EC                   <1>       mov    ah, ch              ; GET TIME BYTE - HOURS
19573 00005A3C B004                   <1>       mov    al, CMOS_HOURS           ; ADDRESS HOURS
19574 00005A3E E8CE010000             <1>       call   CMOS_WRITE          ; UPDATE ADDRESS
19575                                  <1>       ;mov   al, CMOS_REG_B           ; ADDRESS ALARM REGISTER
19576                                  <1>       ;mov   ah, al
19577 00005A43 66B80B0B               <1>       mov    ax, CMOS_REG_B * 257
19578 00005A47 E8AC010000             <1>       call   CMOS_READ           ; READ CURRENT TIME
19579 00005A4C 2462                   <1>       and    al, 01100010b       ; MASK FOR VALID BIT POSITIONS
19580 00005A4E 0C02                   <1>       or     al, 00000010b       ; TURN ON 24 HOUR MODE
```

```
19581 00005A50 80E201            <1>       and   dl, 00000001b       ; USE ONLY THE DSE BIT
19582 00005A53 08D0              <1>       or    al, dl              ; GET DAY LIGHT SAVINGS TIME BIT (OSE)
19583 00005A55 86E0              <1>       xchg  ah, al              ; PLACE IN WORK REGISTER AND GET ADDRESS
19584 00005A57 E8B5010000        <1>       call  CMOS_WRITE          ; SET NEW ALARM SITS
19585 00005A5C F8                <1>       clc                       ; SET CY= 0
19586 00005A5D C3                <1>       retn                      ; RETURN WITH CY= 0
19587                            <1>
19588                            <1> RTC_40:                         ; GET RTC DATE
19589 00005A5E E87A010000        <1>       call  UPD_IPR                   ; CHECK FOR UPDATE IN PROCESS
19590 00005A63 7225              <1>       jc    short RTC_49        ; EXIT IF ERROR (CY= 1)
19591                            <1>
19592 00005A65 B007              <1>       mov   al, CMOS_DAY_MONTH  ; ADDRESS DAY OF MONTH
19593 00005A67 E88C010000        <1>       call  CMOS_READ           ; READ DAY OF MONTH
19594 00005A6C 88C2              <1>       mov   dl, al              ; SAVE
19595 00005A6E B008              <1>       mov   al, CMOS_MONTH            ; ADDRESS MONTH
19596 00005A70 E883010000        <1>       call  CMOS_READ           ; READ MONTH
19597 00005A75 88C6              <1>       mov   dh, al              ; SAVE
19598 00005A77 B009              <1>       mov   al, CMOS_YEAR       ; ADDRESS YEAR
19599 00005A79 E87A010000        <1>       call  CMOS_READ           ; READ YEAR
19600 00005A7E 88C1              <1>       mov   cl, al              ; SAVE
19601 00005A80 B032              <1>       mov   al, CMOS_CENTURY    ; ADDRESS CENTURY LOCATION
19602 00005A82 E871010000        <1>       call  CMOS_READ           ; GET CENTURY BYTE
19603 00005A87 88C5              <1>       mov   ch, al              ; SAVE
19604 00005A89 F8                <1>       clc                       ; SET CY=0
19605                            <1> RTC_49:
19606 00005A8A C3                <1>       retn                      ; RETURN WITH RESULTS IN CARRY FLAG
19607                            <1>
19608                            <1> RTC_50:                         ; SET RTC DATE
19609 00005A8B E84D010000        <1>       call  UPD_IPR                   ; CHECK FOR UPDATE IN PROCESS
19610 00005A90 7305              <1>       jnc   short RTC_55        ; GO AROUND IF NO ERROR
19611 00005A92 E8A8000000        <1>       call  RTC_STA                   ; ELSE INITIALIZE CLOCK
19612                            <1> RTC_55:
19613 00005A97 66B80600          <1>       mov   ax, CMOS_DAY_WEEK   ; ADDRESS OF DAY OF WEEK BYTE
19614 00005A9B E871010000        <1>       call  CMOS_WRITE          ; LOAD ZEROS TO DAY OF WEEK
19615 00005AA0 88D4              <1>       mov   ah, dl              ; GET DAY OF MONTH BYTE
19616 00005AA2 B007              <1>       mov   al, CMOS_DAY_MONTH  ; ADDRESS DAY OF MONTH BYTE
19617 00005AA4 E868010000        <1>       call  CMOS_WRITE          ; WRITE OF DAY OF MONTH REGISTER
19618 00005AA9 88F4              <1>       mov   ah, dh              ; GET MONTH
19619 00005AAB B008              <1>       mov   al, CMOS_MONTH            ; ADDRESS MONTH BYTE
19620 00005AAD E85F010000        <1>       call  CMOS_WRITE          ; WRITE MONTH REGISTER
19621 00005AB2 88CC              <1>       mov   ah, cl              ; GET YEAR BYTE
19622 00005AB4 B009              <1>       mov   al, CMOS_YEAR       ; ADDRESS YEAR REGISTER
19623 00005AB6 E856010000        <1>       call  CMOS_WRITE          ; WRITE YEAR REGISTER
19624 00005ABB 88EC              <1>       mov   ah, ch              ; GET CENTURY BYTE
19625 00005ABD B032              <1>       mov   al, CMOS_CENTURY    ; ADDRESS CENTURY BYTE
19626 00005ABF E84D010000        <1>       call  CMOS_WRITE          ; WRITE CENTURY LOCATION
19627                            <1>       ;mov  al, CMOS_REG_B            ; ADDRESS ALARM REGISTER
19628                            <1>       ;mov  ah, al
19629 00005AC4 66B80B0B          <1>       mov   ax, CMOS_REG_B * 257
19630 00005AC8 E82B010000        <1>       call  CMOS_READ           ; READ WIRRENT SETTINGS
19631 00005ACD 247F              <1>       and   al, 07Fh            ; CLEAR 'SET BIT'
19632 00005ACF 86E0              <1>       xchg  ah, al              ; MOVE TO WORK REGISTER
19633 00005AD1 E83B010000        <1>       call  CMOS_WRITE          ; AND START CLOCK UPDATING
19634 00005AD6 F8                <1>       clc                       ; SET CY= 0
19635 00005AD7 C3                <1>       retn                      ; RETURN CY=0
19636                            <1>
19637                            <1> RTC_60:                         ; SET RTC ALARM
19638 00005AD8 B00B              <1>       mov   al, CMOS_REG_B            ; ADDRESS ALARM
19639 00005ADA E819010000        <1>       call  CMOS_READ           ; READ ALARM REGISTER
19640 00005ADF A820              <1>       test  al, 20h                   ; CHECK FOR ALARM ALREADY ENABLED
19641 00005AE1 F9                <1>       stc                       ; SET CARRY IN CASE OF ERROR
19642 00005AE2 7542              <1>       jnz   short RTC_69        ; ERROR EXIT IF ALARM SET
19643 00005AE4 E8F4000000        <1>       call  UPD_IPR                   ; CHECK FOR UPDATE IN PROCESS
19644 00005AE9 7305              <1>       jnc   short RTC_65        ; SKIP INITIALIZATION IF NO ERROR
19645 00005AEB E84F000000        <1>       call  RTC_STA                   ; ELSE INITIALIZE CLOCK
19646                            <1> RTC_65:
19647 00005AF0 88F4              <1>       mov   ah, dh              ; GET SECONDS BYTE
19648 00005AF2 B001              <1>       mov   al, CMOS_SEC_ALARM  ; ADDRESS THE SECONDS ALARM REGISTER
19649 00005AF4 E818010000        <1>       call  CMOS_WRITE          ; INSERT SECONDS
19650 00005AF9 88CC              <1>       mov   ah, cl              ; GET MINUTES PARAMETER
19651 00005AFB B003              <1>       mov   al, CMOS_MIN_ALARM  ; ADDRESS MINUTES ALARM REGISTER
19652 00005AFD E80F010000        <1>       call  CMOS_WRITE          ; INSERT MINUTES
19653 00005B02 88EC              <1>       mov   ah, ch              ; GET HOURS PARAMETER
19654 00005B04 B005              <1>       mov   al, CMOS_HR_ALARM   ; ADDRESS HOUR ALARM REGISTER
19655 00005B06 E806010000        <1>       call  CMOS_WRITE          ; INSERT HOURS
19656 00005B0B E4A1              <1>       in    al, INTB01          ; READ SECOND INTERRUPT MASK REGISTER
19657 00005B0D 24FE              <1>       and   al, 0FEh            ; ENABLE ALARM TIMER BIT (CY= 0)
19658 00005B0F E6A1              <1>       out   INTB01, al          ; WRITE UPDATED MASK
19659                            <1>       ;mov  al, CMOS_REG_B            ; ADDRESS ALARM REGISTER
19660                            <1>       ;mov  ah, al
19661 00005B11 66B80B0B          <1>       mov   ax, CMOS_REG_B * 257
19662 00005B15 E8DE000000        <1>       call  CMOS_READ           ; READ CURRENT ALARM REGISTER
19663 00005B1A 247F              <1>       and   al, 07Fh            ; ENSURE SET BIT TURNED OFF
19664 00005B1C 0C20              <1>       or    al, 20h                   ; TURN ON ALARM ENABLE
19665 00005B1E 86E0              <1>       xchg  ah, al              ; MOVE MASK TO OUTPUT REGISTER
19666 00005B20 E8EC000000        <1>       call  CMOS_WRITE          ; WRITE NEW ALARM MASK
19667 00005B25 F8                <1>       clc                       ; SET CY= 0
19668                            <1> RTC_69:
19669 00005B26 66B80000          <1>       mov   ax, 0               ; CLEAR AX REGISTER
19670 00005B2A C3                <1>       retn                      ; RETURN WITH RESULTS IN CARRY FLAC
19671                            <1>
19672                            <1> RTC_70:                         ; RESET ALARM
19673                            <1>       ;mov  al, CMOS_REG_B            ; ADDRESS ALARM REGISTER
19674                            <1>       ;mov  ah, al
19675 00005B2B 66B80B0B          <1>       mov   ax, CMOS_REG_B * 257      ; ADDRESS ALARM REGISTER (TO BOTH AH,AL)
19676 00005B2F E8C4000000        <1>       call  CMOS_READ           ; READ ALARM REGISTER
19677 00005B34 2457              <1>       and   al, 57h                   ; TURN OFF ALARM ENABLE
19678 00005B36 86E0              <1>       xchg  ah, al              ; SAVE DATA AND RECOVER ADDRESS
19679 00005B38 E8D4000000        <1>       call  CMOS_WRITE          ; RESTORE NEW VALUE
19680 00005B3D F8                <1>       clc                       ; SET CY= 0
19681 00005B3E C3                <1>       retn                      ; RETURN WITH NO CARRY
19682                            <1>
```

```
19683                                  <1> RTC_STA:                ; INITIALIZE REAL TIME CLOCK
19684                                  <1>       ;mov   al, CMOS_REG_A           ; ADDRESS REGISTER A AND LOAD DATA MASK
19685                                  <1>       ;mov   ah, 26h
19686 00005B3F 66B80A26                <1>       mov    ax, (26h*100h)+CMOS_REG_A
19687 00005B43 E8C9000000              <1>       call   CMOS_WRITE          ; INITIALIZE STATUS REGISTER A
19688                                  <1>       ;mov   al, CMOS_REG_B           ; SET "SET BIT" FOR CLOCK INITIALIZATION
19689                                  <1>       ;mov   ah, 82h
19690 00005B48 66B80B82                <1>       mov    ax, (82h*100h)+CMOS_REG_B
19691 00005B4C E8C0000000              <1>       call   CMOS_WRITE          ; AND 24 HOUR MODE TO REGISTER B
19692 00005B51 B00C                    <1>       mov    al, CMOS_REG_C           ; ADDRESS REGISTER C
19693 00005B53 E8A0000000              <1>       call   CMOS_READ           ; READ REGISTER C TO INITIALIZE
19694 00005B58 B00D                    <1>       mov    al, CMOS_REG_D           ; ADDRESS REGISTER D
19695 00005B5A E899000000              <1>       call   CMOS_READ           ; READ REGISTER D TO INITIALIZE
19696 00005B5F C3                      <1>       retn
19697                                  <1>
19698                                  <1> ; 17/01/2016 (TRDOS 386 = TRDOS v2.0)
19699                                  <1>
19700                                  <1> ;--- HARDWARE INT  70 H -- ( IRQ LEVEL  8) -------------------------------------
19701                                  <1> ; ALARM INTERRUPT HANDLER (RTC)                                          :
19702                                  <1> ;       THIS ROUTINE HANDLES THE PERIODIC AND ALARM INTERRUPTS FROM THE CMOS  :
19703                                  <1> ;       TIMER. INPUT FREQUENCY IS 1.024 KHZ OR APPROXIMATELY 1024 INTERRUPTS  :
19704                                  <1> ;       EVERY SECOND FOR THE PERIODIC INTERRUPT. FOR THE ALARM FUNCTION,      :
19705                                  <1> ;       THE INTERRUPT WILL OCCUR AT THE DESIGNATED TIME.              :
19706                                  <1> ;                                                                 :
19707                                  <1> ;       INTERRUPTS ARE ENABLED WHEN THE EVENT OR ALARM FUNCTION IS ACTIVATED. :
19708                                  <1> ;       FOR THE EVENT INTERRUPT, THE HANDLER WILL DECREMENT THE WAIT COUNTER  :
19709                                  <1> ;       AND WHEN IT EXPIRES WILL SET THE DESIGNATED LOCATION TO 80H. FOR      :
19710                                  <1> ;       THE ALARM INTERRUPT. THE USER MUST PROVIDE A ROUTINE TO INTERCEPT     :
19711                                  <1> ;       THE CORRECT ADDRESS FROM THE VECTOR TABLE INVOKED BY INTERRUPT 4AH    :
19712                                  <1> ;       PRIOR TO SETTING THE REAL TIME CLOCK ALARM (INT 1AH, AH= 06H).        :
19713                                  <1> ;-------------------------------------------------------------------------------
19714                                  <1>
19715                                  <1> RTC_A_INT: ; 07/01/2017
19716                                  <1> ;RTC_INT:                       ; ALARM INTERRUPT
19717 00005B60 1E                      <1>       push   ds                 ; LEAVE INTERRUPTS DISABLED
19718 00005B61 50                      <1>       push   eax                ; SAVE REGISTERS
19719 00005B62 57                      <1>       push   edi
19720                                  <1> RTC_I_1:                        ; CHECK FOR SECOND INTERRUPT
19721 00005B63 66B88C8B                <1>       mov    ax, 256*(CMOS_REG_B+NMI)+CMOS_REG_C+NMI ; ALARM AND STATUS
19722 00005B67 E670                    <1>       out    CMOS_PORT, al      ; WRITE ALARM FLAG MASK ADDRESS
19723 00005B69 90                      <1>       nop                       ; I/O DELAY
19724 00005B6A EB00                    <1>       jmp    short $+2
19725 00005B6C E471                    <1>       in     al, CMOS_DATA      ; READ AND RESET INTERRUPT REQUEST FLAGS
19726 00005B6E A860                    <1>       test   al, 01100000b      ; CHECK FOR EITHER INTERRUPT PENDING
19727 00005B70 745D                    <1>       jz     short  RTC_I_9           ; EXIT IF NOT A VALID RTC INTERRUPT
19728                                  <1>
19729 00005B72 86E0                    <1>       xchg   ah, al             ; SAVE FLAGS AND GET ENABLE ADDRESS
19730 00005B74 E670                    <1>       out    CMOS_PORT, al      ; WRITE ALARM ENABLE MASK ADDRESS
19731 00005B76 90                      <1>       nop                       ; I/O DELAY
19732 00005B77 EB00                    <1>       jmp    short $+2
19733 00005B79 E471                    <1>       in     al, CMOS_DATA      ; READ CURRENT ALARM ENABLE MASK
19734 00005B7B 20E0                    <1>       and    al, ah             ; ALLOW ONLY SOURCES THAT ARE ENABLED
19735 00005B7D A840                    <1>       test   al, 01000000b      ; CHECK FOR PERIODIC INTERRUPT
19736 00005B7F 743B                    <1>       jz     short RTC_I_5      ; SKIP IF NOT A PERIODIC INTERRUPT
19737                                  <1>
19738                                  <1> ;-----      DECREMENT WAIT COUNT BY INTERRUPT INTERVAL
19739                                  <1>
19740 00005B81 66BF1000                <1>       mov    di, KDATA          ; kernel data segment
19741 00005B85 8EDF                    <1>       mov    ds, di
19742                                  <1>
19743 00005B87 812D[404E0100]D003-     <1>       sub    dword [RTC_LH], 976 ; DECREMENT COUNT BY 1/1024
19744 00005B8F 0000                    <1>
19745 00005B91 7329                    <1>       jnc    short RTC_I_5      ; SKIP TILL 32 BIT WORD LESS THAN ZERO
19746                                  <1>
19747                                  <1> ;-----      TURN OFF PERIODIC INTERRUPT ENABLE
19748                                  <1>
19749 00005B93 6650                    <1>       push   ax                 ; SAVE INTERRUPT FLAG MASK
19750 00005B95 66B88B8B                <1>       mov    ax, 257*(CMOS_REG_B+NMI) ; INTERRUPT ENABLE REGISTER
19751 00005B99 E670                    <1>       out    CMOS_PORT, al      ; WRITE ADDRESS TO CMOS CLOCK
19752 00005B9B 90                      <1>       nop                       ; I/O DELAY
19753 00005B9C EB00                    <1>       jmp    short $+2
19754 00005B9E E471                    <1>       in     al, CMOS_DATA      ; READ CURRENT ENABLES
19755 00005BA0 24BF                    <1>       and    al, 0BFh           ; TURN OFF PIE
19756 00005BA2 86C4                    <1>       xchg   al, ah             ; GET CMOS ADDRESS AND SAVE VALUE
19757 00005BA4 E670                    <1>       out    CMOS_PORT, al      ; ADDRESS REGISTER B
19758 00005BA6 86C4                    <1>       xchg   al, ah             ; GET NEW INTERRUPT ENABLE MASK
19759 00005BA8 E671                    <1>       out    CMOS_DATA, al      ; SET MASK IN INTERRUPT ENABLE REGISTER
19760 00005BAA C605[444E0100]00        <1>       mov    byte [RTC_WAIT_FLAG], 0   ; SET FUNCTION ACTIVE FLAG OFF
19761 00005BB1 8B3D[454E0100]          <1>       mov    edi, [USER_FLAG]   ; SET UP (DS:DI) TO POINT TO USER FLAG
19762 00005BB7 C60780                  <1>       mov    byte [edi], 80h           ; TURN ON USERS FLAG
19763 00005BBA 6658                    <1>       pop    ax                 ; GET INTERRUPT SOURCE BACK
19764                                  <1> RTC_I_5:
19765 00005BBC A820                    <1>       test   al, 00100000b      ; TEST FOR ALARM INTERRUPT
19766 00005BBE 740D                    <1>       jz     short RTC_I_7      ; SKIP USER INTERRUPT CALL IF NOT ALARM
19767                                  <1>
19768 00005BC0 B00D                    <1>       mov    al, CMOS_REG_D           ; POINT TO DEFAULT READ ONLY REGISTER
19769 00005BC2 E670                    <1>       out    CMOS_PORT, al      ; ENABLE NMI AND CMOS ADDRESS TO DEFAULT
19770 00005BC4 FB                      <1>       sti                       ; INTERRUPTS BACK ON NOW
19771 00005BC5 52                      <1>       push   edx
19772 00005BC6 E8B2970000              <1>       call   INT4Ah             ; TRANSFER TO USER ROUTINE
19773 00005BCB 5A                      <1>       pop    edx
19774 00005BCC FA                      <1>       cli                       ; BLOCK INTERRUPT FOR RETRY
19775                                  <1> RTC_I_7:                        ; RESTART ROUTINE TO HANDLE DELAYED
19776 00005BCD EB94                    <1>       jmp    short RTC_I_1      ;  ENTRY AND SECOND EVENT BEFORE DONE
19777                                  <1>
19778                                  <1> RTC_I_9:                        ; EXIT - NO PENDING INTERRUPTS
19779 00005BCF B00D                    <1>       mov    al, CMOS_REG_D           ; POINT TO DEFAULT READ ONLY REGISTER
19780 00005BD1 E670                    <1>       out    CMOS_PORT, al      ; ENABLE NMI AND CMOS ADDRESS TO DEFAULT
19781 00005BD3 B020                    <1>       mov    al, EOI                  ; END OF INTERRUPT MASK TO 8259 - 2
19782 00005BD5 E6A0                    <1>       out    INTB00, al         ; TO 8259 - 2
19783 00005BD7 E620                    <1>       out    INTA00,      al           ; TO 8259 - 1
19784 00005BD9 5F                      <1>       pop    edi                ; RESTORE REGISTERS
```

```
19785 00005BDA 58                <1>       pop    eax
19786 00005BDB 1F                <1>       pop    ds
19787 00005BDC CF                <1>       iretd                       ; END OF INTERRUPT
19788                            <1>
19789                            <1>
19790                            <1>       ; 29/05/2016 - TRDOS 386 (TRDOS v2.0)
19791                            <1>       ; 22/08/2014 (Retro UNIX 386 v1)
19792                            <1>       ; IBM PC/AT BIOS source code ----- 10/06/85 (bios2.asm)
19793                            <1> UPD_IPR:                         ; WAIT TILL UPDATE NOT IN PROGRESS
19794 00005BDD 51                <1>       push   ecx
19795                            <1>
19796                            <1>       ; 29/05/2016
19797 00005BDE B968110000        <1>       mov    ecx, ((1984+244)*4)/2     ; AWARD BIOS 1999, ATIME.ASM
19798                            <1>                                  ; 'WAITCPU_CK_UD_STAT'
19799                            <1>                                  ; (244Us + 1984Us)
19800                            <1>                                  ; (assume each read takes
19801                            <1>                                  ;  2 microseconds).
19802                            <1>       ;mov   ecx, 65535
19803                            <1>              ;mov cx, 800        ; SET TIMEOUT LOOP COUNT (= 800)
19804                            <1> UPD_10:
19805 00005BE3 B00A              <1>       mov    al, CMOS_REG_A             ; ADDRESS STATUS REGISTER A
19806 00005BE5 FA                <1>       cli                         ; NO TIMER INTERRUPTS DURING UPDATES
19807 00005BE6 E80D000000        <1>       call   CMOS_READ           ; READ UPDATE IN PROCESS FLAG
19808 00005BEB A880              <1>       test   al, 80h                   ; IF UIP BIT IS ON ( CANNOT READ TIME )
19809 00005BED 7406              <1>       jz     short UPD_90        ; EXIT WITH CY= 0 IF CAN READ CLOCK NOW
19810 00005BEF FB                <1>       sti                         ; ALLOW INTERRUPTS WHILE WAITING
19811 00005BF0 E2F1              <1>       loop   UPD_10              ; LOOP TILL READY OR TIMEOUT
19812 00005BF2 31C0              <1>       xor    eax, eax            ; CLEAR RESULTS IF ERROR
19813                            <1>              ; xor ax, ax
19814 00005BF4 F9                <1>       stc                         ; SET CARRY FOR ERROR
19815                            <1> UPD_90:
19816 00005BF5 59                <1>       pop    ecx                 ; RESTORE CALLERS REGISTER
19817 00005BF6 FA                <1>       cli                         ; INTERRUPTS OFF DURING SET
19818 00005BF7 C3                <1>       retn                        ; RETURN WITH CY FLAG SET
19819                            <1>
19820                            <1>
19821                            <1>       ; 29/05/2016 - TRDOS 386 (TRDOS v2.0)
19822                            <1>       ; 22/08/2014 (Retro UNIX 386 v1)
19823                            <1>       ; IBM PC/AT BIOS source code ----- 10/06/85 (test4.asm)
19824                            <1>
19825                            <1> ;--- CMOS_READ ----------------------------------------------------------------
19826                            <1> ;           READ BYTE FROM CMOS_SYSTEM CLOCK CONFIGURATION TABLE       :
19827                            <1> ;                                                                      :
19828                            <1> ; INPUT: (AL)=     CMOS_TABLE ADDRESS TO BE READ                         :
19829                            <1> ;           BIT    7 = 0 FOR NMI ENABLED AND 1 FOR NMI DISABLED ON EXIT    :
19830                            <1> ;           BITS 6-0 = ADDRESS OF TABLE LOCATION TO READ               :
19831                            <1> ;                                                                      :
19832                            <1> ; OUTPUT: (AL)     VALUE AT LOCATION (AL) MOVED INTO (AL). IF BIT 7 OF (AL) WAS   :
19833                            <1> ;           ON THEN NMI LEFT DISABLED, DURING THE CMOS READ BOTH NMI AND   :
19834                            <1> ;           NORMAL INTERRUPTS ARE DISABLED TO PROTECT CMOS DATA INTEGRITY. :
19835                            <1> ;           THE CMOS ADDRESS REGISTER IS POINTED TO A DEFAULT VALUE AND    :
19836                            <1> ;           THE INTERRUPT FLAG RESTORED TO THE ENTRY STATE ON RETURN.      :
19837                            <1> ;           ONLY THE (AL) REGISTER AND THE NMI STATE IS CHANGED.       :
19838                            <1> ;--------------------------------------------------------------------------
19839                            <1>
19840                            <1> CMOS_READ:
19841 00005BF8 9C                <1>       pushf                       ; SAVE INTERRUPT ENABLE STATUS AND FLAGS
19842 00005BF9 D0C0              <1>       rol    al, 1                ; MOVE NMI BIT TO LOW POSITION
19843 00005BFB F9                <1>       stc                         ; FORCE NMI BIT ON IN CARRY FLAG
19844 00005BFC D0D8              <1>       rcr    al, 1                ; HIGH BIT ON TO DISABLE NMI - OLD IN CY
19845 00005BFE FA                <1>       cli                         ; DISABLE INTERRUPTS
19846 00005BFF E670              <1>       out    CMOS_PORT, al        ; ADDRESS LOCATION AND DISABLE NMI
19847                            <1>       ; 29/05/2016
19848                            <1>       ;nop                        ; I/O DELAY
19849 00005C01 E6EB              <1>       out    0ebh,al      ; NEWIODELAY ; AWARD BIOS 1999, ATIME.ASM
19850                            <1>       ;
19851 00005C03 E471              <1>       in     al, CMOS_DATA        ; READ THE REQUESTED CMOS LOCATION
19852 00005C05 6650              <1>       push   ax                   ; SAVE (AH) REGISTER VALUE AND CMOS BYTE
19853                            <1>       ; 15/03/2015 ; IBM PC/XT Model 286 BIOS source code
19854                            <1>                    ; ----- 10/06/85 (test4.asm)
19855 00005C07 B01E              <1>       mov    al, CMOS_SHUT_DOWN*2       ; GET ADDRESS OF DEFAULT LOCATION
19856                            <1>       ;mov   al, CMOS_REG_D*2    ; GET ADDRESS OF DEFAULT LOCATION
19857 00005C09 D0D8              <1>       rcr    al, 1                ; PUT ORIGINAL NMI MASK BIT INTO ADDRESS
19858 00005C0B E670              <1>       out    CMOS_PORT, al        ; SET DEFAULT TO READ ONLY REGISTER
19859 00005C0D 6658              <1>       pop    ax                   ; RESTORE (AH) AND (AL), CMOS BYTE
19860 00005C0F 9D                <1>       popf
19861 00005C10 C3                <1>       retn                        ; RETURN WITH FLAGS RESTORED
19862                            <1>
19863                            <1> ; 17/01/2016 (TRDOS 386 = TRDOS v2.0)
19864                            <1>
19865                            <1> ;--- CMOS_WRITE ---------------------------------------------------------------
19866                            <1> ;     WRITE BYTE TO CMOS SYSTEM CLOCK CONFIGURATION TABLE             :
19867                            <1> ;                                                                     :
19868                            <1> ; INPUT: (AL)=     CMOS TABLE ADDRESS TO BE WRITTEN TO                    :
19869                            <1> ;           BIT    7 = 0 FOR NMI ENABLED AND 1 FOR NMI DISABLED ON EXIT    :
19870                            <1> ;           BITS 6-0 = ADDRESS OF TABLE LOCATION TO WRITE              :
19871                            <1> ;      (AH)= NEW VALUE TO BE PLACED IN THE ADDRESSED TABLE LOCATION          :
19872                            <1> ;                                                                     :
19873                            <1> ; OUTPUT:    VALUE IN (AH) PLACED IN LOCATION (AL) WITH NMI LEFT DISABLED   :
19874                            <1> ;           IF BIT 7 OF (AL) IS ON, DURING THE CMOS UPDATE BOTH NMI AND    :
19875                            <1> ;           NORMAL INTERRUPTS ARE DISABLED TO PROTECT CMOS DATA INTEGRITY. :
19876                            <1> ;           THE CMOS ADDRESS REGISTER IS POINTED TO A DEFAULT VALUE AND    :
19877                            <1> ;           THE INTERRUPT FLAG RESTORED TO THE ENTRY STATE ON RETURN.      :
19878                            <1> ;           ONLY THE CMOS LOCATION AND THE NMI STATE IS CHANGED.       :
19879                            <1> ;--------------------------------------------------------------------------
19880                            <1>
19881                            <1> CMOS_WRITE:                      ; WRITE (AH) TO LOCATION (AL)
19882 00005C11 9C                <1>       pushf                       ; SAVE INTERRUPT ENABLE STATUS AND FLAGS
19883 00005C12 6650              <1>       push   ax                   ; SAVE WORK REGISTER VALUES
19884 00005C14 D0C0              <1>       rol    al, 1                ; MOVE NMI BIT TO LOW POSITION
19885 00005C16 F9                <1>       stc                         ; FORCE NMI BIT ON IN CARRY FLAG
19886 00005C17 D0D8              <1>       rcr    al, 1                ; HIGH BIT ON TO DISABLE NMI - OLD IN CY
```

```
19887 00005C19 FA                <1>       cli                         ; DISABLE INTERRUPTS
19888 00005C1A E670              <1>       out    CMOS_PORT, al        ; ADDRESS LOCATION AND DISABLE NMI
19889 00005C1C 88E0              <1>       mov    al, ah               ; GET THE DATA BYTE TO WRITE
19890 00005C1E E671              <1>       out    CMOS_DATA, al        ; PLACE IN REQUESTED CMOS LOCATION
19891 00005C20 B01E              <1>       mov    al, CMOS_SHUT_DOWN*2       ; GET ADDRESS OF DEFAULT LOCATION
19892                            <1>       ;mov   al, CMOS_REG_D*2     ; GET ADDRESS OF DEFAULT LOCATION
19893 00005C22 D0D8              <1>       rcr    al, 1                ; PUT ORIGINAL NMI MASK BIT INTO ADDRESS
19894 00005C24 E670              <1>       out    CMOS_PORT, al        ; SET DEFAULT TO READ ONLY REGISTER
19895 00005C26 90                <1>       nop                         ; I/O DELAY
19896 00005C27 E471              <1>       in     al, CMOS_DATA        ; OPEN STANDBY LATCH
19897 00005C29 6658              <1>       pop    ax                   ; RESTORE WORK REGISTERS
19898 00005C2B 9D                <1>       popf
19899 00005C2C C3                <1>       retn
19900                            <1>
19901                            <1> ; /// End Of TIMER FUNCTIONS ///
19902
19903 00005C2D 90<rept>              Align 16
19904
19905                                gdt:  ; Global Descriptor Table
19906                                      ; (30/07/2015, conforming cs)
19907                                      ; (26/03/2015)
19908                                      ; (24/03/2015, tss)
19909                                      ; (19/03/2015)
19910                                      ; (29/12/2013)
19911                                      ;
19912 00005C30 0000000000000000            dw 0, 0, 0, 0       ; NULL descriptor
19913                                      ; 18/08/2014
19914                                                  ; 8h kernel code segment, base = 00000000h
19915                                      ;dw 0FFFFh, 0, 9E00h, 00CFh      ; KCODE  ; 30/12/2016
19916 00005C38 FFFF0000009ACF00            dw 0FFFFh, 0, 9A00h, 00CFh; KCODE
19917                                                   ; 10h kernel data segment, base = 00000000h
19918 00005C40 FFFF00000092CF00            dw 0FFFFh, 0, 9200h, 00CFh; KDATA
19919                                                   ; 1Bh user code segment, base address = 400000h ; CORE
19920                                      ;dw 0FBFFh, 0, 0FE40h, 00CFh     ; UCODE  ; 30/12/2016
19921 00005C48 FFFB000040FACF00            dw 0FBFFh, 0, 0FA40h, 00CFh      ; UCODE
19922                                                   ; 23h user data segment, base address = 400000h ; CORE
19923 00005C50 FFFB000040F2CF00            dw 0FBFFh, 0, 0F240h, 00CFh      ; UDATA
19924                                                   ; Task State Segment
19925 00005C58 6700                        dw 0067h ; Limit = 103 ; (104-1, tss size = 104 byte,
19926                                                             ;  no IO permission in ring 3)
19927                                  gdt_tss0:
19928 00005C5A 0000                        dw 0  ; TSS base address, bits 0-15
19929                                  gdt_tss1:
19930 00005C5C 00                          db 0  ; TSS base address, bits 16-23
19931                                                   ; 49h
19932 00005C5D E9                          db 11101001b ; E9h => P=1/DPL=11/0/1/0/B/1 --> B = Task is busy (1)
19933 00005C5E 00                          db 0 ; G/0/0/AVL/LIMIT=0000 ; (Limit bits 16-19 = 0000) (G=0, 1 byte)
19934                                  gdt_tss2:
19935 00005C5F 00                          db 0  ; TSS base address, bits 24-31
19936
19937                                  gdt_end:
19938                                      ;; 9Eh = 1001 1110b (GDT byte 5) P=1/DPL=00/1/TYPE=1110,
19939                                                                ;; Type= 1 (code)/C=1/R=1/A=0
19940                                            ; P= Present, DPL=0=ring 0,  1= user (0= system)
19941                                            ; 1= Code C= Conforming, R= Readable, A = Accessed
19942
19943                                      ;; 9Ah = 1001 1010b (GDT byte 5) P=1/DPL=00/1/TYPE=1010,
19944                                                                ;; Type= 1 (code)/C=0/R=1/A=0
19945                                            ; P= Present, DPL=0=ring 0,  1= user (0= system)
19946                                            ; 1= Code C= non-Conforming, R= Readable, A = Accessed
19947
19948                                      ;; 92h = 1001 0010b (GDT byte 5) P=1/DPL=00/1/TYPE=1010,
19949                                                                ;; Type= 0 (data)/E=0/W=1/A=0
19950                                            ; P= Present, DPL=0=ring 0,  1= user (0= system)
19951                                            ; 0= Data E= Expansion direction (1= down, 0= up)
19952                                            ; W= Writeable, A= Accessed
19953
19954                                      ;; FEh = 1111 1110b (GDT byte 5) P=1/DPL=11/1/TYPE=1110,
19955                                                                ;; Type= 1 (code)/C=1/R=1/A=0
19956                                            ; P= Present, DPL=3=ring 3,  1= user (0= system)
19957                                            ; 1= Code C= Conforming, R= Readable, A = Accessed
19958
19959                                      ;; FAh = 1111 1010b (GDT byte 5) P=1/DPL=11/1/TYPE=1010,
19960                                                                ;; Type= 1 (code)/C=0/R=1/A=0
19961                                            ; P= Present, DPL=3=ring 3,  1= user (0= system)
19962                                            ; 1= Code C= non-Conforming, R= Readable, A = Accessed
19963
19964                                      ;; F2h = 1111 0010b (GDT byte 5) P=1/DPL=11/1/TYPE=0010,
19965                                                                ;; Type= 0 (data)/E=0/W=1/A=0
19966                                            ; P= Present, DPL=3=ring 3,  1= user (0= system)
19967                                            ; 0= Data E= Expansion direction (1= down, 0= up)
19968
19969                                      ;; CFh = 1100 1111b (GDT byte 6) G=1/B=1/0/AVL=0, Limit=1111b (3)
19970
19971                                            ;; Limit = FFFFFh (=> FFFFFh+1= 100000h) // bits 0-15, 48-51 //
19972                                            ;       = 100000h * 1000h (G=1) = 4GB
19973                                            ;; Limit = FFBFFh (=> FFBFFh+1= FFC00h) // bits 0-15, 48-51 //
19974                                            ;       = FFC00h * 1000h (G=1) = 4GB - 4MB
19975                                            ; G= Granularity (1= 4KB), B= Big (32 bit),
19976                                            ; AVL= Available to programmers
19977
19978                                  gdtd:
19979 00005C60 2F00                        dw gdt_end - gdt - 1    ; Limit (size)
19980 00005C62 [305C0000]                  dd gdt                  ; Address of the GDT
19981
19982                                      ; 20/08/2014
19983                                  idtd:
19984 00005C66 7F02                        dw idt_end - idt - 1    ; Limit (size)
19985 00005C68 [E04A0100]                  dd idt                  ; Address of the IDT
19986
19987                                  ; 20/02/2017
19988                                  ;;; 11/03/2015
```

```
19989                                   %include 'diskdata.s'     ; DISK (BIOS) DATA (initialized)
19990                                <1> ; ********************************************************************************
19991                                <1> ; TRDOS386.ASM (TRDOS 386 Kernel) - v2.0.0 - diskdata.s
19992                                <1> ; --------------------------------------------------------------------------------
19993                                <1> ; Last Update: 24/01/2016
19994                                <1> ; --------------------------------------------------------------------------------
19995                                <1> ; Beginning: 24/01/2016
19996                                <1> ; --------------------------------------------------------------------------------
19997                                <1> ; Assembler: NASM version 2.11 (trdos386.s)
19998                                <1> ; --------------------------------------------------------------------------------
19999                                <1> ; Turkish Rational DOS
20000                                <1> ; Operating System Project v2.0 by ERDOGAN TAN (Beginning: 04/01/2016)
20001                                <1> ;
20002                                <1> ; Derived from 'Retro UNIX 386 Kernel - v0.2.1.0' source code by Erdogan Tan
20003                                <1> ; diskdata.inc (11/03/2015)
20004                                <1> ;
20005                                <1> ; Derived from 'IBM PC-XT-286' BIOS source code (1986)
20006                                <1> ; ********************************************************************************
20007                                <1>
20008                                <1> ; Retro UNIX 386 v1 Kernel - DISKDATA.INC
20009                                <1> ; Last Modification: 11/03/2015
20010                                <1> ;     (Initialized Disk Parameters Data section for 'DISKIO.INC')
20011                                <1> ;
20012                                <1>
20013                                <1> ;---------------------------------------
20014                                <1> ;     80286 INTERRUPT LOCATIONS :
20015                                <1> ;     REFERENCED BY POST & BIOS :
20016                                <1> ;---------------------------------------
20017                                <1>
20018 00005C6C [CF5C0000]            <1> DISK_POINTER:      dd     MD_TBL6            ; Pointer to Diskette Parameter Table
20019                                <1>
20020                                <1> ; IBM PC-XT Model 286 source code ORGS.ASM (06/10/85) - 14/12/2014
20021                                <1> ;--------------------------------------------------------------
20022                                <1> ; DISK_BASE                                          :
20023                                <1> ;     THIS IS THE SET OF PARAMETERS REQUIRED FOR          :
20024                                <1> ;     DISKETTE OPERATION. THEY ARE POINTED AT BY THE             :
20025                                <1> ;     DATA VARIABLE @DISK_POINTER. TO MODIFY THE PARAMETERS,     :
20026                                <1> ;     BUILD ANOTHER PARAMETER BLOCK AND POINT AT IT       :
20027                                <1> ;--------------------------------------------------------------
20028                                <1>
20029                                <1> ;DISK_BASE:
20030                                <1> ;     DB     11011111B    ; SRT=D, HD UNLOAD=0F - 1ST SPECIFY BYTE
20031                                <1> ;     DB     2            ; HD LOAD=1, MODE=DMA - 2ND SPECIFY BYTE
20032                                <1> ;     DB     MOTOR_WAIT   ; WAIT TIME AFTER OPERATION TILL MOTOR OFF
20033                                <1> ;     DB     2            ; 512 BYTES/SECTOR
20034                                <1> ;     ;DB    15           ; EOT (LAST SECTOR ON TRACK)
20035                                <1> ;     db     18           ; (EOT for 1.44MB diskette)
20036                                <1> ;     DB     01BH         ; GAP LENGTH
20037                                <1> ;     DB     0FFH         ; DTL
20038                                <1> ;     ;DB    054H         ; GAP LENGTH FOR FORMAT
20039                                <1> ;     db     06ch         ; (for 1.44MB dsikette)
20040                                <1> ;     DB     0F6H         ; FILL BYTE FOR FORMAT
20041                                <1> ;     DB     15           ; HEAD SETTLE TIME (MILLISECONDS)
20042                                <1> ;     DB     8            ; MOTOR START TIME (1/8 SECONDS)
20043                                <1>
20044                                <1> ;---------------------------------------
20045                                <1> ;     ROM BIOS DATA AREAS       :
20046                                <1> ;---------------------------------------
20047                                <1>
20048                                <1> ;DATA       SEGMENT AT 40H            ; ADDRESS= 0040:0000
20049                                <1>
20050                                <1> ;@EQUIP_FLAG DW    ?            ; INSTALLED HARDWARE FLAGS
20051                                <1>
20052                                <1> ;---------------------------------------
20053                                <1> ;     DISKETTE DATA AREAS       :
20054                                <1> ;---------------------------------------
20055                                <1>
20056                                <1> ;@SEEK_STATUS      DB     ?            ; DRIVE RECALIBRATION STATUS
20057                                <1> ;                                     ; BIT 3-0 = DRIVE 3-0 RECALIBRATION
20058                                <1> ;                                     ; BEFORE NEXT SEEK IF BIT IS = 0
20059                                <1> ;@MOTOR_STATUS     DB     ?            ; MOTOR STATUS
20060                                <1> ;                                     ; BIT 3-0 = DRIVE 3-0 CURRENTLY RUNNING
20061                                <1> ;                                     ; BIT 7 = CURRENT OPERATION IS A WRITE
20062                                <1> ;@MOTOR_COUNT      DB     ?            ; TIME OUT COUNTER FOR MOTOR(S) TURN OFF
20063                                <1> ;@DSKETTE_STATUS DB       ?            ; RETURN CODE STATUS BYTE
20064                                <1> ;                                     ; CMD_BLOCK  IN STACK FOR DISK OPERATION
20065                                <1> ;@NEC_STATUS DB     7 DUP(?)     ; STATUS BYTES FROM DISKETTE OPERATION
20066                                <1>
20067                                <1> ;---------------------------------------
20068                                <1> ;     POST AND BIOS WORK DATA AREA     :
20069                                <1> ;---------------------------------------
20070                                <1>
20071                                <1> ;@INTR_FLAG  DB    ?            ; FLAG INDICATING AN INTERRUPT HAPPENED
20072                                <1>
20073                                <1> ;---------------------------------------
20074                                <1> ;     TIMER DATA AREA           :
20075                                <1> ;---------------------------------------
20076                                <1>
20077                                <1> ; 17/12/2014  (IRQ 0 - INT 08H)
20078                                <1> ;TIMER_LOW   equ    46Ch         ; Timer ticks (counter)  @ 40h:006Ch
20079                                <1> ;TIMER_HIGH  equ    46Eh         ; (18.2 timer ticks per second)
20080                                <1> ;TIMER_OFL   equ    470h         ; Timer - 24 hours flag  @ 40h:0070h
20081                                <1>
20082                                <1> ;---------------------------------------
20083                                <1> ;     ADDITIONAL MEDIA DATA            :
20084                                <1> ;---------------------------------------
20085                                <1>
20086                                <1> ;@LASTRATE   DB     ?            ; LAST DISKETTE DATA RATE SELECTED
20087                                <1> ;@DSK_STATE  DB     ?            ; DRIVE 0 MEDIA STATE
20088                                <1> ;            DB     ?            ; DRIVE 1 MEDIA STATE
20089                                <1> ;            DB     ?            ; DRIVE 0 OPERATION START STATE
20090                                <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:
00005C70 01             <1>              DB     01           ;DRIVE TYPE, MEDIA TABLE
                        <1>                 ;DW      MD_TBL1
00005C71 [8E5C0000]     <1>              dd     MD_TBL1
00005C75 82             <1>              DB     02+BIT7ON
                        <1>              ;DW      MD_TBL2
00005C76 [9B5C0000]     <1>                 dd      MD_TBL2
00005C7A 02             <1> DR_DEFAULT:  DB     02
                        <1>                 ;DW      MD_TBL3
00005C7B [A85C0000]     <1>              dd      MD_TBL3
00005C7F 03             <1>              DB     03
                        <1>                 ;DW      MD_TBL4
00005C80 [B55C0000]     <1>              dd      MD_TBL4
00005C84 84             <1>              DB     04+BIT7ON
                        <1>                 ;DW      MD_TBL5
00005C85 [C25C0000]     <1>              dd      MD_TBL5
00005C89 04             <1>              DB     04
                        <1>                 ;DW      MD_TBL6
00005C8A [CF5C0000]     <1>              dd      MD_TBL6
                        <1> DR_TYPE_E       equ $                   ; END OF TABLE
                        <1> ;DR_CNT            EQU    (DR_TYPE_E-DR_TYPE)/3
                        <1> DR_CNT             equ    (DR_TYPE_E-DR_TYPE)/5
                        <1> ;--------------------------------------------------------
                        <1> ;     MEDIA/DRIVE PARAMETER TABLES            :
                        <1> ;--------------------------------------------------------
                        <1> ;--------------------------------------------------------
                        <1> ;     360 KB MEDIA IN 360 KB DRIVE            :
                        <1> ;--------------------------------------------------------
                        <1> MD_TBL1:
00005C8E DF             <1>       DB     11011111B    ; SRT=D, HD UNLOAD=0F - 1ST SPECIFY BYTE
00005C8F 02             <1>       DB     2            ; HD LOAD=1, MODE=DMA - 2ND SPECIFY BYTE
00005C90 25             <1>       DB     MOTOR_WAIT   ; WAIT TIME AFTER OPERATION TILL MOTOR OFF
00005C91 02             <1>       DB     2            ; 512 BYTES/SECTOR
00005C92 09             <1>       DB     09           ; EOT (LAST SECTOR ON TRACK)
00005C93 2A             <1>       DB     02AH         ; GAP LENGTH
00005C94 FF             <1>       DB     0FFH         ; DTL
00005C95 50             <1>       DB     050H         ; GAP LENGTH FOR FORMAT
00005C96 F6             <1>       DB     0F6H         ; FILL BYTE FOR FORMAT
00005C97 0F             <1>       DB     15           ; HEAD SETTLE TIME (MILLISECONDS)
00005C98 08             <1>       DB     8            ; MOTOR START TIME (1/8 SECONDS)
00005C99 27             <1>       DB     39           ; MAX. TRACK NUMBER
00005C9A 80             <1>       DB     RATE_250     ; DATA TRANSFER RATE
                        <1> ;--------------------------------------------------------
                        <1> ;     360 KB MEDIA IN 1.2 MB DRIVE            :
                        <1> ;--------------------------------------------------------
                        <1> MD_TBL2:
00005C9B DF             <1>       DB     11011111B    ; SRT=D, HD UNLOAD=0F - 1ST SPECIFY BYTE
00005C9C 02             <1>       DB     2            ; HD LOAD=1, MODE=DMA - 2ND SPECIFY BYTE
00005C9D 25             <1>       DB     MOTOR_WAIT   ; WAIT TIME AFTER OPERATION TILL MOTOR OFF
00005C9E 02             <1>       DB     2            ; 512 BYTES/SECTOR
00005C9F 09             <1>       DB     09           ; EOT (LAST SECTOR ON TRACK)
00005CA0 2A             <1>       DB     02AH         ; GAP LENGTH
00005CA1 FF             <1>       DB     0FFH         ; DTL
00005CA2 50             <1>       DB     050H         ; GAP LENGTH FOR FORMAT
00005CA3 F6             <1>       DB     0F6H         ; FILL BYTE FOR FORMAT
00005CA4 0F             <1>       DB     15           ; HEAD SETTLE TIME (MILLISECONDS)
00005CA5 08             <1>       DB     8            ; MOTOR START TIME (1/8 SECONDS)
00005CA6 27             <1>       DB     39           ; MAX. TRACK NUMBER
00005CA7 40             <1>       DB     RATE_300     ; DATA TRANSFER RATE
                        <1> ;--------------------------------------------------------
                        <1> ;     1.2 MB MEDIA IN 1.2 MB DRIVE            :
                        <1> ;--------------------------------------------------------
                        <1> MD_TBL3:
00005CA8 DF             <1>       DB     11011111B    ; SRT=D, HD UNLOAD=0F - 1ST SPECIFY BYTE
00005CA9 02             <1>       DB     2            ; HD LOAD=1, MODE=DMA - 2ND SPECIFY BYTE
00005CAA 25             <1>       DB     MOTOR_WAIT   ; WAIT TIME AFTER OPERATION TILL MOTOR OFF
00005CAB 02             <1>       DB     2            ; 512 BYTES/SECTOR
00005CAC 0F             <1>       DB     15           ; EOT (LAST SECTOR ON TRACK)
00005CAD 1B             <1>       DB     01BH         ; GAP LENGTH
00005CAE FF             <1>       DB     0FFH         ; DTL
00005CAF 54             <1>       DB     054H         ; GAP LENGTH FOR FORMAT
00005CB0 F6             <1>       DB     0F6H         ; FILL BYTE FOR FORMAT
00005CB1 0F             <1>       DB     15           ; HEAD SETTLE TIME (MILLISECONDS)
00005CB2 08             <1>       DB     8            ; MOTOR START TIME (1/8 SECONDS)
00005CB3 4F             <1>       DB     79           ; MAX. TRACK NUMBER
00005CB4 00             <1>       DB     RATE_500     ; DATA TRANSFER RATE
                        <1> ;--------------------------------------------------------
                        <1> ;     720 KB MEDIA IN 720 KB DRIVE              :
                        <1> ;--------------------------------------------------------
                        <1> MD_TBL4:
00005CB5 DF             <1>       DB     11011111B    ; SRT=D, HD UNLOAD=0F - 1ST SPECIFY BYTE
00005CB6 02             <1>       DB     2            ; HD LOAD=1, MODE=DMA - 2ND SPECIFY BYTE
00005CB7 25             <1>       DB     MOTOR_WAIT   ; WAIT TIME AFTER OPERATION TILL MOTOR OFF
00005CB8 02             <1>       DB     2            ; 512 BYTES/SECTOR
00005CB9 09             <1>       DB     09           ; EOT (LAST SECTOR ON TRACK)
00005CBA 2A             <1>       DB     02AH         ; GAP LENGTH
00005CBB FF             <1>       DB     0FFH         ; DTL
00005CBC 50             <1>       DB     050H         ; GAP LENGTH FOR FORMAT
00005CBD F6             <1>       DB     0F6H         ; FILL BYTE FOR FORMAT
00005CBE 0F             <1>       DB     15           ; HEAD SETTLE TIME (MILLISECONDS)
00005CBF 08             <1>       DB     8            ; MOTOR START TIME (1/8 SECONDS)
00005CC0 4F             <1>       DB     79           ; MAX. TRACK NUMBER
00005CC1 80             <1>       DB     RATE_250     ; DATA TRANSFER RATE
```

```
20193                                  <1> ;--------------------------------------------------
20194                                  <1> ;     720 KB MEDIA IN 1.44 MB DRIVE                :
20195                                  <1> ;--------------------------------------------------
20196                                  <1> MD_TBL5:
20197 00005CC2 DF                      <1>       DB    11011111B    ; SRT=D, HD UNLOAD=0F - 1ST SPECIFY BYTE
20198 00005CC3 02                      <1>       DB    2            ; HD LOAD=1, MODE=DMA - 2ND SPECIFY BYTE
20199 00005CC4 25                      <1>       DB    MOTOR_WAIT   ; WAIT TIME AFTER OPERATION TILL MOTOR OFF
20200 00005CC5 02                      <1>       DB    2            ; 512 BYTES/SECTOR
20201 00005CC6 09                      <1>       DB    09           ; EOT (LAST SECTOR ON TRACK)
20202 00005CC7 2A                      <1>       DB    02AH         ; GAP LENGTH
20203 00005CC8 FF                      <1>       DB    0FFH         ; DTL
20204 00005CC9 50                      <1>       DB    050H         ; GAP LENGTH FOR FORMAT
20205 00005CCA F6                      <1>       DB    0F6H         ; FILL BYTE FOR FORMAT
20206 00005CCB 0F                      <1>       DB    15           ; HEAD SETTLE TIME (MILLISECONDS)
20207 00005CCC 08                      <1>       DB    8            ; MOTOR START TIME (1/8 SECONDS)
20208 00005CCD 4F                      <1>       DB    79           ; MAX. TRACK NUMBER
20209 00005CCE 80                      <1>       DB    RATE_250     ; DATA TRANSFER RATE
20210                                  <1> ;--------------------------------------------------
20211                                  <1> ;     1.44 MB MEDIA IN 1.44 MB DRIVE               :
20212                                  <1> ;--------------------------------------------------
20213                                  <1> MD_TBL6:
20214 00005CCF AF                      <1>       DB    10101111B    ; SRT=A, HD UNLOAD=0F - 1ST SPECIFY BYTE
20215 00005CD0 02                      <1>       DB    2            ; HD LOAD=1, MODE=DMA - 2ND SPECIFY BYTE
20216 00005CD1 25                      <1>       DB    MOTOR_WAIT   ; WAIT TIME AFTER OPERATION TILL MOTOR OFF
20217 00005CD2 02                      <1>       DB    2            ; 512 BYTES/SECTOR
20218 00005CD3 12                      <1>       DB    18           ; EOT (LAST SECTOR ON TRACK)
20219 00005CD4 1B                      <1>       DB    01BH         ; GAP LENGTH
20220 00005CD5 FF                      <1>       DB    0FFH         ; DTL
20221 00005CD6 6C                      <1>       DB    06CH         ; GAP LENGTH FOR FORMAT
20222 00005CD7 F6                      <1>       DB    0F6H         ; FILL BYTE FOR FORMAT
20223 00005CD8 0F                      <1>       DB    15           ; HEAD SETTLE TIME (MILLISECONDS)
20224 00005CD9 08                      <1>       DB    8            ; MOTOR START TIME (1/8 SECONDS)
20225 00005CDA 4F                      <1>       DB    79           ; MAX. TRACK NUMBER
20226 00005CDB 00                      <1>       DB    RATE_500     ; DATA TRANSFER RATE
20227                                  <1>
20228                                  <1>
20229                                  <1> ; << diskette.inc >>
20230                                  <1> ; +++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
20231                                  <1> ;
20232                                  <1> ;---------------------------------------
20233                                  <1> ;     ROM BIOS DATA AREAS       :
20234                                  <1> ;---------------------------------------
20235                                  <1>
20236                                  <1> ;DATA       SEGMENT AT 40H            ; ADDRESS= 0040:0000
20237                                  <1>
20238                                  <1> ;---------------------------------------
20239                                  <1> ;     FIXED DISK DATA AREAS           :
20240                                  <1> ;---------------------------------------
20241                                  <1>
20242                                  <1> ;DISK_STATUS1:     DB     0           ; FIXED DISK STATUS
20243                                  <1> ;HF_NUM:           DB     0           ; COUNT OF FIXED DISK DRIVES
20244                                  <1> ;CONTROL_BYTE:     DB     0           ; HEAD CONTROL BYTE
20245                                  <1> ;@PORT_OFF   DB    ?            ;  RESERVED (PORT OFFSET)
20246                                  <1>
20247                                  <1> ;---------------------------------------
20248                                  <1> ;     ADDITIONAL MEDIA DATA           :
20249                                  <1> ;---------------------------------------
20250                                  <1>
20251                                  <1> ;@LASTRATE   DB     ?           ; LAST DISKETTE DATA RATE SELECTED
20252                                  <1> ;HF_STATUS   DB     0           ; STATUS REGISTER
20253                                  <1> ;HF_ERROR    DB     0           ; ERROR REGISTER
20254                                  <1> ;HF_INT_FLAG DB     0           ; FIXED DISK INTERRUPT FLAG
20255                                  <1> ;HF_CNTRL    DB     0           ; COMBO FIXED DISK/DISKETTE CARD BIT 0=1
20256                                  <1> ;@DSK_STATE  DB     ?           ; DRIVE 0 MEDIA STATE
20257                                  <1> ;            DB     ?           ; DRIVE 1 MEDIA STATE
20258                                  <1> ;            DB     ?           ; DRIVE 0 OPERATION START STATE
20259                                  <1> ;            DB     ?           ; DRIVE 1 OPERATION START STATE
20260                                  <1> ;@DSK_TRK    DB     ?           ; DRIVE 0 PRESENT CYLINDER
20261                                  <1> ;            DB     ?           ; DRIVE 1 PRESENT CYLINDER
20262                                  <1>
20263                                  <1> ;DATA       ENDS                ; END OF BIOS DATA SEGMENT
20264                                  <1> ;
20265                                  <1> ; +++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
20266                                  <1>
20267                                  <1> ERR_TBL:
20268 00005CDC E0                      <1>       db    NO_ERR
20269 00005CDD 024001BB                <1>       db    BAD_ADDR_MARK,BAD_SEEK,BAD_CMD,UNDEF_ERR
20270 00005CE1 04BB100A                <1>       db    RECORD_NOT_FND,UNDEF_ERR,BAD_ECC,BAD_SECTOR
20271                                  <1>
20272                                  <1> ; 17/12/2014 (mov ax, [cfd])
20273                                  <1> ; 11/12/2014
20274 00005CE5 00                      <1> cfd:         db 0               ; current floppy drive (for GET_PARM)
20275                                  <1> ; 17/12/2014                    ; instead of 'DISK_POINTER'
20276 00005CE6 01                      <1> pfd:         db 1               ; previous floppy drive (for GET_PARM)
20277                                  <1>                                 ; (initial value of 'pfd
20278                                  <1>                                 ; must be different then 'cfd' value
20279                                  <1>                                 ; to force updating/initializing
20280                                  <1>                                 ; current drive parameters)
20281 00005CE7 90                      <1> align 2
20282                                  <1>
20283 00005CE8 F001                    <1> HF_PORT:     dw     1F0h  ; Default = 1F0h
20284                                  <1>                            ; (170h)
20285 00005CEA F603                    <1> HF_REG_PORT: dw     3F6h  ; HF_PORT + 206h
20286                                  <1>
20287                                  <1> ; 05/01/2015
20288 00005CEC 00                      <1> hf_m_s:         db      0     ; (0 = Master, 1 = Slave)
20289                                  <1>
20290                                  <1> ; ************************************************************************
20291
20292 00005CED 90                          Align 2
20293
20294                                      ; 04/11/2014 (Retro UNIX 386 v1)
```

```
20295 00005CEE 0000                    mem_1m_1k:   dw 0  ; Number of contiguous KB between
20296                                                     ; 1 and 16 MB, max. 3C00h = 15 MB.
20297 00005CF0 0000                    mem_16m_64k: dw 0  ; Number of contiguous 64 KB blocks
20298                                                  ; between 16 MB and 4 GB.
20299
20300                                  ; 12/11/2014 (Retro UNIX 386 v1)
20301 00005CF2 00                      boot_drv:    db 0 ; boot drive number (physical)
20302                                  ; 24/11/2014
20303 00005CF3 00                      drv:       db 0
20304 00005CF4 00                      last_drv:    db 0 ; last hdd
20305 00005CF5 00                      hdc:         db 0  ; number of hard disk drives
20306                                                    ; (present/detected)
20307
20308                                  ; 24/11/2014 (Retro UNIX 386 v1)
20309                                  ; Physical drive type & flags
20310 00005CF6 00                      fd0_type:    db 0  ; floppy drive type
20311 00005CF7 00                      fd1_type:    db 0  ; 4 = 1.44 Mb, 80 track, 3.5" (18 spt)
20312                                                    ; 6 = 2.88 Mb, 80 track, 3.5" (36 spt)
20313                                                    ; 3 = 720 Kb, 80 track, 3.5" (9 spt)
20314                                                    ; 2 = 1.2 Mb, 80 track, 5.25" (15 spt)
20315                                                    ; 1 = 360 Kb, 40 track, 5.25" (9 spt)
20316 00005CF8 00                      hd0_type:    db 0  ; EDD status for hd0 (bit 7 = present flag)
20317 00005CF9 00                      hd1_type:    db 0  ; EDD status for hd1 (bit 7 = present flag)
20318 00005CFA 00                      hd2_type:    db 0  ; EDD status for hd2 (bit 7 = present flag)
20319 00005CFB 00                      hd3_type:    db 0  ; EDD status for hd3 (bit 7 = present flag)
20320                                                    ; bit 0 - Fixed disk access subset supported
20321                                                    ; bit 1 - Drive locking and ejecting
20322                                                    ; bit 2 - Enhanced disk drive support
20323                                                    ; bit 3 = Reserved (64 bit EDD support)
20324                                                    ; (If bit 0 is '1' Retro UNIX 386 v1
20325                                                    ; will interpret it as 'LBA ready'!)
20326
20327                                  ; 11/03/2015 - 10/07/2015
20328 00005CFC 000000000000000000-     drv.cylinders: dw 0,0,0,0,0,0,0,0
20329 00005D05 0000000000
20330 00005D0A 000000000000000000-     drv.heads:     dw 0,0,0,0,0,0,0,0
20331 00005D13 0000000000
20332 00005D18 000000000000000000-     drv.spt:       dw 0,0,0,0,0,0,0,0
20333 00005D21 0000000000
20334 00005D26 000000000000000000-     drv.size:      dd 0,0,0,0,0,0,0,0
20335 00005D2F 000000000000000000-
20336 00005D38 000000000000000000-
20337 00005D41 00
20338 00005D42 00000000000000          drv.status:    db 0,0,0,0,0,0,0,0
20339 00005D49 00000000000000          drv.error:     db 0,0,0,0,0,0,0,0
20340
20341                                  Align 2
20342
20343                                  ;;; 11/03/2015
20344                                  %include 'kybdata.s'      ; KEYBOARD (BIOS) DATA
20345                                <1> ; ****************************************************************************
20346                                <1> ; TRDOS386.ASM (TRDOS 386 Kernel) - v2.0.0 - kybdata.s
20347                                <1> ; ----------------------------------------------------------------------------
20348                                <1> ; Last Update: 17/01/2016
20349                                <1> ; ----------------------------------------------------------------------------
20350                                <1> ; Beginning: 17/01/2016
20351                                <1> ; ----------------------------------------------------------------------------
20352                                <1> ; Assembler: NASM version 2.11 (trdos386.s)
20353                                <1> ; ----------------------------------------------------------------------------
20354                                <1> ; Turkish Rational DOS
20355                                <1> ; Operating System Project v2.0 by ERDOGAN TAN (Beginning: 04/01/2016)
20356                                <1> ;
20357                                <1> ; Derived from 'Retro UNIX 386 Kernel - v0.2.1.0' source code by Erdogan Tan
20358                                <1> ; kybdata.inc (11/03/2015)
20359                                <1> ;
20360                                <1> ; Derived from 'IBM PC-XT-286' BIOS source code (1986)
20361                                <1> ; ****************************************************************************
20362                                <1>
20363                                <1> ; Retro UNIX 386 v1 Kernel - KYBDATA.INC
20364                                <1> ; Last Modification: 11/03/2015
20365                                <1> ;            (Data Section for 'KEYBOARD.INC')
20366                                <1> ;
20367                                <1> ; ///////// KEYBOARD DATA ///////////////
20368                                <1>
20369                                <1> ; 05/12/2014
20370                                <1> ; 04/12/2014 (derived from pc-xt-286 bios source code -1986-)
20371                                <1> ; 03/06/86  KEYBOARD BIOS
20372                                <1>
20373                                <1> ;--------------------------------------------------------------------------------
20374                                <1> ;     KEY IDENTIFICATION SCAN TABLES
20375                                <1> ;--------------------------------------------------------------------------------
20376                                <1>
20377                                <1> ;-----       TABLES FOR ALT CASE ------------
20378                                <1> ;-----       ALT-INPUT-TABLE
20379 00005D50 524F50514B            <1> K30:  db     82,79,80,81,75
20380 00005D55 4C4D474849            <1>       db     76,77,71,72,73            ; 10 NUMBER ON KEYPAD
20381                                <1> ;-----       SUPER-SHIFT-TABLE
20382 00005D5A 101112131415          <1>       db     16,17,18,19,20,21   ; A-Z TYPEWRITER CHARS
20383 00005D60 161718191E1F          <1>       db     22,23,24,25,30,31
20384 00005D66 202122232425          <1>       db     32,33,34,35,36,37
20385 00005D6C 262C2D2E2F30          <1>       db     38,44,45,46,47,48
20386 00005D72 3132                  <1>       db     49,50
20387                                <1>
20388                                <1> ;-----       TABLE OF SHIFT KEYS AND MASK VALUES
20389                                <1> ;-----       KEY_TABLE
20390 00005D74 52                    <1> _K6:    db      INS_KEY                 ; INSERT KEY
20391 00005D75 3A4546381D            <1>       db     CAPS_KEY,NUM_KEY,SCROLL_KEY,ALT_KEY,CTL_KEY
20392 00005D7A 2A36                  <1>         db      LEFT_KEY,RIGHT_KEY
20393                                <1> _K6L    equ     $-_K6
20394                                <1>
20395                                <1> ;-----       MASK_TABLE
20396 00005D7C 80                    <1> _K7:    db      INS_SHIFT               ; INSERT MODE SHIFT
```

```
20397 00005D7D 4020100804          <1>       db    CAPS_SHIFT,NUM_SHIFT,SCROLL_SHIFT,ALT_SHIFT,CTL_SHIFT
20398 00005D82 0201                <1>       db    LEFT_SHIFT,RIGHT_SHIFT
20399                              <1>
20400                              <1> ;-----      TABLES FOR CTRL CASE             ;---- CHARACTERS ------
20401 00005D84 1BFF00FFFFFF        <1> _K8:  db    27,-1,0,-1,-1,-1    ; Esc, 1, 2, 3, 4, 5
20402 00005D8A 1EFFFFFFFF1F        <1>       db    30,-1,-1,-1,-1,31   ; 6, 7, 8, 9, 0, -
20403 00005D90 FF7FFF111705        <1>       db    -1,127,-1,17,23,5   ; =, Bksp, Tab, Q, W, E
20404 00005D96 12141915090F        <1>       db    18,20,25,21,9,15    ; R, T, Y, U, I, O
20405 00005D9C 101B1D0AFF01        <1>       db    16,27,29,10,-1,1    ; P, [, ], Enter, Ctrl, A
20406 00005DA2 13040607080A        <1>       db    19,4,6,7,8,10       ; S, D, F, G, H, J
20407 00005DA8 0B0CFFFFFFFF        <1>       db    11,12,-1,-1,-1,-1   ; K, L, :, ', `, LShift
20408 00005DAE 1C1A18031602        <1>       db    28,26,24,3,22,2           ; Bkslash, Z, X, C, V, B
20409 00005DB4 0E0DFFFFFFFF        <1>       db    14,13,-1,-1,-1,-1   ; N, M, ,, ., /, RShift
20410 00005DBA 96FF20FF            <1>       db    150,-1,' ',-1       ; *, ALT, Spc, CL
20411                              <1>       ;                         ;----- FUNCTIONS ------
20412 00005DBE 5E5F60616263        <1>       db    94,95,96,97,98,99   ; F1 - F6
20413 00005DC4 64656667FFFF        <1>       db    100,101,102,103,-1,-1     ; F7 - F10, NL, SL
20414 00005DCA 778D848E738F        <1>       db    119,141,132,142,115,143   ; Home, Up, PgUp, -, Left, Pad5
20415 00005DD0 749075917692        <1>       db    116,144,117,145,118,146 ; Right, +, End, Down, PgDn, Ins
20416 00005DD6 93FFFFFF898A        <1>       db    147,-1,-1,-1,137,138      ; Del, SysReq, Undef, WT, F11, F12
20417                              <1>
20418                              <1> ;-----      TABLES FOR LOWER CASE ----------
20419 00005DDC 1B3132333435363738- <1> K10:  db    27,'1234567890-=',8,9
20420 00005DE5 39302D3D0809        <1>
20421 00005DEB 7177657274975696F- <1>       db    'qwertyuiop[]',13,-1,'asdfghjkl;',39
20422 00005DF4 705B5D0DFF61736466- <1>
20423 00005DFD 67686A6B6C3B27      <1>
20424 00005E04 60FF5C7A786376626E- <1>       db    96,-1,92,'zxcvbnm,./',-1,'*',-1,' ',-1
20425 00005E0D 6D2C2E2FFF2AFF20FF  <1>
20426                              <1> ;-----      LC TABLE SCAN
20427 00005E16 3B3C3D3E3F          <1>       db    59,60,61,62,63            ; BASE STATE OF F1 - F10
20428 00005E1B 4041424344          <1>       db    64,65,66,67,68
20429 00005E20 FFFF                <1>       db    -1,-1               ; NL, SL
20430                              <1>
20431                              <1> ;-----      KEYPAD TABLE
20432 00005E22 474849FF4BFF        <1> K15:  db    71,72,73,-1,75,-1   ; BASE STATE OF KEYPAD KEYS
20433 00005E28 4DFF4F50515253      <1>       db    77,-1,79,80,81,82,83
20434 00005E2F FFFF5C8586          <1>       db    -1,-1,92,133,134    ; SysRq, Undef, WT, F11, F12
20435                              <1>
20436                              <1> ;-----      TABLES FOR UPPER CASE ----------
20437 00005E34 1B21402324255E262A- <1> K11:  db    27,'!@#$%',94,'&*()_+',8,0
20438 00005E3D 28295F2B0800        <1>
20439 00005E43 51574552545955494F- <1>       db    'QWERTYUIOP{}',13,-1,'ASDFGHJKL:"'
20440 00005E4C 507B7D0DFF41534446- <1>
20441 00005E55 47484A4B4C3A22      <1>
20442 00005E5C 7EFF7C5A584356424E- <1>       db    126,-1,'|ZXCVBNM<>?',-1,'*',-1,' ',-1
20443 00005E65 4D3C3E3FFF2AFF20FF  <1>
20444                              <1> ;-----      UC TABLE SCAN
20445 00005E6E 5455565758          <1> K12:  db    84,85,86,87,88            ; SHIFTED STATE OF F1 - F10
20446 00005E73 595A5B5C5D          <1>       db    89,90,91,92,93
20447 00005E78 FFFF                <1>       db    -1,-1               ; NL, SL
20448                              <1>
20449                              <1> ;-----      NUM STATE TABLE
20450 00005E7A 3738392D3435362B31- <1> K14:  db    '789-456+1230.'           ; NUMLOCK STATE OF KEYPAD KEYS
20451 00005E83 3233302E            <1>
20452                              <1>       ;
20453 00005E87 FFFF7C8788          <1>       db    -1,-1,124,135,136   ; SysRq, Undef, WT, F11, F12
20454                              <1>
20455                              <1> ; 26/08/2014
20456                              <1> ; Retro UNIX 8086 v1 - UNIX.ASM (03/03/2014)
20457                              <1> ; Derived from IBM "pc-at"
20458                              <1> ; rombios source code (06/10/1985)
20459                              <1> ; 'dseg.inc'
20460                              <1>
20461                              <1> ;---------------------------------------;
20462                              <1> ;     SYSTEM DATA AREA          ;
20463                              <1> ;---------------------------------------
20464 00005E8C 00                  <1> BIOS_BREAK   db     0           ; BIT 7=1 IF BREAK KEY HAS BEEN PRESSED
20465                              <1>
20466                              <1> ;---------------------------------------
20467                              <1> ;     KEYBOARD DATA AREAS       ;
20468                              <1> ;---------------------------------------
20469                              <1>
20470 00005E8D 00                  <1> KB_FLAG           db    0           ; KEYBOARD SHIFT STATE AND STATUS FLAGS
20471 00005E8E 00                  <1> KB_FLAG_1    db     0           ; SECOND BYTE OF KEYBOARD STATUS
20472 00005E8F 00                  <1> KB_FLAG_2    db     0           ; KEYBOARD LED FLAGS
20473 00005E90 00                  <1> KB_FLAG_3    db     0           ; KEYBOARD MODE STATE AND TYPE FLAGS
20474 00005E91 00                  <1> ALT_INPUT    db     0           ; STORAGE FOR ALTERNATE KEY PAD ENTRY
20475 00005E92 [A25E0000]          <1> BUFFER_START dd     KB_BUFFER    ; OFFSET OF KEYBOARD BUFFER START
20476 00005E96 [C25E0000]          <1> BUFFER_END   dd     KB_BUFFER + 32      ; OFFSET OF END OF BUFFER
20477 00005E9A [A25E0000]          <1> BUFFER_HEAD  dd     KB_BUFFER    ; POINTER TO HEAD OF KEYBOARD BUFFER
20478 00005E9E [A25E0000]          <1> BUFFER_TAIL  dd     KB_BUFFER    ; POINTER TO TAIL OF KEYBOARD BUFFER
20479                              <1> ; ------    HEAD = TAIL  INDICATES THAT THE BUFFER IS EMPTY
20480 00005EA2 0000<rept>          <1> KB_BUFFER    times  16 dw 0            ; ROOM FOR 16 SCAN CODE ENTRIES
20481                              <1>
20482                              <1> ; /// End Of KEYBOARD DATA ///
20483                                  %include 'vidata.s'       ; VIDEO (BIOS) DATA
20484                              <1> ; ********************************************************************************
20485                              <1> ; TRDOS386.ASM (TRDOS 386 Kernel) - v2.0.0 - vidata.s
20486                              <1> ; --------------------------------------------------------------------------------
20487                              <1> ; Last Update: 31/07/2016
20488                              <1> ; --------------------------------------------------------------------------------
20489                              <1> ; Beginning: 16/01/2016
20490                              <1> ; --------------------------------------------------------------------------------
20491                              <1> ; Assembler: NASM version 2.11 (trdos386.s)
20492                              <1> ; --------------------------------------------------------------------------------
20493                              <1> ; Turkish Rational DOS
20494                              <1> ; Operating System Project v2.0 by ERDOGAN TAN (Beginning: 04/01/2016)
20495                              <1> ;
20496                              <1> ; Derived from 'Retro UNIX 386 Kernel - v0.2.1.0' source code by Erdogan Tan
20497                              <1> ; vidata.inc (11/03/2015)
20498                              <1> ;
```

```
20499                                  <1> ; Derived from 'IBM PC-AT' BIOS source code (1985)
20500                                  <1> ; ********************************************************************************
20501                                  <1>
20502                                  <1> ; Retro UNIX 386 v1 Kernel - VIDATA.S
20503                                  <1> ; Last Modification: 11/03/2015
20504                                  <1> ;                (Data section for 'VIDEO.INC')
20505                                  <1> ;
20506                                  <1> ; //////// VIDEO DATA ///////////////
20507                                  <1>
20508                                  <1> ;----------------------------------------
20509                                  <1> ;     VIDEO DISPLAY DATA AREA          ;
20510                                  <1> ;----------------------------------------
20511 00005EC2 03                      <1> CRT_MODE:    db     3      ; CURRENT DISPLAY MODE (TYPE)
20512 00005EC3 29                      <1> CRT_MODE_SET:      db     29h    ; CURRENT SETTING OF THE 3X8 REGISTER
20513                                  <1>                            ; (29h default setting for video mode 3)
20514                                  <1>                            ; Mode Select register Bits
20515                                  <1>                            ;   BIT 0 - 80x25 (1), 40x25 (0)
20516                                  <1>                            ;   BIT 1 - ALPHA (0), 320x200 GRAPHICS (1)
20517                                  <1>                            ;   BIT 2 - COLOR (0), BW (1)
20518                                  <1>                            ;   BIT 3 - Video Sig. ENABLE (1), DISABLE (0)
20519                                  <1>                            ;   BIT 4 - 640x200 B&W Graphics Mode (1)
20520                                  <1>                            ;   BIT 5 - ALPHA mode BLINKING (1)
20521                                  <1>                            ;   BIT 6, 7 - Not Used
20522                                  <1>
20523                                  <1> ; Mode 0 - 2Ch = 101100b  ; 40x25 text, 16 gray colors
20524                                  <1> ; Mode 1 - 28h = 101000b  ; 40x25 text, 16 fore colors, 8 back colors
20525                                  <1> ; Mode 2 - 2Dh = 101101b  ; 80x25 text, 16 gray colors
20526                                  <1> ; Mode 3 - 29h = 101001b  ; 80x25 text, 16 fore color, 8 back color
20527                                  <1> ; Mode 4 - 2Ah = 101010b  ; 320x200 graphics, 4 colors
20528                                  <1> ; Mode 5 - 2Eh = 101110b  ; 320x200 graphics, 4 gray colors
20529                                  <1> ; Mode 6 - 1Eh = 011110b  ; 640x200 graphics, 2 colors
20530                                  <1> ; Mode 7 - 29h = 101001b  ; 80x25 text, black & white colors
20531                                  <1> ; Mode & 37h = Video signal OFF
20532                                  <1>
20533                                  <1> ; 24/06/2016
20534 00005EC4 50                      <1> CRT_COLS:    db     80     ; Number of columns
20535                                  <1>
20536                                  <1> ; 01/07/2016
20537 00005EC5 00                      <1> CRT_PALETTE: db     0      ; Current palette setting
20538                                  <1>
20539                                  <1> ; 03/07/2016
20540 00005EC6 10                      <1> CHAR_HEIGHT: db     16     ; Default character height
20541 00005EC7 60                      <1> VGA_VIDEO_CTL:     db     60h    ; ROM BIOS DATA AREA Offset 87h
20542 00005EC8 F9                      <1> VGA_SWITCHES:      db     0F9h   ; Feature Bit Switches (the basic screen)
20543 00005EC9 51                      <1> VGA_MODESET_CTL: db       051h   ; Basic mode set options (VGA video flags)
20544                                  <1>                            ; ROM BIOS DATA AREA Offset 89h
20545                                  <1>                            ; Bit 7, 4 : Mode
20546                                  <1>                            ;       01 : 400-line mode
20547                                  <1>                            ; Bit 6      : Display switch enabled =  1
20548                                  <1>                            ; Bit 5      : Reserved  = 0
20549                                  <1>                            ; Bit 3      : Default palette loading
20550                                  <1>                            ;       disabled = 0
20551                                  <1>                            ; Bit 2 : Color monitor = 0
20552                                  <1>                            ; Bit 1 = Gray scale summing
20553                                  <1>                            ;       disabled = 0
20554                                  <1>                            ; Bit 0 = VGA active = 1
20555 00005ECA 19                      <1> VGA_ROWS:    db     25
20556                                  <1>
20557                                  <1> ; 16/01/2016
20558                                  <1> chr_attrib:  ; Character color/attributes for viode pages (0 to 7)
20559 00005ECB 0707070707070707        <1>       db     07h, 07h, 07h, 07h, 07h, 07h, 07h, 07h
20560                                  <1> ; 30/01/2016
20561                                  <1> vmode:
20562 00005ED3 0303030303030303        <1>       db     3,3,3,3,3,3,3,3 ; video modes for pseudo screens
20563                                  <1>
20564                                  <1> CURSOR_MODE: ; cursor start (ch) = 14, cursor end (cl) = 15
20565 00005EDB 0F0E                    <1>       db     15, 14 ; 07/07/2016 - TRDOS 386 (TRDOS v2.0)
20566                                  <1>
20567                                  <1> ;align 4
20568                                  <1> ;VGA_BASE: ; 26/07/2016
20569                                  <1> ;     dd      0B8000h  ; (Mode < 0Dh) or 0A0000h (mode >= 0Dh)
20570                                  <1>
20571 00005EDD 90                      <1> align 2
20572                                  <1>
20573                                  <1> vga_modes:
20574                                  <1>       ; 25/07/2016
20575                                  <1>       ; 09/07/2016
20576                                  <1>       ; 03/07/2016
20577                                  <1>       ; valid (implemented) video modes (>7, extension to IBM PC CGA modes)
20578 00005EDE 0302010007040506        <1>       db     03h, 02h, 01h, 00h, 07h, 04h, 05h, 06h
20579                                  <1> vga_g_modes: ; 31/07/2016
20580 00005EE6 13F0126A0D0E1011        <1>       db     13h, 0F0h, 12h, 6Ah, 0Dh, 0Eh, 10h, 11h
20581                                  <1> vga_mode_count equ $ - vga_modes
20582                                  <1> vga_g_mode_count equ $ - vga_g_modes
20583                                  <1>
20584                                  <1> vga_mode_tbl_ptr:
20585                                  <1>       ; 25/07/2016
20586 00005EEE [4E5F0000]             <1>       dd     vga_mode_03h
20587 00005EF2 [4E5F0000]             <1>       dd     vga_mode_03h ; mode 02h -> mode 03h
20588 00005EF6 [8E5F0000]             <1>       dd     vga_mode_01h
20589 00005EFA [8E5F0000]             <1>       dd     vga_mode_01h ; mode 00h -> mode 01h
20590                                  <1>       ;dd    vga_mode_07h
20591 00005EFE [4E5F0000]             <1>       dd     vga_mode_03h ; mode 07h -> mode 03h
20592 00005F02 [CE5F0000]             <1>       dd     vga_mode_04h
20593 00005F06 [CE5F0000]             <1>       dd     vga_mode_04h ; mode 05h -> mode 04h
20594 00005F0A [0E600000]             <1>       dd     vga_mode_06h
20595 00005F0E [4E600000]             <1>       dd     vga_mode_13h
20596 00005F12 [8E600000]             <1>       dd     vga_mode_F0h
20597 00005F16 [CE600000]             <1>       dd     vga_mode_12h
20598 00005F1A [0E610000]             <1>       dd     vga_mode_6Ah
20599 00005F1E [4E610000]             <1>       dd     vga_mode_0Dh
20600 00005F22 [8E610000]             <1>       dd     vga_mode_0Eh
```

```
20601 00005F26 [CE610000]          <1>       dd    vga_mode_10h
20602 00005F2A [0E620000]          <1>       dd    vga_mode_11h
20603                              <1>
20604                              <1> vga_memmodel:
20605                              <1>       ; 25/07/2016
20606                              <1>       ; 07/07/2016
20607                              <1>       CTEXT  equ 0
20608                              <1>       ;MTEXT equ 1
20609                              <1>       MTEXT  equ 0 ; mode 07h -> mode 03h
20610                              <1>       CGA    equ 2
20611                              <1>       LINEAR8 equ 5
20612                              <1>       PLANAR4      equ 4
20613                              <1>       PLANAR1      equ 3
20614 00005F2E 0000000000020202   <1>       db    CTEXT, CTEXT, CTEXT, CTEXT, MTEXT, CGA, CGA, CGA
20615                              <1> vga_g_memmodel: ; 31/07/2016
20616 00005F36 0504040404040403   <1>       db    LINEAR8, PLANAR4, PLANAR4, PLANAR4, PLANAR4, PLANAR4, PLANAR4, PLANAR1
20617                              <1> ;vga_pixbits:
20618                              <1> ;     ; 25/07/2016
20619                              <1> ;     ; 08/07/2016
20620                              <1> ;     db    4, 4, 4, 4, 4, 2, 2, 1, 8, 4, 4, 4, 4, 4, 4, 1
20621                              <1> vga_dac_s:
20622 00005F3E 020202020001010103- <1>       db    2, 2, 2, 2, 0, 1, 1, 1, 3, 3, 2, 2, 1, 1, 2, 2
20623 00005F47 03020201010202      <1>
20624                              <1>
20625                              <1> vga_params:
20626                              <1>       ; 25/07/2016
20627                              <1>       ; 19/07/2016
20628                              <1>       ; 03/07/2016
20629                              <1>       ; derived from 'Plex86/Bochs VGABios' source code
20630                              <1>       ; vgabios-0.7a (2011)
20631                              <1>       ; by the LGPL VGABios Developers Team (2001-2008)
20632                              <1>       ; 'vgatables.h'
20633                              <1>       ; Oracle VirtualBox 5.0.24 VGABios Source Code
20634                              <1>       ; ('vgabios.c', 'vgatables.h', 'vgafonts.h', 'vgarom.asm')
20635                              <1>       ;
20636                              <1> vga_mode_03h:  ; mode 03h, 80*25 text, CGA colors
20637 00005F4E 5018100010          <1>       db    80, 24, 16, 00h, 10h ; tw, th-1, ch, slength (5)
20638 00005F53 00030002            <1>       db    00h, 03h, 00h, 02h ; sequ regs (4)
20639 00005F57 67                  <1>       db    67h   ; misc reg (1)
20640 00005F58 5F4F50825581BF1F    <1>       db    5Fh, 4Fh, 50h, 82h, 55h, 81h, 0BFh, 1Fh
20641 00005F60 004F                <1>       db    00h, 4Fh
20642                              <1> vga_p_cm_pos equ $ - vga_mode_03h
20643 00005F62 0D0E00000000        <1>       db    0Dh, 0Eh, 00h, 00h, 00h, 00h
20644 00005F68 9C8E8F281F96B9A3    <1>       db    9Ch, 8Eh, 8Fh, 28h, 1Fh, 96h, 0B9h, 0A3h
20645 00005F70 FF                  <1>       db    0FFh  ; crtc_regs (25)
20646 00005F71 0001020304051407    <1>       db    00h, 01h, 02h, 03h, 04h, 05h, 14h, 07h
20647 00005F79 38393A3B3C3D3E3F    <1>       db    38h, 39h, 3Ah, 3Bh, 3Ch, 3Dh, 3Eh, 3Fh
20648 00005F81 0C000F08            <1>       db    0Ch, 00h, 0Fh, 08h  ; actl regs (20)
20649 00005F85 0000000000100E0FFF  <1>       db    00h, 00h, 00h, 00h, 00h, 10h, 0Eh, 0Fh, 0FFh ; grdc regs (9)
20650                              <1> vga_mode_01h:      ; mode 01h, 40*25 text, CGA colors
20651 00005F8E 2818100008          <1>       db    40, 24, 16, 00h, 08h ; tw, th-1, ch, slength
20652 00005F93 08030002            <1>       db    08h, 03h, 00h, 02h  ; sequ regs
20653 00005F97 67                  <1>       db    67h   ; misc reg
20654 00005F98 2D2728902BA0BF1F    <1>       db    2Dh, 27h, 28h, 90h, 2Bh, 0A0h, 0BFh, 1Fh
20655 00005FA0 004F0D0E00000000    <1>       db    00h, 4Fh, 0Dh, 0Eh, 00h, 00h, 00h, 00h
20656 00005FA8 9C8E8F141F96B9A3    <1>       db    9Ch, 8Eh, 8Fh, 14h, 1Fh, 96h, 0B9h, 0A3h
20657 00005FB0 FF                  <1>       db    0FFh  ; crtc_regs
20658 00005FB1 0001020304051407    <1>       db    00h, 01h, 02h, 03h, 04h, 05h, 14h, 07h
20659 00005FB9 38393A3B3C3D3E3F    <1>       db    38h, 39h, 3Ah, 3Bh, 3Ch, 3Dh, 3Eh, 3Fh
20660 00005FC1 0C000F08            <1>       db    0Ch, 00h, 0Fh, 08h  ; actl regs
20661 00005FC5 0000000000100E0FFF  <1>       db    00h, 00h, 00h, 00h, 00h, 10h, 0Eh, 0Fh, 0FFh ; grdc regs
20662                              <1> ;vga_mode_07h:     ; mode 07h, 80*25 text, mono color
20663                              <1> ;     db    80, 24, 16, 00h, 10h  ; tw, th-1, ch, slength
20664                              <1> ;     db    00h, 03h, 00h, 02h ; sequ regs
20665                              <1> ;     db    66h   ; misc reg
20666                              <1> ;     db    5Fh, 4Fh, 50h, 82h, 55h, 81h, 0BFh, 1Fh
20667                              <1> ;     db    00h, 4Fh, 0Dh, 0Eh, 00h, 00h, 00h, 00h
20668                              <1> ;     db    9Ch, 8Eh, 8Fh, 28h, 0Fh, 96h, 0B9h, 0A3h
20669                              <1> ;     db    0FFh  ; crtc regs
20670                              <1> ;     db    00h, 08h, 08h, 08h, 08h, 08h, 08h, 08h
20671                              <1> ;     db    10h, 18h, 18h, 18h, 18h, 18h, 18h, 18h
20672                              <1> ;     db    0Eh, 00h, 0Fh, 08h  ; actl regs
20673                              <1> ;     db    00h, 00h, 00h, 00h, 00h, 10h, 0Ah, 0Fh, 0FFh ; grdc regs
20674                              <1> vga_mode_04h:      ; 320*200 graphics, 4 colors, CGA
20675 00005FCE 2818080008          <1>       db    40, 24, 8, 00h, 08h   ; tw, th-1, ch, slength
20676 00005FD3 09030002            <1>       db    09h, 03h, 00h, 02h ; sequ regs
20677 00005FD7 63                  <1>       db    63h   ; misc reg
20678 00005FD8 2D2728902B80BF1F    <1>       db    2Dh, 27h, 28h, 90h, 2Bh, 80h, 0BFh, 1Fh
20679 00005FE0 00C1000000000000    <1>       db    00h, 0C1h, 00h, 00h, 00h, 00h, 00h, 00h
20680 00005FE8 9C8E8F140096B9A2    <1>       db    9Ch, 8Eh, 8Fh, 14h, 00h, 96h, 0B9h, 0A2h
20681 00005FF0 FF                  <1>       db    0FFh  ; crtc_regs
20682 00005FF1 0013151702040607    <1>       db    00h, 13h, 15h, 17h, 02h, 04h, 06h, 07h
20683 00005FF9 1011121314151617    <1>       db    10h, 11h, 12h, 13h, 14h, 15h, 16h, 17h
20684 00006001 01000300            <1>       db    01h, 00h, 03h, 00h ; actl regs
20685 00006005 0000000000300F0FFF  <1>       db    00h, 00h, 00h, 00h, 00h, 30h, 0Fh, 0Fh, 0FFh ; grdc regs
20686                              <1> vga_mode_06h:      ; 640*200 graphics, 2 colors, CGA
20687 0000600E 5018080010          <1>       db    80, 24, 8, 00h, 10h   ;   tw, th-1, ch, slength
20688 00006013 01010006            <1>       db    01h, 01h, 00h, 06h ; sequ regs
20689 00006017 63                  <1>       db    63h   ; misc reg
20690 00006018 5F4F50825480BF1F    <1>       db    5Fh, 4Fh, 50h, 82h, 54h, 80h, 0BFh, 1Fh
20691 00006020 00C1000000000000    <1>       db    00h, 0C1h, 00h, 00h, 00h, 00h, 00h, 00h
20692 00006028 9C8E8F280096B9C2    <1>       db    9Ch, 8Eh, 8Fh, 28h, 00h, 96h, 0B9h, 0C2h
20693 00006030 FF                  <1>       db    0FFh  ; crtc regs
20694 00006031 0017171717171717    <1>       db    00h, 17h, 17h, 17h, 17h, 17h, 17h, 17h
20695 00006039 1717171717171717    <1>       db    17h, 17h, 17h, 17h, 17h, 17h, 17h, 17h
20696 00006041 01000100            <1>       db    01h, 00h, 01, 00h  ; actl regs
20697 00006045 0000000000000D0FFF  <1>       db    00h, 00h, 00h, 00h, 00h, 00h, 0Dh, 0Fh, 0FFh ; grdc regs
20698                              <1> vga_mode_13h:  ; mode 13h, 300*200, 256 colors, linear
20699 0000604E 2818080000          <1>       db    40, 24, 8, 0, 0     ; tw, th-1, ch, slength (5)
20700 00006053 010F000E            <1>       db    01h, 0Fh, 00h, 0Eh ; sequ regs (4)
20701 00006057 63                  <1>       db    63h   ; misc reg (1)
20702 00006058 5F4F50825480BF1F    <1>       db    5Fh, 4Fh, 50h, 82h, 54h, 80h, 0BFh, 1Fh
```

```
20703 00006060 0041000000000000    <1>       db    00h, 41h, 00h, 00h, 00h, 00h, 00h, 00h
20704 00006068 9C8E8F284096B9A3    <1>       db    9Ch, 8Eh, 8Fh, 28h, 40h, 96h, 0B9h, 0A3h
20705 00006070 FF                  <1>       db    0FFh  ; crtc regs (25)
20706 00006071 0001020304050607    <1>       db    00h, 01h, 02h, 03h, 04h, 05h, 06h, 07h
20707 00006079 08090A0B0C0D0E0F    <1>       db    08h, 09h, 0Ah, 0Bh, 0Ch, 0Dh, 0Eh, 0Fh
20708 00006081 41000F00            <1>       db    41h, 00h, 0Fh, 00h  ; actl regs (20)
20709 00006085 000000000040050FFF  <1>       db    00h, 00h, 00h, 00h, 00h, 40h, 05h, 0Fh, 0FFh ; grdc regs (9)
20710                              <1> vga_mode_setl equ $ - vga_mode_13h  ; = 64
20711                              <1> vga_mode_F0h:  ; mode X ; 320*240, 256 colors, planar
20712 0000608E 2818080000          <1>       db    40, 24, 8, 0, 0     ; tw, th-1, ch, slength
20713 00006093 010F0006            <1>       db    01h, 0Fh, 00h, 06h ; sequ regs
20714 00006097 E3                  <1>       db    0E3h  ; misc reg
20715 00006098 5F4F508254800D3E    <1>       db    5Fh, 4Fh, 50h, 82h, 54h, 80h, 0Dh, 3Eh
20716 000060A0 0041000000000000    <1>       db    00h, 41h, 00h, 00h, 00h, 00h, 00h, 00h
20717 000060A8 EAACDF2800E706E3    <1>       db    0EAh, 0ACh, 0DFh, 28h, 00h, 0E7h, 06h, 0E3h
20718 000060B0 FF                  <1>       db    0FFh  ; crtc regs (25)
20719 000060B1 0001020304050607    <1>       db    00h, 01h, 02h, 03h, 04h, 05h, 06h, 07h
20720 000060B9 08090A0B0C0D0E0F    <1>       db    08h, 09h, 0Ah, 0Bh, 0Ch, 0Dh, 0Eh, 0Fh
20721 000060C1 41000F00            <1>       db    41h, 00h, 0Fh, 00h  ; actl regs
20722 000060C5 000000000040050FFF  <1>       db    00h, 00h, 00h, 00h, 00h, 40h, 05h, 0Fh, 0FFh ; grdc regs
20723                              <1> vga_mode_12h:  ; mode 12h, 640*480, 16 colors, planar
20724 000060CE 501D100000          <1>       db    80, 29, 16, 0, 0 ; tw, th-1, ch, slength
20725 000060D3 010F0006            <1>       db    01h, 0Fh, 00h, 06h ; sequ regs
20726 000060D7 E3                  <1>       db    0E3h  ; misc reg
20727 000060D8 5F4F508254800B3E    <1>       db    5Fh, 4Fh, 50h, 82h, 54h, 80h, 0Bh, 3Eh
20728 000060E0 0040000000000000    <1>       db    00h, 40h, 00h, 00h, 00h, 00h, 00h, 00h
20729 000060E8 EA8CDF2800E704E3    <1>       db    0EAh, 8Ch, 0DFh, 28h, 00h, 0E7h, 04h, 0E3h
20730 000060F0 FF                  <1>       db    0FFh  ; crtc regs
20731 000060F1 0001020304051407    <1>       db    00h, 01h, 02h, 03h, 04h, 05h, 14h, 07h
20732 000060F9 38393A3B3C3D3E3F    <1>       db    38h, 39h, 3Ah, 3Bh, 3Ch, 3Dh, 3Eh, 3Fh
20733 00006101 01000F00            <1>       db    01h, 00h, 0Fh, 00h  ; actl regs
20734 00006105 000000000000050FFF  <1>       db    00h, 00h, 00h, 00h, 00h, 00h, 05h, 0Fh, 0FFh ; grdc regs
20735                              <1> vga_mode_6Ah:  ; mode 6Ah, 800*600, 16 colors, planar
20736 0000610E 6424100000          <1>       db    100, 36, 16, 0, 0 ; tw, th-1, ch, slength
20737 00006113 010F0006            <1>       db    01h, 0Fh, 00h, 06h ; sequ regs
20738 00006117 E3                  <1>       db    0E3h  ; misc reg
20739 00006118 7F6363836B1B72F0    <1>       db    7Fh, 63h, 63h, 83h, 6Bh, 1Bh, 72h, 0F0h
20740 00006120 0060000000000000    <1>       db    00h, 60h, 00h, 00h, 00h, 00h, 00h, 00h
20741 00006128 598D5732005773E3    <1>       db    59h, 8Dh, 57h, 32h, 00h, 57h, 73h, 0E3h
20742 00006130 FF                  <1>       db    0FFh  ; crtc regs
20743 00006131 0001020304051407    <1>       db    00h, 01h, 02h, 03h, 04h, 05h, 14h, 07h
20744 00006139 38393A3B3C3D3E3F    <1>       db    38h, 39h, 3Ah, 3Bh, 3Ch, 3Dh, 3Eh, 3Fh
20745 00006141 01000F00            <1>       db    01h, 00h, 0Fh, 00h  ; actl regs
20746 00006145 000000000000050FFF  <1>       db    00h, 00h, 00h, 00h, 00h, 00h, 05h, 0Fh, 0FFh ; grdc regs
20747                              <1> vga_mode_0Dh:  ; mode 0Dh, 320*200, 16 colors, planar
20748 0000614E 2818080020          <1>       db    40, 24, 8, 0, 20h ; tw, th-1, ch, slength
20749 00006153 090F0006            <1>       db    09h, 0Fh, 00h, 06h ; sequ regs
20750 00006157 63                  <1>       db    63h   ; misc reg
20751 00006158 2D2728902B80BF1F    <1>       db    2Dh, 27h, 28h, 90h, 2Bh, 80h, 0BFh, 1Fh
20752 00006160 00C0000000000000    <1>       db    00h, 0C0h, 00h, 00h, 00h, 00h, 00h, 00h
20753 00006168 9C8E8F140096B9E3    <1>       db    9Ch, 8Eh, 8Fh, 14h, 00h, 96h, 0B9h, 0E3h
20754 00006170 FF                  <1>       db    0FFh  ; crtc regs
20755 00006171 0001020304050607    <1>       db    00h, 01h, 02h, 03h, 04h, 05h, 06h, 07h
20756 00006179 1011121314151617    <1>       db    10h, 11h, 12h, 13h, 14h, 15h, 16h, 17h
20757 00006181 01000F00            <1>       db    01h, 00h, 0Fh, 00h  ; actl regs
20758 00006185 000000000000050FFF  <1>       db    00h, 00h, 00h, 00h, 00h, 00h, 05h, 0Fh, 0FFh ; grdc regs
20759                              <1> vga_mode_0Eh:  ; mode 0Eh, 640*200, 16 colors, planar
20760 0000618E 5018080040          <1>       db    80, 24, 8, 0, 40h ; tw, th-1, ch, slength
20761 00006193 010F0006            <1>       db    01h, 0Fh, 00h, 06h ; sequ regs
20762 00006197 63                  <1>       db    63h   ; misc reg
20763 00006198 5F4F50825480BF1F    <1>       db    5Fh, 4Fh, 50h, 82h, 54h, 80h, 0BFh, 1Fh
20764 000061A0 00C0000000000000    <1>       db    00h, 0C0h, 00h, 00h, 00h, 00h, 00h, 00h
20765 000061A8 9C8E8F280096B9E3    <1>       db    9Ch, 8Eh, 8Fh, 28h, 00h, 96h, 0B9h, 0E3h
20766 000061B0 FF                  <1>       db    0FFh  ; crtc regs
20767 000061B1 0001020304050607    <1>       db    00h, 01h, 02h, 03h, 04h, 05h, 06h, 07h
20768 000061B9 1011121314151617    <1>       db    10h, 11h, 12h, 13h, 14h, 15h, 16h, 17h
20769 000061C1 01000F00            <1>       db    01h, 00h, 0Fh, 00h  ; actl regs
20770 000061C5 000000000000050FFF  <1>       db    00h, 00h, 00h, 00h, 00h, 00h, 05h, 0Fh, 0FFh ; grdc regs
20771                              <1> vga_mode_10h: ; mode 10h, 640*350, 16 colors, planar
20772 000061CE 50180E0080          <1>       db    80, 24, 14, 0, 80h ; tw, th-1, ch, slength
20773 000061D3 010F0006            <1>       db    01h, 0Fh, 00h, 06h ; sequ regs
20774 000061D7 A3                  <1>       db    0A3h  ; misc reg
20775 000061D8 5F4F50825480BF1F    <1>       db    5Fh, 4Fh, 50h, 82h, 54h, 80h, 0BFh, 1Fh
20776 000061E0 0040000000000000    <1>       db    00h, 40h, 00h, 00h, 00h, 00h, 00h, 00h
20777 000061E8 83855D280F63BAE3    <1>       db    83h, 85h, 5Dh, 28h, 0Fh, 63h, 0BAh, 0E3h
20778 000061F0 FF                  <1>       db    0FFh  ; crtc regs
20779 000061F1 0001020304051407    <1>       db    00h, 01h, 02h, 03h, 04h, 05h, 14h, 07h
20780 000061F9 38393A3B3C3D3E3F    <1>       db    38h, 39h, 3Ah, 3Bh, 3Ch, 3Dh, 3Eh, 3Fh
20781 00006201 01000F00            <1>       db    01h, 00h, 0Fh, 00h  ; actl regs
20782 00006205 000000000000050FFF  <1>       db    00h, 00h, 00h, 00h, 00h, 00h, 05h, 0Fh, 0FFh ; grdc regs
20783                              <1> vga_mode_11h: ; mode 11h, 640*480, mono color, planar
20784 0000620E 501D100000          <1>       db    80, 29, 16, 0, 0   ; tw, th-1, ch, slength
20785 00006213 010F0006            <1>       db    01h, 0Fh, 00h, 06h ; sequ regs
20786 00006217 E3                  <1>       db    0E3h  ; misc reg
20787 00006218 5F4F508254800B3E    <1>       db    5Fh, 4Fh, 50h, 82h, 54h, 80h, 0Bh, 3Eh
20788 00006220 0040000000000000    <1>       db    00h, 40h, 00h, 00h, 00h, 00h, 00h, 00h
20789 00006228 EA8CDF2800E704E3    <1>       db    0EAh, 8Ch, 0DFh, 28h, 00h, 0E7h, 04h, 0E3h
20790 00006230 FF                  <1>       db    0FFh  ; crtc regs
20791 00006231 003F003F003F003F    <1>       db    00h, 3Fh, 00h, 3Fh, 00h, 3Fh, 00h, 3Fh
20792 00006239 003F003F003F003F    <1>       db    00h, 3Fh, 00h, 3Fh, 00h, 3Fh, 00h, 3Fh
20793 00006241 01000F00            <1>       db    01h, 00h, 0Fh, 00h  ; actl regs
20794 00006245 000000000000050FFF  <1>       db    00h, 00h, 00h, 00h, 00h, 00h, 05h, 0Fh, 0FFh ; grdc regs
20795                              <1> end_of_vga_params:
20796                              <1>
20797                              <1> ; /// End Of VIDEO DATA ///
20798                                  ;%include 'diskdata.s'    ; DISK (BIOS) DATA (initialized)
20799                                  ;;;
20800
20801                                  Align 2
20802
20803                                  %include 'sysdefs.s' ; 24/01/2015
20804                              <1> ; ****************************************************************************
```

```
20805                                  <1> ; TRDOS386.ASM (TRDOS 386 Kernel - v2.0.0) - SYSTEM DEFINITIONS : sysdefs.s
20806                                  <1> ; ----------------------------------------------------------------------------
20807                                  <1> ; Last Update: 10/04/2017
20808                                  <1> ; ----------------------------------------------------------------------------
20809                                  <1> ; Beginning: 24/01/2016
20810                                  <1> ; ----------------------------------------------------------------------------
20811                                  <1> ; Assembler: NASM version 2.11 (trdos386.s)
20812                                  <1> ; ----------------------------------------------------------------------------
20813                                  <1> ; Derived from 'Retro UNIX 386 Kernel - v0.2.1.0' source code by Erdogan Tan
20814                                  <1> ; sysdefs.inc (14/11/2015)
20815                                  <1> ; ****************************************************************************
20816                                  <1>
20817                                  <1> ; Retro UNIX 386 v1 Kernel - SYSDEFS.INC
20818                                  <1> ; Last Modification: 14/11/2015
20819                                  <1> ;
20820                                  <1> ; ///////// RETRO UNIX 386 V1 SYSTEM DEFINITIONS ///////////////
20821                                  <1> ; (Modified from
20822                                  <1> ;     Retro UNIX 8086 v1 system definitions in 'UNIX.ASM', 01/09/2014)
20823                                  <1> ; ((UNIX.ASM (RETRO UNIX 8086 V1 Kernel), 11/03/2013 - 01/09/2014))
20824                                  <1> ;     UNIX.ASM (MASM 6.11) --> SYSDEFS.INC (NASM 2.11)
20825                                  <1> ; ----------------------------------------------------------------------------
20826                                  <1> ;
20827                                  <1> ; Derived from UNIX Operating System (v1.0 for PDP-11)
20828                                  <1> ; (Original) Source Code by Ken Thompson (1971-1972)
20829                                  <1> ; <Bell Laboratories (17/3/1972)>
20830                                  <1> ; <Preliminary Release of UNIX Implementation Document>
20831                                  <1> ;
20832                                  <1> ; ****************************************************************************
20833                                  <1>
20834                                  <1> nproc       equ    16  ; number of processes
20835                                  <1> nfiles      equ    50
20836                                  <1> ntty  equ     8   ; 8+1 -> 8 (10/05/2013)
20837                                  <1> nbuf  equ    4   ; 6 ;; 21/08/2015 - 'namei' buffer problem when nbuf > 4
20838                                  <1>              ; NOTE: If fd0 super block buffer addres is beyond of the 1st
20839                                  <1>              ; 32K, DMA r/w routine or someting else causes a jump to
20840                                  <1>              ; kernel panic routine (in 'alloc' routine, in u5.s)
20841                                  <1>              ; because of invalid buffer content (r/w error).
20842                                  <1>              ; When all buffers are set before the end of the 1st 32k,
20843                                  <1>              ; there is no problem!? (14/11/2015)
20844                                  <1>
20845                                  <1> ;csgmnt     equ    2000h  ; 26/05/2013 (segment of process 1)
20846                                  <1> ;core equ    0          ; 19/04/2013
20847                                  <1> ;ecore      equ    32768 - 64  ; 04/06/2013 (24/05/2013)
20848                                  <1>       ; (if total size of argument list and arguments is 128 bytes)
20849                                  <1>       ; maximum executable file size = 32768-(64+40+128-6) = 32530 bytes
20850                                  <1>       ; maximum stack size = 40 bytes (+6 bytes for 'IRET' at 32570)
20851                                  <1>       ; initial value of user's stack pointer = 32768-64-128-2 = 32574
20852                                  <1>       ;     (sp=32768-args_space-2 at the beginning of execution)
20853                                  <1>       ; argument list offset = 32768-64-128 = 32576 (if it is 128 bytes)
20854                                  <1>       ; 'u' structure offset (for the '/core' dump file) = 32704
20855                                  <1>       ; '/core' dump file size = 32768 bytes
20856                                  <1>
20857                                  <1> ; 08/03/2014
20858                                  <1> ;sdsegmnt equ      6C0h  ; 256*16 bytes (swap data segment size for 16 processes)

20859                                  <1> ; 19/04/2013 Retro UNIX 8086 v1 feaure only !
20860                                  <1> ;;sdsegmnt equ     740h  ; swap data segment (for user structures and registers)
20861                                  <1>
20862                                  <1> ; 30/08/2013
20863                                  <1> time_count equ 4 ; 10 --> 4 01/02/2014
20864                                  <1>
20865                                  <1> ; 05/02/2014
20866                                  <1> ; process status
20867                                  <1> ;SFREE      equ 0
20868                                  <1> ;SRUN equ 1
20869                                  <1> ;SWAIT      equ 2
20870                                  <1> ;SZOMB      equ 3
20871                                  <1> ;SSLEEP     equ 4 ; Retro UNIX 8086 V1 extension (for sleep and wakeup)
20872                                  <1>
20873                                  <1> ; 09/03/2015
20874                                  <1> userdata equ 80000h ; user structure data address for current user ; temporary
20875                                  <1> swap_queue equ 90000h - 2000h ; swap queue address ; temporary
20876                                  <1> swap_alloc_table equ 0D0000h  ;  swap allocation table address ; temporary
20877                                  <1>
20878                                  <1> ; 17/09/2015
20879                                  <1> ESPACE equ 48 ; [u.usp] (at 'sysent') - [u.sp] value for error return
20880                                  <1>
20881                                  <1> ; 19/02/2017
20882                                  <1> ; 15/10/2016
20883                                  <1> ; 20/05/2016
20884                                  <1> ; 19/05/2016
20885                                  <1> ; 18/05/2016
20886                                  <1> ; 29/04/2016
20887                                  <1> ; TRDOS 386 (TRDOS v2.0) system calls - temporary List
20888                                  <1> ; 14/07/2013 - 21/09/2015 (Retro UNIX 8086 & 386 system calls)
20889                                  <1> ; UNIX v1 system calls
20890                                  <1> ;_rele      equ 0
20891                                  <1> _ver  equ 0 ; Get TRDOS version (v2.0)
20892                                  <1> _exit       equ 1
20893                                  <1> _fork       equ 2
20894                                  <1> _read       equ 3
20895                                  <1> _write      equ 4
20896                                  <1> _open equ 5
20897                                  <1> _close      equ 6
20898                                  <1> _wait       equ 7
20899                                  <1> _creat      equ 8
20900                                  <1> _link       equ 9
20901                                  <1> _unlink     equ 10
20902                                  <1> _exec equ 11
20903                                  <1> _chdir      equ 12
20904                                  <1> _time       equ 13
20905                                  <1> _mkdir      equ 14
```

```
20906                                   <1> _chmod      equ 15
20907                                   <1> _chown      equ 16
20908                                   <1> _break      equ 17
20909                                   <1> _stat equ 18
20910                                   <1> _seek equ 19
20911                                   <1> _tell       equ 20
20912                                   <1> _mount      equ 21
20913                                   <1> _umount     equ 22
20914                                   <1> _setuid     equ 23
20915                                   <1> _getuid     equ 24
20916                                   <1> _stime      equ 25
20917                                   <1> _quit equ 26
20918                                   <1> _intr equ 27
20919                                   <1> _fstat      equ 28
20920                                   <1> _emt  equ 29
20921                                   <1> _mdate      equ 30
20922                                   <1> ;_stty      equ 31
20923                                   <1> _video  equ 31 ; TRDOS 386 Video Functions (16/05/2016)
20924                                   <1> ;_gtty      equ 32
20925                                   <1> _audio      equ 32 ; TRDOS 386 Video Functions (16/05/2016)
20926                                   <1> ;_ilgins equ 33
20927                                   <1> _timer      equ 33 ; TRDOS 386 Timer Functions (18/05/2016)
20928                                   <1> _sleep      equ 34 ; Retro UNIX 8086 v1 feature only !
20929                                   <1> _msg  equ 35 ; Retro UNIX 386 v1 feature only !
20930                                   <1> _geterr     equ 36 ; Retro UNIX 386 v1 feature only !
20931                                   <1> _fpsave equ 37 ; TRDOS 386 FPU state option (28/02/2017)
20932                                   <1> _pri  equ 38 ; change priority - TRDOS 386 (20/05/2016)
20933                                   <1> _rele equ 39 ; TRDOS 386 (19/05/2016)
20934                                   <1> _fff  equ 40 ; Find First File - TRDOS 386 (15/10/2016)
20935                                   <1> _fnf  equ 41 ; Find Next File - TRDOS 386 (15/10/2016)
20936                                   <1> _alloc      equ 42 ; Allocate memory - TRDOS 386 (19/02/2017)
20937                                   <1>             ; TRDOS 386 (19/02/2017) DMA buff fuctions
20938                                   <1> _dalloc equ 43 ; Deallocate mem - TRDOS 386 (19/02/2017)
20939                                   <1> _calbac equ 44 ; Set IRQ callback - TRDOS 386 (20/02/2017)
20940                                   <1>
20941                                   <1> %macro sys 1-4
20942                                   <1>     ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
20943                                   <1>     ; 03/09/2015
20944                                   <1>     ; 13/04/2015
20945                                   <1>     ; Retro UNIX 386 v1 system call.
20946                                   <1>     %if %0 >= 2
20947                                   <1>         mov ebx, %2
20948                                   <1>         %if %0 >= 3
20949                                   <1>             mov ecx, %3
20950                                   <1>             %if %0 = 4
20951                                   <1>                mov edx, %4
20952                                   <1>             %endif
20953                                   <1>         %endif
20954                                   <1>     %endif
20955                                   <1>     mov eax, %1
20956                                   <1>     ;int 30h
20957                                   <1>     int 40h ; TRDOS 386 (TRDOS v2.0)
20958                                   <1> %endmacro
20959                                   <1>
20960                                   <1> ; TRDOS 386 system calls, interrupt number
20961                                   <1> ; 25/12/2016
20962                                   <1> SYSCALL_INT_NUM   equ '40' ; '40h'
20963                                   <1>
20964                                   <1> ; 13/05/2015 - ERROR CODES
20965                                   <1> ERR_FILE_NOT_OPEN  equ 10 ; 'file not open !' error
20966                                   <1> ERR_FILE_ACCESS    equ 11 ; 'permission denied !' error
20967                                   <1> ; 14/05/2015
20968                                   <1> ERR_DIR_ACCESS     equ 11 ; 'permission denied !' error
20969                                   <1> ERR_FILE_NOT_FOUND equ 12 ; 'file not found !' error
20970                                   <1> ERR_TOO_MANY_FILES equ 13 ; 'too many open files !' error
20971                                   <1> ERR_DIR_EXISTS     equ 14 ; 'directory already exists !' error
20972                                   <1> ; 16/05/2015
20973                                   <1> ERR_DRV_NOT_RDY    equ 15 ; 'drive not ready !' error
20974                                   <1> ; 18/05/2015
20975                                   <1> ERR_DEV_NOT_RDY    equ 15 ; 'device not ready !' error
20976                                   <1> ERR_DEV_ACCESS     equ 11 ; 'permission denied !' error
20977                                   <1> ERR_DEV_NOT_OPEN   equ 10 ; 'device not open !' error
20978                                   <1> ; 07/06/2015
20979                                   <1> ERR_FILE_EOF    equ 16 ; 'end of file !' error
20980                                   <1> ERR_DEV_VOL_SIZE   equ 16 ; 'out of volume !' error
20981                                   <1> ; 09/06/2015
20982                                   <1> ERR_DRV_READ    equ 17 ; 'disk read error !'
20983                                   <1> ERR_DRV_WRITE         equ 18 ; 'disk write error !'
20984                                   <1> ; 16/06/2015
20985                                   <1> ERR_NOT_DIR     equ 19 ; 'not a (valid) directory !' error
20986                                   <1> ERR_FILE_SIZE         equ 20 ; 'file size error !'
20987                                   <1> ; 22/06/2015
20988                                   <1> ERR_NOT_SUPERUSER  equ 11 ; 'permission denied !' error
20989                                   <1> ERR_NOT_OWNER      equ 11 ; 'permission denied !' error
20990                                   <1> ERR_NOT_FILE       equ 11 ; 'permission denied !' error
20991                                   <1> ; 23/06/2015
20992                                   <1> ERR_FILE_EXISTS    equ 14 ; 'file already exists !' error
20993                                   <1> ERR_DRV_NOT_SAME   equ 21 ; 'not same drive !' error
20994                                   <1> ERR_DIR_NOT_FOUND  equ 12 ; 'directory not found !' error
20995                                   <1> ERR_NOT_EXECUTABLE equ 22 ; 'not executable file !' error
20996                                   <1> ; 27/06/2015
20997                                   <1> ERR_INV_PARAMETER  equ 23 ; 'invalid parameter !' error
20998                                   <1> ERR_INV_DEV_NAME   equ 24 ; 'invalid device name !' error
20999                                   <1> ; 29/06/2015
21000                                   <1> ERR_TIME_OUT    equ 25 ; 'time out !' error
21001                                   <1> ERR_DEV_NOT_RESP   equ 25 ; 'device not responding !' error
21002                                   <1> ; 10/10/2016
21003                                   <1> ERR_INV_FILE_NAME  equ 26 ; 'invalid file name !' error
21004                                   <1> ERR_INV_FLAGS         equ 23 ; 'invalid flags !' error
21005                                   <1> ; For code compatibility with previous version of TRDOS (2011)
21006                                   <1> ; (Temporary error codes for current TRDOS 386 -2016- version)
21007                                   <1> ERR_NO_MORE_FILES  equ 12 ; 'no more files !' error
```

```
21008                                  <1> ERR_PATH_NOT_FOUND equ  3 ; 'path not found !' error
21009                                  <1>                       ; 'dir not found !' ; TRDOS 8086
21010                                  <1> ERR_NOT_FOUND:        equ  2 ; 'file not found !' ; TRDOS 8086
21011                                  <1> ERR_DISK_SPACE        equ 39 ; 'out of volume !' TRDOS 8086
21012                                  <1>                       ; 'insufficient disk space !' ; 27h
21013                                  <1> ERR_DISK_WRITE        equ 30 ; 'disk write protected !' ; 16/10/2016
21014                                  <1> ERR_ACCESS_DENIED  equ  5 ; 'access denied !' ; TRDOS 8086
21015                                  <1> ; 28/02/2017
21016                                  <1> ERR_PERM_DENIED       equ 11 ; 'permission denied !' error
21017                                  <1> ; 18/05/2016
21018                                  <1> ERR_MISC       equ 27 ; miscellaneous/other errors
21019                                  <1> ; 15/10/2016
21020                                  <1> ; TRDOS 8086 -> TRDOS 386 (0Bh -> 28)
21021                                  <1> ERR_INV_FORMAT        equ 28 ; 'invalid format !' error
21022                                  <1> ; TRDOS 8086 -> TRDOS 386 (0Dh -> 29)
21023                                  <1> ERR_INV_DATA    equ 29 ; 'invalid data !' error
21024                                  <1> ; TRDOS 8086 -> TRDOS 386 (0Eh -> 20)
21025                                  <1> ERR_ZERO_LENGTH       equ 20  ; 'zero length !' error
21026                                  <1> ; TRDOS 8086 -> TRDOS 386 (15h -> 17, 1Dh -> 18, 1Eh -> 17)
21027                                  <1> ERR_DRV_NR_READ       equ 17 ; 'drive not ready or read error !'
21028                                  <1> ERR_DRV_NR_WRITE   equ 18 ; 'drive not ready or write error !'
21029                                  <1> ; 15/10/2016
21030                                  <1> ERR_INV_PATH_NAME  equ 19 ; 'bad path name !' error
21031                                  <1> ERR_BAD_CMD_ARG       equ  1 ; 'bad command argument !' ; TRDOS 8086
21032                                  <1> ERR_INV_FNUMBER       equ  1 ; 'invalid function number !' ; TRDOS 8086
21033                                  <1> ERR_BIG_FILE    equ  8 ; 'big file & out of memory ! ; TRDOS 8086
21034                                  <1> ERR_BIG_DATA    equ  8 ; 'big data & out of memory ! ; TRDOS 8086
21035                                  <1> ERR_CLUSTER     equ 35 ; 'cluster not available !' ; TRDOS 8086
21036                                  <1> ERR_OUT_OF_MEMORY  equ  4 ; 'out of memory !'
21037                                  <1>                       ; 'insufficient memory !'
21038                                  <1> ERR_P_VIOLATION       equ 6 ; 'protection violation !'
21039                                  <1> ERR_PAGE_FAULT        equ 224 ;'page fault !' ;0E0h
21040                                  <1> ERR_SWP_DISK_READ           equ 40
21041                                  <1> ERR_SWP_DISK_NOT_PRESENT   equ 41
21042                                  <1> ERR_SWP_SECTOR_NOT_PRESENT equ 42
21043                                  <1> ERR_SWP_NO_FREE_SPACE      equ 43
21044                                  <1> ERR_SWP_DISK_WRITE         equ 44
21045                                  <1> ERR_SWP_NO_PAGE_TO_SWAP    equ 45
21046                                  <1> ; 10/04/2017
21047                                  <1> ERR_BUFFER     equ 46  ; 'buffer error ''
21048                                  <1>
21049                                  <1> ; 26/08/2015
21050                                  <1> ; 24/07/2015
21051                                  <1> ; 24/06/2015
21052                                  <1> MAX_ARG_LEN    equ 256 ; max. length of sys exec arguments
21053                                  <1> ; 01/07/2015
21054                                  <1> MAX_MSG_LEN    equ 255 ; max. msg length for 'sysmsg'
21055                                  <1> ;
21056                                  <1> ; 06/10/2016
21057                                  <1> OPENFILES      equ 10  ; max. number of open files (system)
21058                                  <1> ; 07/10/2016
21059                                  <1> ;NUMOFDEVICES         equ 20  ; max. num of available devices (sys)
21060                                  <1>
21061                                      %include 'trdosk0.s' ; 04/01/2016
21062                                  <1> ; ****************************************************************************
21063                                  <1> ; TRDOS386.ASM (TRDOS 386 Kernel - v2.0.0) - DEFINITIONS : trdosk0.s
21064                                  <1> ; ----------------------------------------------------------------------------
21065                                  <1> ; Last Update: 29/02/2016
21066                                  <1> ; ----------------------------------------------------------------------------
21067                                  <1> ; Beginning: 04/01/2016
21068                                  <1> ; ----------------------------------------------------------------------------
21069                                  <1> ; Assembler: NASM version 2.11 (trdos386.s)
21070                                  <1> ; ----------------------------------------------------------------------------
21071                                  <1> ; Derived from TRDOS Operating System v1.0 (8086) source code by Erdogan Tan
21072                                  <1> ; TRDOS2.ASM (09/11/2011)
21073                                  <1> ; ****************************************************************************
21074                                  <1> ; TRDOS2.ASM (c) 2004-2011 Erdogan TAN [ 17/01/2004 ] Last Update: 09/11/2011
21075                                  <1> ;
21076                                  <1> ; Masterboot / Partition Table at Beginning+1BEh
21077                                  <1> ptBootable       equ 0
21078                                  <1> ptBeginHead      equ 1
21079                                  <1> ptBeginSector    equ 2
21080                                  <1> ptBeginCylinder  equ 3
21081                                  <1> ptFileSystemID   equ 4
21082                                  <1> ptEndHead        equ 5
21083                                  <1> ptEndSector      equ 6
21084                                  <1> ptEndCylinder    equ 7
21085                                  <1> ptStartSector    equ 8
21086                                  <1> ptSectors        equ 12
21087                                  <1>
21088                                  <1> ; Boot Sector Parameters at 7C00h
21089                                  <1> DataArea1     equ -4
21090                                  <1> DataArea2     equ -2
21091                                  <1> BootStart     equ 0h
21092                                  <1> OemName       equ 03h
21093                                  <1> BytesPerSec   equ 0Bh
21094                                  <1> SecPerClust   equ 0Dh
21095                                  <1> ResSectors    equ 0Eh
21096                                  <1> FATs          equ 10h
21097                                  <1> RootDirEnts   equ 11h
21098                                  <1> Sectors       equ 13h
21099                                  <1> Media         equ 15h
21100                                  <1> FATSecs       equ 16h
21101                                  <1> SecPerTrack   equ 18h
21102                                  <1> Heads         equ 1Ah
21103                                  <1> Hidden1       equ 1Ch
21104                                  <1> Hidden2       equ 1Eh
21105                                  <1> HugeSec1      equ 20h
21106                                  <1> HugeSec2      equ 22h
21107                                  <1> DriveNumber   equ 24h
21108                                  <1> Reserved1     equ 25h
21109                                  <1> bootsignature equ 26h
```

```
21110                                   <1> VolumeID      equ 27h
21111                                   <1> VolumeLabel   equ 2Bh
21112                                   <1> FileSysType   equ 36h
21113                                   <1> Reserved2     equ 3Eh                         ; Starting cluster of P2000
21114                                   <1>
21115                                   <1> ; FAT32 BPB Structure
21116                                   <1> FAT32_FAT_Size equ 36
21117                                   <1> FAT32_RootFClust equ 44
21118                                   <1> FAT32_FSInfoSec equ 48
21119                                   <1> FAT32_DrvNum equ 64
21120                                   <1> FAT32_BootSig equ 66
21121                                   <1> FAT32_VolID equ 67
21122                                   <1> FAT32_VolLab equ 71
21123                                   <1> FAT32_FilSysType equ 82
21124                                   <1>
21125                                   <1> ; BIOS Disk Parameters
21126                                   <1> DPDiskNumber  equ 0h
21127                                   <1> DPDType       equ 1h
21128                                   <1> DPReturn      equ 2h
21129                                   <1> DPHeads       equ 3h
21130                                   <1> DPCylinders   equ 4h
21131                                   <1> DPSecPerTrack equ 6h
21132                                   <1> DPDisks       equ 7h
21133                                   <1> DPTableOff    equ 8h
21134                                   <1> DPTableSeg    equ 0Ah
21135                                   <1> DPNumOfSecs   equ 0Ch
21136                                   <1>
21137                                   <1> ; BIOS INT 13h Extensions (LBA extensions)
21138                                   <1> ; Just After DP Data (DPDiskNumber+)
21139                                   <1> DAP_PacketSize equ 10h  ; If extensions present, this byte will be >=10h
21140                                   <1> DAP_Reserved1 equ 11h   ; Reserved Byte
21141                                   <1> DAP_NumOfBlocks equ 12h ; Value of this byte must be 0 to 127
21142                                   <1> DAP_Reserved2 equ 13h   ; Reserved Byte
21143                                   <1> DAP_Destination equ 14h ; Address of Transfer Buffer as SEGMENT:OFFSET
21144                                   <1> DAP_LBA_Address equ 18h ; LBA=(C1*H0+H1)*S0+S1-1
21145                                   <1>                         ; C1= Selected Cylinder Number
21146                                   <1>                         ; H0= Number Of Heads (Maximum Head Number + 1)
21147                                   <1>                         ; H1= Selected Head Number
21148                                   <1>                         ; S0= Maximum Sector Number
21149                                   <1>                         ; S1= Selected Sector Number
21150                                   <1>                         ; QUAD WORD
21151                                   <1> ; DAP_Flat_Destination equ 20h ; 64 bit address, if value in 4h is FFFF:FFFFh
21152                                   <1>                               ; QUAD WORD (Also, value in 0h must be 18h)
21153                                   <1>                               ; TR-DOS will not use 64 bit Flat Address
21154                                   <1>
21155                                   <1> ; INT 13h Function 48h "Get Enhanced Disk Drive Parameters"
21156                                   <1> ; Just After DP Data (DPDiskNumber+)
21157                                   <1> GetDParams_48h equ 20h ; Word. Data Length, must be 26 (1Ah) for short data.
21158                                   <1> GDP_48h_InfoFlag equ 22h ; Word
21159                                   <1> ; Bit 1 = 1 -> The geometry returned in bytes 4-15 is valid.
21160                                   <1> GDP_48h_NumOfPCyls equ 24h ; Double Word. Number physical cylinders.
21161                                   <1> GDP_48h_NumOfPHeads equ 28h ; Double Word. Number of physical heads.
21162                                   <1> GDP_48h_NumOfPSpT equ 2Ch ; Double word. Num of physical sectors per track.
21163                                   <1> GDP_48h_LBA_Sectors equ 30h ; 8 bytes. Number of physical/LBA sectors.
21164                                   <1> GDP_48h_BytesPerSec equ 38h ; Word. Number of bytes in a sector.
21165                                   <1>
21166                                   <1> ; TR-DOS Standalone Program Extensions to the DiskParams Block
21167                                   <1> ; Just After DP Data (DPDiskNumber+)
21168                                   <1> TRDP_CurrentSector equ 3Ah  ; DX:AX (LBA)
21169                                   <1> TRDP_SectorCount equ 3Eh    ; CX (or Counter)
21170                                   <1>
21171                                   <1>
21172                                   <1> ; DOS Logical Disks
21173                                   <1> LD_Name equ 0
21174                                   <1> LD_DiskType equ 1
21175                                   <1> LD_PhyDrvNo equ 2
21176                                   <1> LD_FATType equ 3
21177                                   <1> LD_FSType equ 4
21178                                   <1> LD_LBAYes equ 5
21179                                   <1> LD_BPB equ 6
21180                                   <1> LD_FATBegin equ 96
21181                                   <1> LD_ROOTBegin equ 100
21182                                   <1> LD_DATABegin equ 104
21183                                   <1> LD_StartSector equ 108
21184                                   <1> LD_TotalSectors equ 112
21185                                   <1> LD_FreeSectors equ 116
21186                                   <1> LD_Clusters equ 120
21187                                   <1> LD_PartitionEntry equ 124
21188                                   <1> LD_DParamEntry equ 125
21189                                   <1> LD_MediaChanged equ 126
21190                                   <1> LD_CDirLevel equ 127
21191                                   <1> LD_CurrentDirectory equ 128
21192                                   <1>
21193                                   <1> ; Singlix FS Extensions to DOS Logical Disks
21194                                   <1> ; 03/01/2010 (LD_BPB compatibility for CHS r/w)
21195                                   <1>
21196                                   <1> LD_FS_Name equ 0
21197                                   <1> LD_FS_DiskType equ 1
21198                                   <1> LD_FS_PhyDrvNo equ 2
21199                                   <1> LD_FS_FATType equ 3
21200                                   <1> LD_FS_FSType equ 4
21201                                   <1> LD_FS_LBAYes equ 5
21202                                   <1> LD_FS_BPB equ 6
21203                                   <1> LD_FS_MediaAttrib equ 6
21204                                   <1> LD_FS_VersionMajor equ 7
21205                                   <1> LD_FS_RootDirD equ 8
21206                                   <1> LD_FS_MATLocation equ 12
21207                                   <1> LD_FS_Reserved1 equ 16 ;1 reserved byte
21208                                   <1> LD_FS_BytesPerSec equ 17 ; LD_BPB + 0Bh
21209                                   <1> LD_FS_Reserved2 equ 19 ;2 reserved byte
21210                                   <1> LD_FS_DATLocation equ 20
21211                                   <1> LD_FS_DATSectors equ 24
```

```
21212                                  <1> LD_FS_Reserved3 equ 28 ;3 reserved word
21213                                  <1> LD_FS_SecPerTrack equ 30 ; LD_BPB + 18h
21214                                  <1> LD_FS_NumHeads equ 32    ; LD_BPB + 1Ah
21215                                  <1> LD_FS_UnDelDirD equ 34
21216                                  <1> LD_FS_Reserved4 equ 38 ;4 reserved word
21217                                  <1> LD_FS_VolumeSerial equ 40
21218                                  <1> LD_FS_VolumeName equ 44
21219                                  <1> LD_FS_BeginSector equ 108
21220                                  <1> LD_FS_VolumeSize equ 112
21221                                  <1> LD_FS_FreeSectors equ 116
21222                                  <1> LD_FS_FirstFreeSector equ 120
21223                                  <1> LD_FS_PartitionEntry equ 124
21224                                  <1> LD_FS_DParamEntry equ 125
21225                                  <1> LD_FS_MediaChanged equ 126
21226                                  <1> LD_FS_CDirLevel equ 127
21227                                  <1> LD_FS_CDIR_Converted equ 128
21228                                  <1>
21229                                  <1> ; Valid FAT Types
21230                                  <1> FS_FAT12 equ 1
21231                                  <1> FS_FAT16_CHS equ 2
21232                                  <1> FS_FAT32_CHS equ 3
21233                                  <1> FS_FAT16_LBA equ 4
21234                                  <1> FS_FAT32_LBA equ 5
21235                                  <1>
21236                                  <1> ; Cursor Location
21237                                  <1> CCCpointer equ  0450h   ; BIOS data, current cursor column
21238                                  <1> ; FAT Clusters EOC sign
21239                                  <1> FAT12EOC equ 0FFFh
21240                                  <1> FAT16EOC equ 0FFFFh
21241                                  <1> ;FAT32EOC equ 0FFFFFFFh ; It is not direct usable for 8086 code
21242                                  <1> ; BAD Cluster
21243                                  <1> FAT12BADC equ 0FF7h
21244                                  <1> FAT16BADC equ 0FFF7h
21245                                  <1> ;FAT32BADC equ 0FFFFFF7h ; It is not direct usable for 8086 code
21246                                  <1> ; MS-DOS FAT16 FS (Maximum Possible) Last Cluster Number= 0FFF6h
21247                                  <1>
21248                                  <1> ; TRFS
21249                                  <1>
21250                                  <1> bs_FS_JmpBoot equ 0 ; jmp short bsBootCode
21251                                  <1>                     ; db 0EBh, db 3Fh, db 90h
21252                                  <1> bs_FS_Identifier equ 3  ; db 'FS', db 0
21253                                  <1> bs_FS_BytesPerSec equ 6 ; dw 512
21254                                  <1> bs_FS_MediaAttrib equ 8 ; db 3
21255                                  <1> bs_FS_PartitionID equ 9 ; db 0A1h
21256                                  <1> bs_FS_VersionMaj equ 10 ; db 01h
21257                                  <1> bs_FS_VersionMin equ 11 ; db 0
21258                                  <1> bs_FS_BeginSector equ 12   ; dd 0
21259                                  <1> bs_FS_VolumeSize equ 16 ; dd 2880
21260                                  <1> bs_FS_StartupFD equ 20 ; dd 0
21261                                  <1> bs_FS_MATLocation equ 24 ; dd 1
21262                                  <1> bs_FS_RootDirD equ 28 ; dd 8
21263                                  <1> bs_FS_SystemConfFD equ 32 ; dd 0
21264                                  <1> bs_FS_SwapFD equ 36 ; dd 0
21265                                  <1> bs_FS_UnDelDirD equ 40 ; dd 0
21266                                  <1> bs_FS_DriveNumber equ 44 ; db 0
21267                                  <1> bs_FS_LBA_Ready equ 45 ; db 0
21268                                  <1> bs_FS_MagicWord equ 46
21269                                  <1> bs_FS_SecPerTrack equ 46 ; db 0A1h
21270                                  <1> bs_FS_Heads equ 47 ; db 01h
21271                                  <1> bs_FS_OperationSys equ 48 ; db "TR-SINGLIX v1.0b"
21272                                  <1> bs_FS_Terminator equ 64 ; db 0
21273                                  <1> bs_FS_BootCode equ 65
21274                                  <1>
21275                                  <1> FS_MAT_DATLocation equ 12
21276                                  <1> FS_MAT_DATScount equ 16
21277                                  <1> FS_MAT_FreeSectors equ 20
21278                                  <1> FS_MAT_FirstFreeSector equ 24
21279                                  <1> FS_RDT_VolumeSerialNo equ 28
21280                                  <1> FS_RDT_VolumeName equ 64
21281                                  <1>
21282                                  <1> ; FAT12 + FAT16 + FAT32
21283                                  <1> BS_JmpBoot equ 0
21284                                  <1> BS_OEMName equ 3
21285                                  <1> BPB_BytsPerSec equ 11
21286                                  <1> BPB_SecPerClust equ 13
21287                                  <1> BPB_RsvdSecCnt equ 14
21288                                  <1> BPB_NumFATs equ 16
21289                                  <1> BPB_RootEntCnt equ 17
21290                                  <1> BPB_TotalSec16 equ 19
21291                                  <1> BPB_Media equ 21
21292                                  <1> BPB_FATSz16 equ 22
21293                                  <1> BPB_SecPerTrk equ 24
21294                                  <1> BPB_NumHeads equ 26
21295                                  <1> BPB_HiddSec equ 28
21296                                  <1> BPB_TotalSec32 equ 32
21297                                  <1>
21298                                  <1> ; FAT12 and FAT16 only
21299                                  <1> BS_DrvNum equ 36
21300                                  <1> BS_Reserved1 equ 37
21301                                  <1> BS_BootSig equ 38
21302                                  <1> BS_VolID equ 39
21303                                  <1> BS_VolLab equ 43
21304                                  <1> BS_FilSysType equ 54 ; 8 bytes
21305                                  <1> BS_BootCode equ 62
21306                                  <1>
21307                                  <1> ; FAT32 only
21308                                  <1> BPB_FATSz32 equ 36 ; FAT32, 4 bytes
21309                                  <1> BPB_ExtFlags equ 40 ; FAT32, 2 bytes
21310                                  <1> BPB_FSVer equ 42 ; FAT32, 2 bytes
21311                                  <1> BPB_RootClus equ 44 ; FAT32, 4 bytes
21312                                  <1> BPB_FSInfo equ 48 ; FAT 32, 2 bytes
21313                                  <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 - v2.0.0) - SYS INIT : trdosk1.s
                        <1> ; ----------------------------------------------------------------------------
                        <1> ; Last Update: 23/01/2017
                        <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>
                        <1> sys_init:
                        <1>       ; 23/01/2017
                        <1>       ; 07/05/2016
                        <1>       ; 02/05/2016
                        <1>       ; 24/04/2016
                        <1>       ; 14/04/2016
                        <1>       ; 13/04/2016
                        <1>       ; 30/03/2016
                        <1>       ; 24/01/2016
                        <1>       ; 06/01/2016
                        <1>       ; 04/01/2016
                        <1>
                        <1>       ; 23/01/2017 - reset timer frequency (to 18.2Hz)
0000624E B036           <1>       mov    al, 00110110b ; 36h
00006250 E643           <1>       out    43h, al
00006252 31C0           <1>       xor    eax, eax  ; sub     al, al ; 0
00006254 E640           <1>       out    40h, al ; LB
00006256 E640           <1>       out    40h, al ; HB
                        <1>       ;
                        <1>       ; 30/03/2016
                        <1>       ; Clear Logical DOS Disk Description Tables Area
                        <1>       ;xor   eax, eax
00006258 BF00010900     <1>       mov    edi, Logical_DOSDisks
0000625D B980060000     <1>       mov    ecx, 6656/4 ; 26*256 = 6656 bytes
00006262 F3AB           <1>       rep    stosd ; 1664 times 4 bytes
                        <1>
00006264 B83F3A2F00     <1>       mov    eax, '?:/'
00006269 A3[8F4E0100]   <1>       mov    [Current_Dir_Drv], eax
                        <1>
                        <1>       ; Logical DRV INIT (only for hard disks)
0000626E E8B3010000     <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>
00006273 BE00010900     <1>       mov    esi, Logical_DOSDisks
00006278 B001           <1>       mov    al, 1 ; Initialization sign (invalid_fd_parameter)
0000627A 83C67E         <1>       add    esi, LD_MediaChanged ; Media Change Status = 1 (init needed)
0000627D 8806           <1>       mov    [esi], al ; A:
0000627F 81C600010000   <1>       add    esi, 100h
00006285 8806           <1>       mov    [esi], al ; B:
                        <1>
                        <1> _current_drive_bootdisk:
00006287 8A15[F25C0000] <1>       mov    dl, [boot_drv] ; physical drive number
0000628D 80FAFF         <1>       cmp    dl, 0FFh
00006290 740A           <1>       je     short _last_dos_diskno_check
                        <1> _boot_drive_check:
00006292 80FA80         <1>       cmp    dl, 80h
00006295 7218           <1>       jb     short _current_drive_a
00006297 80EA7E         <1>       sub    dl, 7Eh ; C = 2 , D = 3
0000629A EB13           <1>       jmp    short _current_drive_a
                        <1>
                        <1> _last_dos_diskno_check:
```

```
21416 0000629C 8A15[97020100]      <1>       mov    dl, [Last_DOS_DiskNo]
21417 000062A2 80FA02              <1>       cmp    dl, 2
21418 000062A5 7706                <1>       ja     short _current_drive_c
21419 000062A7 7406                <1>       je     short _current_drive_a
21420 000062A9 30D2                <1>       xor    dl, dl ; A:
21421 000062AB EB02                <1>       jmp    short _current_drive_a
21422                              <1>
21423                              <1> _current_drive_c:
21424 000062AD B202                <1>       mov    dl, 2 ; C:
21425                              <1>
21426                              <1> _current_drive_a:
21427 000062AF 8815[F35C0000]      <1>       mov    [drv], dl
21428 000062B5 BE[99020100]        <1>         mov     esi, msg_CRLF_temp
21429 000062BA E89E000000          <1>       call   print_msg
21430                              <1>
21431 000062BF 8A15[F35C0000]      <1>       mov    dl, [drv]
21432 000062C5 E893090000          <1>       call   change_current_drive
21433 000062CA 730C                <1>       jnc    short _start_mainprog
21434                              <1>
21435                              <1> _drv_not_ready_error:
21436 000062CC BE[54050100]        <1>       mov    esi, msgl_drv_not_ready
21437 000062D1 E887000000          <1>       call   print_msg
21438 000062D6 EB63                <1>         jmp     _end_of_mainprog
21439                              <1>
21440                              <1> _start_mainprog:
21441                              <1>       ; 07/01/2017
21442                              <1>       ; 07/05/2016
21443                              <1>       ; 02/05/2016
21444                              <1>       ; 24/04/2016
21445                              <1>       ; Retro UNIX 386 v1, 'sys_init' (u0.s)
21446                              <1>       ; 23/06/2015
21447                              <1>
21448                              <1>       ; 02/05/2016
21449                              <1>       ; 24/04/2016
21450 000062D8 66B80100            <1>       mov    ax, 1
21451 000062DC A2[B3030600]        <1>       mov    [u.uno], al
21452 000062E1 66A3[4E030600]      <1>       mov    [mpid], ax
21453 000062E7 66A3[20000600]      <1>       mov    [p.pid], ax
21454 000062ED A2[B0000600]        <1>       mov    [p.stat], al
21455 000062F2 C605[A8030600]04    <1>       mov    byte [u.quant], time_count  ; 07/01/2017
21456                              <1>       ;
21457 000062F9 A1[C84D0100]        <1>       mov    eax, [k_page_dir]
21458 000062FE A3[B8030600]        <1>       mov    [u.pgdir], eax ; reset
21459                              <1>       ;
21460 00006303 E871E8FFFF          <1>       call   allocate_page
21461 00006308 0F82A3000000        <1>       jc     panic
21462 0000630E A3[B4030600]        <1>       mov    [u.upage], eax ; user structure page
21463 00006313 A3[C0000600]        <1>       mov    [p.upage], eax
21464 00006318 E8D6E8FFFF          <1>       call   clear_page
21465                              <1>       ;
21466                              <1>       ; 24/08/2015
21467 0000631D FE0D[5B030600]      <1>       dec    byte [sysflg] ; FFh = ready for system call
21468                              <1>                            ; 0 = executing a system call
21469                              <1>       ; 13/04/2016
21470                              <1>       ; Clear Environment Variables Page/Area
21471 00006323 BF00300900          <1>       mov    edi, Env_Page ; 93000h
21472 00006328 B980000000          <1>       mov    ecx, Env_Page_Size / 4    ; 512/4  (4096/4)

21473 0000632D 31C0                <1>       xor    eax, eax
21474 0000632F F3AB                <1>       rep    stosd
21475                              <1>
21476                              <1>       ; 14/04/2016
21477 00006331 E8C3320000          <1>       call   mainprog_startup_configuration
21478                              <1>
21479 00006336 E8630A0000          <1>         call    dos_prompt
21480                              <1>
21481                              <1> _end_of_mainprog:
21482 0000633B BE[99020100]        <1>         mov     esi, msg_CRLF_temp
21483 00006340 E818000000          <1>       call   print_msg
21484 00006345 BE[9F020100]        <1>       mov    esi, mainprog_Version
21485 0000634A E80E000000          <1>       call   print_msg
21486                              <1>       ; 24/01/2016
21487 0000634F 28E4                <1>       sub    ah, ah
21488 00006351 E8C0A8FFFF          <1>       call   int16h ; call getch
21489 00006356 E9A0ADFFFF          <1>       jmp    cpu_reset
21490                              <1>
21491 0000635B EBFE                <1> infinitiveloop: jmp short infinitiveloop
21492                              <1>
21493                              <1> print_msg:
21494                              <1>       ; 13/05/2016
21495                              <1>       ; 04/01/2016
21496                              <1>       ; 01/07/2015
21497                              <1>       ; 13/03/2015 (Retro UNIX 386 v1)
21498                              <1>       ; 07/03/2014 (Retro UNIX 8086 v1)
21499                              <1>       ; (Modified registers: EAX, EBX, ECX, EDX, ESI, EDI)
21500                              <1>       ;
21501 0000635D 8A3D[F64D0100]      <1>       mov    bh, [ACTIVE_PAGE] ; 04/01/2016 (ptty)
21502                              <1>       ;mov   bl, 07h ; Black background, light gray forecolor
21503                              <1>
21504 00006363 AC                  <1>       lodsb
21505                              <1> pmsg1:
21506 00006364 56                  <1>       push   esi
21507                              <1>       ;mov   bh, [ACTIVE_PAGE] ; 04/01/2016 (ptty)
21508 00006365 B307                <1>       mov    bl, 07h ; Black background, light gray forecolor
21509 00006367 E846B9FFFF          <1>       call   _write_tty
21510 0000636C 5E                  <1>       pop    esi
21511 0000636D AC                  <1>       lodsb
21512 0000636E 20C0                <1>       and    al, al
21513 00006370 75F2                <1>       jnz    short pmsg1
21514 00006372 C3                  <1>       retn
21515                              <1>
21516                              <1> clear_screen:
```

```
21517                                  <1>       ; 13/05/2016
21518                                  <1>       ; 30/01/2016
21519                                  <1>       ; 24/01/2016
21520                                  <1>       ; 04/01/2016
21521 00006373 0FB61D[F64D0100]        <1>       movzx  ebx, byte [ACTIVE_PAGE] ; video page number (0 to 7)
21522 0000637A 8AA3[D35E0000]          <1>       mov    ah, [ebx+vmode] ; default = 03h (80x25 text)
21523 00006380 80FC04                  <1>       cmp    ah, 4
21524 00006383 7205                    <1>       jb     short cls1
21525 00006385 80FC07                  <1>       cmp    ah, 7
21526 00006388 7526                    <1>       jne    short vga_clear
21527                                  <1> cls1:
21528                                  <1>       ;mov   bh, bl
21529                                  <1>       ;mov   bl, 7
21530 0000638A 3A25[C25E0000]          <1>       cmp    ah, [CRT_MODE] ; current video mode ?
21531                                  <1>       ;je    short cls2 ; yes (current video mode = 3)
21532                                  <1>       ;;call set_mode_3 ; set video mode to 3 (& clear screen)
21533                                  <1>       ;;retn
21534                                  <1>       ;jmp   set_mode_3
21535 00006390 0F8526B9FFFF            <1>       jne    set_mode_3
21536                                  <1> cls2:
21537 00006396 88DF                    <1>       mov    bh, bl ; video page (0 to 7)
21538 00006398 B307                    <1>       mov    bl, 07h ; attribute to be used on blanked line
21539 0000639A 28C0                    <1>       sub    al, al ; 0 =  entire window
21540 0000639C 6631C9                  <1>       xor    cx, cx
21541 0000639F 66BA4F18                <1>       mov    dx, 184Fh
21542 000063A3 E862B6FFFF              <1>       call   _scroll_up ; 24/01/2016
21543                                  <1>       ;
21544                                  <1>       ;mov   bh, [ACTIVE_PAGE] ; video page number (0 to 7)
21545 000063A8 6631D2                  <1>       xor    dx, dx
21546 000063AB E898B9FFFF              <1>       call   _set_cpos ; 24/01/2016
21547                                  <1>       ;retn
21548                                  <1> vga_clear:
21549 000063B0 C3                      <1>       retn
21550                                  <1>
21551                                  <1> panic:
21552                                  <1>       ; 13/05/2016 (TRDOS 386 = TRDOS v2)
21553                                  <1>       ; 13/03/2015 (Retro UNIX 386 v1)
21554                                  <1>       ; 07/03/2014 (Retro UNIX 8086 v1)
21555 000063B1 BE[060F0100]            <1>       mov    esi, panic_msg
21556 000063B6 E8A2FFFFFF              <1>       call   print_msg
21557                                  <1> key_to_reboot:
21558                                  <1>         ; 24/01/2016
21559 000063BB 28E4                    <1>         sub     ah, ah
21560 000063BD E854A8FFFF              <1>         call    int16h ; call   getch
21561                                  <1>         ; wait for a character from the current tty
21562                                  <1>       ;
21563 000063C2 B00A                    <1>       mov    al, 0Ah
21564 000063C4 8A3D[F64D0100]          <1>       mov    bh, [ptty] ; [ACTIVE_PAGE]
21565 000063CA B307                    <1>       mov    bl, 07h ; Black background,
21566                                  <1>                    ; light gray forecolor
21567 000063CC E8E1B8FFFF              <1>       call   _write_tty
21568 000063D1 E925ADFFFF              <1>       jmp    cpu_reset
21569                                  <1>
21570                                  <1> ctrlbrk:
21571                                  <1>       ; 12/11/2015
21572                                  <1>       ; 13/03/2015 (Retro UNIX 386 v1)
21573                                  <1>       ; 06/12/2013 (Retro UNIX 8086 v1)
21574                                  <1>       ;
21575                                  <1>       ; INT 1Bh (control+break) handler
21576                                  <1>       ;
21577                                  <1>             ; Retro Unix 8086 v1 feature only!
21578                                  <1>             ;
21579 000063D6 66833D[AA030600]00      <1>       cmp    word [u.intr], 0
21580 000063DE 7645                    <1>       jna    short cbrk4
21581                                  <1> cbrk0:
21582                                  <1>       ; 12/11/2015
21583                                  <1>       ; 06/12/2013
21584 000063E0 66833D[AC030600]00      <1>       cmp    word [u.quit], 0
21585 000063E8 743B                    <1>       jz     short cbrk4
21586                                  <1>       ;
21587                                  <1>       ; 20/09/2013
21588 000063EA 6650                    <1>       push   ax
21589 000063EC A0[F64D0100]            <1>       mov    al, [ptty]
21590                                  <1>       ;
21591                                  <1>       ; 12/11/2015
21592                                  <1>       ;
21593                                  <1>       ; ctrl+break (EOT, CTRL+D) from serial port
21594                                  <1>       ; or ctrl+break from console (pseudo) tty
21595                                  <1>       ; (!redirection!)
21596                                  <1>       ;
21597 000063F1 3C08                    <1>       cmp    al, 8 ; serial port tty nums > 7
21598 000063F3 7211                    <1>         jb      short cbrk1 ; console (pseudo) tty
21599                                  <1>       ;
21600                                  <1>       ; Serial port interrupt handler sets [ptty]
21601                                  <1>       ; to the port's tty number (as temporary).
21602                                  <1>       ;
21603                                  <1>       ; If active process is using a stdin or
21604                                  <1>       ; stdout redirection (by the shell),
21605                                  <1>         ; console tty keyboard must be available
21606                                  <1>       ; to terminate running process,
21607                                  <1>       ; in order to prevent a deadlock.
21608                                  <1>       ;
21609 000063F5 52                      <1>       push   edx
21610 000063F6 0FB615[B3030600]        <1>       movzx  edx, byte [u.uno]
21611 000063FD 3A82[7F000600]          <1>       cmp     al, [edx+p.ttyc-1] ; console tty (rw)
21612 00006403 5A                      <1>       pop    edx
21613 00006404 7412                    <1>       je     short cbrk2
21614                                  <1> cbrk1:
21615 00006406 FEC0                    <1>       inc    al  ; [u.ttyp] : 1 based tty number
21616                                  <1>       ; 06/12/2013
21617 00006408 3A05[94030600]          <1>       cmp    al, [u.ttyp] ; recent open tty (r)
21618 0000640E 7408                    <1>       je     short cbrk2
```

```
21619 00006410 3A05[95030600]      <1>         cmp     al, [u.ttyp+1] ; recent open tty (w)
21620 00006416 750B                <1>       jne    short cbrk3
21621                              <1> cbrk2:
21622                              <1>       ;; 06/12/2013
21623                              <1>       ;mov   ax, [u.quit]
21624                              <1>       ;and   ax, ax
21625                              <1>       ;jz    short cbrk3
21626                              <1>       ;
21627 00006418 6631C0              <1>       xor    ax, ax ; 0
21628 0000641B 6648                <1>       dec    ax
21629                              <1>       ; 0FFFFh = 'ctrl+brk' keystroke
21630 0000641D 66A3[AC030600]      <1>       mov    [u.quit], ax
21631                              <1> cbrk3:
21632 00006423 6658                <1>       pop    ax
21633                              <1> cbrk4:
21634 00006425 C3                  <1>       retn
21635                                  %include 'trdosk2.s' ; 04/01/2016
21636                              <1> ; ****************************************************************************
21637                              <1> ; TRDOS386.ASM (TRDOS 386 Kernel - v2.0.0) - DRV INIT : trdosk2.s
21638                              <1> ; ----------------------------------------------------------------------------
21639                              <1> ; Last Update: 06/07/2016
21640                              <1> ; ----------------------------------------------------------------------------
21641                              <1> ; Beginning: 04/01/2016
21642                              <1> ; ----------------------------------------------------------------------------
21643                              <1> ; Assembler: NASM version 2.11 (trdos386.s)
21644                              <1> ; ----------------------------------------------------------------------------
21645                              <1> ; Derived from TRDOS Operating System v1.0 (8086) source code by Erdogan Tan
21646                              <1> ; TRDOS2.ASM (09/11/2011)
21647                              <1> ; ****************************************************************************
21648                              <1> ; DRV_INIT.ASM (c) 2009-2011 Erdogan TAN  [26/09/2009] Last Update: 07/08/2011
21649                              <1> ;
21650                              <1>
21651                              <1> ldrv_init: ; Logical Drive Initialization
21652                              <1>       ; 12/02/2016
21653                              <1>       ; 06/01/2016
21654                              <1>       ;      ('diskinit.inc', 'diskio.inc' integration)
21655                              <1>       ; 04/01/2016 (TRDOS 386 = TRDOS v2.0)
21656                              <1>       ; 07/08/2011
21657                              <1>       ; 20/09/2009
21658                              <1>       ; 2005
21659 00006426 0FB60D[644E0100]    <1>       movzx  ecx, byte [HF_NUM] ; number of fixed disks
21660 0000642D 80F901              <1>       cmp    cl, 1
21661 00006430 7301                <1>       jnb    short load_hd_partition_tables
21662                              <1>       ; No hard disks
21663 00006432 C3                  <1>       retn
21664                              <1> load_hd_partition_tables:
21665 00006433 8B35[684E0100]      <1>       mov    esi, [HDPM_TBL_VEC] ; primary master disk FDPT
21666 00006439 BF[8E520100]        <1>       mov    edi, PTable_hd0
21667 0000643E B280                <1>       mov    dl, 80h
21668                              <1> load_next_hd_partition_table:
21669 00006440 51                  <1>       push   ecx
21670 00006441 57                  <1>       push   edi
21671 00006442 56                  <1>       push   esi ; FDPT (+ DPTE) address
21672 00006443 8A4614              <1>       mov    al, [esi+20] ; DPTE offset 4
21673 00006446 2440                <1>       and    al, 40h ;  LBA bit (bit 6)
21674                              <1>       ;shr   al, 6
21675 00006448 A2[8F540100]        <1>       mov    [HD_LBA_yes], al
21676 0000644D E81C040000          <1>       call   load_masterboot
21677 00006452 7275                <1>       jc     short pass_pt_this_hard_disk
21678                              <1>
21679 00006454 BE[4C520100]        <1>       mov    esi, PartitionTable
21680 00006459 89F3                <1>       mov    ebx, esi
21681                              <1>       ;mov   ecx, 16
21682 0000645B B110                <1>       mov    cl, 16
21683 0000645D F3A5                <1>       rep    movsd
21684 0000645F 89DE                <1>       mov    esi, ebx
21685 00006461 C605[F55C0000]04    <1>       mov    byte [hdc], 4 ; 4 - partition index
21686                              <1> loc_validate_hdp_partition:
21687 00006468 807E0400            <1>       cmp    byte [esi+ptFileSystemID], 0
21688 0000646C 7641                <1>       jna    short loc_validate_next_hdp_partition2
21689 0000646E 56                  <1>       push   esi ; Masterboot partition table offset
21690 0000646F 52                  <1>       push   edx ; dl = Physical drive number
21691 00006470 FE05[90540100]      <1>       inc    byte [PP_Counter]
21692 00006476 31FF                <1>         xor  edi, edi ; 0
21693                              <1>       ; Input -> ESI = PartitionTable offset
21694                              <1>       ; DL = Hard disk drive number
21695                              <1>       ; EDI = 0 -> Primary Partition
21696                              <1>       ; EDI > 0 -> Extended Partition's Start Sector
21697 00006478 E879010000          <1>       call   validate_hd_fat_partition
21698 0000647D 730A                <1>       jnc    short loc_set_valid_hdp_partition_entry
21699                              <1>       ;pop   edx
21700                              <1>       ;push  edx
21701 0000647F 8B1424              <1>       mov    edx, [esp]
21702 00006482 E8C5020000          <1>       call   validate_hd_fs_partition
21703 00006487 7224                <1>       jc     short loc_validate_next_hdp_partition1
21704                              <1> loc_set_valid_hdp_partition_entry:
21705 00006489 8A0D[97020100]      <1>       mov    cl, [Last_DOS_DiskNo]
21706 0000648F 80C141              <1>       add    cl, 'A'
21707                              <1>       ; ESI = Logical dos drive description table address
21708 00006492 880E                <1>       mov    [esi+LD_Name], cl
21709 00006494 8A6602              <1>       mov    ah, [esi+LD_PhyDrvNo]
21710 00006497 88E0                <1>       mov    al, ah ; Physical drive number
21711 00006499 2C80                <1>       sub    al, 80h
21712 0000649B C0E002              <1>       shl    al, 2
21713 0000649E 0404                <1>       add    al, 4 ; 0 Based
21714 000064A0 2A05[F55C0000]      <1>       sub    al, [hdc] ; 4 - partition index
21715                              <1>       ; AL = Partition entry/index, 0 based
21716                              <1>       ;  0 -> hd 0, Partition Table offset = 0
21717                              <1>       ; 15 -> hd 3, Partition Table offset = 3
21718                              <1>       ;mov   [esi+LD_PartitionEntry], al
21719 000064A6 80EC7E              <1>       sub    ah, 7Eh
21720                              <1>       ; AH = Physical drive index, zero based
```

```
21721                                  <1>       ;  0 for drive A:, 2 for drive C:
21722                                  <1>       ;mov   [esi+LD_DParamEntry], ah
21723 000064A9 6689467C                <1>       mov    [esi+LD_PartitionEntry], ax
21724                                  <1> loc_validate_next_hdp_partition1:
21725 000064AD 5A                      <1>       pop    edx ; dl = Physical drive number
21726 000064AE 5E                      <1>       pop    esi ; Masterboot partition table offset
21727                                  <1> loc_validate_next_hdp_partition2:
21728                                  <1>       ; ESI = PartitionTable offset
21729                                  <1>       ; DL = Hard/Fixed disk drive number
21730 000064AF FE0D[F55C0000]          <1>       dec    byte [hdc] ; 4 - partition index
21731 000064B5 7412                    <1>       jz     short pass_pt_this_hard_disk
21732 000064B7 83C610                  <1>       add    esi, 16 ; 10h
21733 000064BA EBAC                    <1>       jmp    short loc_validate_hdp_partition
21734                                  <1> loc_next_hd_partition_table:
21735 000064BC FEC2                    <1>       inc    dl
21736 000064BE 83C620                  <1>       add    esi, 32 ; next FDPT address
21737 000064C1 83C740                  <1>       add    edi, 64 ; next partition table destination
21738 000064C4 E977FFFFFF              <1>         jmp     load_next_hd_partition_table
21739                                  <1> pass_pt_this_hard_disk:
21740 000064C9 5E                      <1>       pop    esi ; FDPT (+ DPTE) address
21741 000064CA 5F                      <1>       pop    edi ; Ptable_hd?
21742 000064CB 59                      <1>       pop    ecx
21743 000064CC E2EE                    <1>       loop   loc_next_hd_partition_table
21744 000064CE 803D[90540100]01        <1>       cmp    byte [PP_Counter], 1
21745 000064D5 7301                    <1>       jnb    short load_extended_dos_partitions
21746                                  <1>       ; Empty partition table
21747 000064D7 C3                      <1>       retn
21748                                  <1> load_extended_dos_partitions:
21749 000064D8 BE[8E520100]            <1>       mov    esi, PTable_hd0
21750 000064DD BF[8E530100]            <1>       mov    edi, PTable_ep0
21751 000064E2 C605[F55C0000]80        <1>       mov    byte [hdc], 80h
21752                                  <1> next_hd_extd_partition:
21753 000064E9 56                      <1>       push   esi ; PTable_hd? offset
21754 000064EA 57                      <1>       push   edi ; PTable_ep?
21755                                  <1>       ;mov   ecx, 4
21756 000064EB B104                    <1>       mov    cl, 4
21757 000064ED 8A15[F55C0000]          <1>       mov    dl, byte [hdc]
21758                                  <1> hd_check_fs_id_05h:
21759 000064F3 8A4604                  <1>       mov    al, [esi+ptFileSystemID]
21760 000064F6 3C05                    <1>       cmp    al, 05h ; Is it an extended dos partition ?
21761 000064F8 7404                    <1>       je     short loc_set_ep_start_sector
21762 000064FA 3C0F                    <1>       cmp    al, 0Fh ; Is it an extended win4 (LBA mode) partition ?
21763 000064FC 7546                    <1>       jne    short continue_to_check_ep
21764                                  <1> loc_set_ep_start_sector:
21765 000064FE FE05[91540100]          <1>       inc    byte [EP_Counter]
21766 00006504 88D4                    <1>       mov    ah, dl ; byte [hdc]
21767 00006506 86E0                    <1>       xchg   ah, al ; al = Drv Number, ah = Partition Identifier
21768 00006508 50                      <1>       push   eax
21769 00006509 30E4                    <1>       xor    ah, ah
21770 0000650B 2C80                    <1>       sub    al, 80h
21771 0000650D 50                      <1>       push   eax
21772 0000650E C0E002                  <1>       shl    al, 2 ; al = al * 4
21773 00006511 0FB6D8                  <1>       movzx  ebx, al
21774 00006514 81C3[92540100]          <1>       add    ebx, EP_StartSector
21775 0000651A 8B4608                  <1>       mov    eax, [esi+ptStartSector]
21776                                  <1>         ; EAX = Extended partition's start sector
21777 0000651D 8903                    <1>         mov [ebx], eax
21778 0000651F 58                      <1>       pop    eax ; AL = Drv number - 80h, AH = 0
21779 00006520 5A                      <1>       pop    edx ; DL = Drv number, DH = Partition ID
21780 00006521 BB[8E500100]            <1>       mov    ebx, MasterBootBuff
21781 00006526 803D[8F540100]01        <1>       cmp    byte [HD_LBA_yes], 1 ; LBA ready = Yes
21782 0000652D 7240                    <1>       jb     short loc_hd_load_ep_05h
21783 0000652F 80FE05                  <1>       cmp    dh, 05h
21784 00006532 743B                    <1>       je     short loc_hd_load_ep_05h
21785                                  <1> loc_hd_load_ep_0Fh:
21786                                  <1>       ; 04/01/2016
21787 00006534 51                      <1>       push   ecx
21788 00006535 8B4E08                  <1>       mov    ecx, [esi+ptStartSector] ; sector number
21789                                  <1>       ;mov   ebx, MasterBootBuff ; buffer address
21790                                  <1>       ; LBA read/write (with private LBA function)
21791                                  <1>       ;((Retro UNIX 386 v1 - DISK I/O code by Erdogan Tan))
21792                                  <1>       ; dl = physical drive number (0,1, 80h, 81h, 82h, 83h)
21793 00006538 B41B                    <1>       mov    ah, 1Bh ; LBA read
21794 0000653A B001                    <1>       mov    al, 1 ; sector count
21795 0000653C E8C5DCFFFF              <1>       call   int13h
21796 00006541 59                      <1>       pop    ecx
21797 00006542 733F                    <1>       jnc    short loc_hd_move_ep_table
21798                                  <1> continue_to_check_ep:
21799 00006544 83C610                  <1>       add    esi, 16
21800 00006547 E2AA                    <1>       loop   hd_check_fs_id_05h
21801                                  <1> continue_check_ep_next_disk:
21802 00006549 5F                      <1>       pop    edi ; PTable_ep?
21803 0000654A 5E                      <1>       pop    esi ; PTable_hd?
21804 0000654B A0[644E0100]            <1>       mov    al, [HF_NUM] ; number of hard disks
21805 00006550 047F                    <1>       add    al, 7Fh
21806 00006552 3805[F55C0000]          <1>       cmp    [hdc], al
21807 00006558 0F8392000000            <1>       jnb    loc_validating_hd_partitions_ok
21808 0000655E 83C640                  <1>       add    esi, 64
21809 00006561 83C740                  <1>       add    edi, 64
21810 00006564 FE05[F55C0000]          <1>       inc    byte [hdc]
21811 0000656A E97AFFFFFF              <1>       jmp    next_hd_extd_partition
21812                                  <1> loc_hd_load_ep_05h:
21813 0000656F 51                      <1>       push   ecx
21814 00006570 8A7601                  <1>       mov    dh, [esi+ptBeginHead]
21815 00006573 668B4E02                <1>         mov     cx, word [esi+ptBeginSector]
21816 00006577 66B80102                <1>       mov    ax, 0201h ; Read 1 sector
21817                                  <1>       ;mov   ebx, MasterBootBuff
21818 0000657B E886DCFFFF              <1>       call   int13h
21819 00006580 59                      <1>       pop    ecx
21820 00006581 72C1                    <1>       jc     short continue_to_check_ep
21821                                  <1> loc_hd_move_ep_table:
21822                                  <1>         ;pop edi
```

```
21823                                  <1>       ;push  edi  ; PTable_ep?
21824 00006583 8B3C24                  <1>       mov    edi, [esp]
21825 00006586 BE[4C520100]            <1>         mov  esi, PartitionTable ; Extended
21826 0000658B 89F3                    <1>       mov    ebx, esi
21827                                  <1>       ;mov   ecx, 16
21828 0000658D B110                    <1>       mov    cl, 16
21829 0000658F F3A5                    <1>              rep    movsd
21830 00006591 89DE                    <1>       mov    esi, ebx
21831                                  <1> loc_set_hde_sub_partition_count:
21832 00006593 C605[90540100]04        <1>       mov    byte [PP_Counter], 4
21833                                  <1> loc_validate_hde_partition:
21834 0000659A 807E0400                <1>       cmp    byte [esi+ptFileSystemID], 0
21835 0000659E 763F                    <1>       jna    short loc_validate_next_hde_partition2
21836 000065A0 56                      <1>       push   esi ; Extended partition table offset
21837 000065A1 8A15[F55C0000]          <1>       mov    dl, byte [hdc]
21838 000065A7 0FB6C2                  <1>       movzx  eax, dl
21839 000065AA 2C80                    <1>       sub    al, 80h
21840 000065AC C0E002                  <1>       shl    al, 2
21841                                  <1>       ; 06/01/2016
21842                                  <1>       ; (TRDOS v1.0 had a bug here, in 'DRV_INIT.ASM')
21843                                  <1>       ; BUGFIX *
21844                                  <1>       ;mov   ecx, eax
21845 000065AF 88C1                    <1>       mov    cl, al
21846 000065B1 80C104                  <1>       add    cl, 4
21847 000065B4 2A0D[90540100]          <1>       sub    cl, [PP_Counter] ; 4 to 1
21848                                  <1>       ; CL = Partition entry/index, 0 based
21849                                  <1>         ;  0 -> hd 0, Partition Table offset = 0
21850                                  <1>         ; 15 -> hd 3, Partition Table offset = 3
21851 000065BA 88D5                    <1>              mov    ch, dl
21852 000065BC 80ED7E                  <1>       sub    ch, 7Eh ;
21853                                  <1>       ; CH = Physical drive index, zero based
21854                                  <1>       ;  0 for drive A:, 2 for drive C:
21855                                  <1>       ; BUGFIX *
21856 000065BF 51                      <1>       push   ecx ; *
21857 000065C0 BF[92540100]            <1>       mov    edi, EP_StartSector
21858 000065C5 01C7                    <1>       add    edi, eax
21859                                  <1>       ; Input -> ESI = PartitionTable offset
21860                                  <1>       ; DL = Hard disk drive number
21861                                  <1>       ; EDI = Extended partition start sector pointer
21862 000065C7 E82A000000              <1>       call   validate_hd_fat_partition
21863 000065CC 59                      <1>       pop    ecx ; *
21864 000065CD 720F                    <1>       jc     short loc_validate_next_hde_partition1
21865                                  <1> loc_set_valid_hde_partition_entry:
21866                                  <1>       ; 06/01/2016 (TRDOS v2.0)
21867                                  <1>       ; BUGFIX *
21868                                  <1>       ;mov   [esi+LD_PartitionEntry], cl
21869                                  <1>       ;mov   [esi+LD_DParamEntry], ch
21870 000065CF 66894E7C                <1>       mov    [esi+LD_PartitionEntry], cx
21871                                  <1>       ;
21872 000065D3 8A0D[97020100]          <1>       mov    cl, [Last_DOS_DiskNo]
21873 000065D9 80C141                  <1>       add    cl, 'A'
21874 000065DC 880E                    <1>       mov    [esi+LD_Name], cl
21875                                  <1> loc_validate_next_hde_partition1:
21876 000065DE 5E                      <1>       pop    esi ; Extended partition table offset
21877                                  <1> loc_validate_next_hde_partition2:
21878                                  <1>       ; ESI = Extended partition table offset
21879                                  <1>       ; DL = Hard disk drive number
21880 000065DF FE0D[90540100]          <1>       dec    byte [PP_Counter]
21881 000065E5 0F845EFFFFFF            <1>       jz     continue_check_ep_next_disk
21882 000065EB 83C610                  <1>       add    esi, 16 ; 10h
21883 000065EE EBAA                    <1>       jmp    short loc_validate_hde_partition
21884                                  <1> loc_validating_hd_partitions_ok:
21885 000065F0 A0[97020100]            <1>       mov    al, [Last_DOS_DiskNo]
21886                                  <1> loc_drv_init_retn:
21887 000065F5 C3                      <1>       retn
21888                                  <1>
21889                                  <1> validate_hd_fat_partition:
21890                                  <1>       ; 12/02/2016
21891                                  <1>       ; 07/01/2016 (TRDOS 386 = TRDOS v2.0)
21892                                  <1>       ; 07/08/2011
21893                                  <1>       ; 23/07/2011
21894                                  <1>       ; Input
21895                                  <1>       ;   DL = Hard/Fixed Disk Drive Number
21896                                  <1>       ;   ESI = PartitionTable offset
21897                                  <1>       ;   EDI = Extend. Part. Start Sector Pointer
21898                                  <1>       ;   EDI = 0 -> Primary Partition
21899                                  <1>       ;   byte [Last_DOS_DiskNo]
21900                                  <1>       ; Output
21901                                  <1>       ;  cf=0 -> Validated
21902                                  <1>       ;   ESI = Logical dos drv desc. table
21903                                  <1>       ;   EBX = FAT boot sector buffer
21904                                  <1>       ;   byte [Last_DOS_DiskNo]
21905                                  <1>       ;  cf=1 -> Not a valid FAT partition
21906                                  <1>       ; EAX, EDX, ECX, EDI -> changed
21907                                  <1>
21908                                  <1>       ;mov   esi, PartitionTable
21909 000065F6 8A6604                  <1>       mov    ah, [esi+ptFileSystemID]
21910 000065F9 80FC06                  <1>       cmp    ah, 06h ; FAT16 CHS partition
21911                                  <1>       ; 12/02/2016
21912                                  <1>       ;jb    short loc_not_a_valid_fat_partition2
21913 000065FC 7305                    <1>       jnb    short vhdp_FAT16_32
21914                                  <1>       ;
21915 000065FE 80FC04                  <1>       cmp    ah, 04h ; FAT16 CHS partition (< 32MB)
21916 00006601 7519                    <1>       jne    short loc_not_a_valid_fat_partition1
21917                                  <1> vhdp_FAT16_32:
21918 00006603 B002                    <1>       mov    al, 2
21919 00006605 7417                    <1>       je     short loc_set_valid_hd_partition_params
21920 00006607 80FC0E                  <1>       cmp    ah, 0Eh ; FAT16 LBA partition
21921 0000660A 7710                    <1>       ja     short loc_not_a_valid_fat_partition1
21922 0000660C 7410                    <1>       je     short loc_set_valid_hd_partition_params
21923                                  <1>
21924 0000660E FEC0                    <1>       inc    al ; 3
```

```
21925 00006610 80FC0B              <1>       cmp    ah, 0Bh ; FAT32 CHS partition
21926 00006613 7409                <1>       je     short loc_set_valid_hd_partition_params
21927 00006615 7206                <1>       jb     short loc_not_a_valid_fat_partition2
21928 00006617 80FC0C              <1>       cmp    ah, 0Ch ; FAT32 LBA partition
21929 0000661A 7402                <1>       je     short loc_set_valid_hd_partition_params
21930                              <1> loc_not_a_valid_fat_partition1:
21931 0000661C F9                  <1>       stc
21932                              <1> loc_not_a_valid_fat_partition2:
21933 0000661D C3                  <1>       retn
21934                              <1>
21935                              <1> loc_set_valid_hd_partition_params:
21936 0000661E FE05[97020100]      <1>       inc    byte [Last_DOS_DiskNo] ; > 1
21937                              <1>       ;
21938 00006624 31DB                <1>       xor    ebx, ebx
21939 00006626 8A3D[97020100]      <1>       mov    bh, [Last_DOS_DiskNo] ; * 256
21940 0000662C 81C300010900        <1>       add    ebx, Logical_DOSDisks
21941                              <1>       ;
21942 00006632 C6430102            <1>       mov    byte [ebx+LD_DiskType], 2
21943 00006636 885302              <1>       mov    byte [ebx+LD_PhyDrvNo], dl
21944                              <1>       ;mov   byte [ebx+LD_FATType], al ; 2 or 3
21945                              <1>       ;mov   byte [ebx+LD_FSType], ah ; 06h, 0Eh, 0Bh, 0Ch
21946 00006639 66894303           <1>       mov    word [ebx+LD_FATType], ax
21947                              <1>       ;
21948 0000663D 8B4E08              <1>       mov    ecx, [esi+ptStartSector]
21949 00006640 09FF                <1>       or     edi, edi
21950 00006642 7402                <1>       jz     short pass_hd_FAT_ep_start_sector_adding
21951                              <1> loc_add_hd_FAT_ep_start_sector:
21952 00006644 030F                <1>       add    ecx, [edi]
21953                              <1> pass_hd_FAT_ep_start_sector_adding:
21954 00006646 894B6C              <1>       mov    [ebx+LD_StartSector], ecx
21955                              <1> loc_hd_FAT_logical_drv_init:
21956 00006649 89DD                <1>       mov    ebp, ebx
21957                              <1>       ;mov   dl, [ebx+LD_PhyDrvNo]
21958 0000664B A0[8F540100]        <1>       mov    al, [HD_LBA_yes] ; 07/01/2016
21959 00006650 884305              <1>       mov    [ebx+LD_LBAYes], al
21960 00006653 BB[A2540100]        <1>       mov    ebx, DOSBootSectorBuff ; buffer address
21961 00006658 08C0                <1>       or     al, al
21962 0000665A 740C                <1>       jz     short loc_hd_FAT_drv_init_load_bs_chs
21963                              <1> loc_hd_FAT_drv_init_load_bs_lba:
21964                              <1>       ; DL = Physical drive number
21965                              <1>       ;mov   ecx, [esi+ptStartSector] ; sector number
21966                              <1>       ;mov   ebx, DOSBootSectorBuff ; buffer address
21967                              <1>       ; LBA read/write (with private LBA function)
21968                              <1>       ;((Retro UNIX 386 v1 - DISK I/O code by Erdogan Tan))
21969                              <1>       ; dl = physical drive number (0,1, 80h, 81h, 82h, 83h)
21970 0000665C B41B                <1>       mov    ah, 1Bh ; LBA read
21971 0000665E B001                <1>       mov    al, 1 ; sector count
21972 00006660 E8A1DBFFFF          <1>       call   int13h
21973 00006665 7313                <1>       jnc    short loc_hd_drv_FAT_boot_validation
21974                              <1> loc_not_a_valid_fat_partition3:
21975 00006667 C3                  <1>       retn
21976                              <1> loc_hd_FAT_drv_init_load_bs_chs:
21977 00006668 8A7601              <1>       mov    dh, [esi+ptBeginHead]
21978 0000666B 668B4E02            <1>       mov    cx, [esi+ptBeginSector]
21979 0000666F 66B80102            <1>       mov    ax, 0201h ; Read 1 sector
21980                              <1>       ;mov   ebx, DOSBootSectorBuff
21981 00006673 E88EDBFFFF          <1>       call   int13h
21982 00006678 72ED                <1>       jc     short loc_not_a_valid_fat_partition3
21983                              <1> loc_hd_drv_FAT_boot_validation:
21984                              <1>       ;mov   esi, DOSBootSectorBuff
21985 0000667A 89DE                <1>       mov    esi, ebx
21986 0000667C 6681BEFE01000055AA  <1>       cmp    word [esi+BS_Validation], 0AA55h
21987 00006685 751A                <1>       jne    short loc_not_a_valid_fat_partition4
21988 00006687 807E15F8            <1>       cmp    byte [esi+BPB_Media], 0F8h
21989 0000668B 7514                <1>       jne    short loc_not_a_valid_fat_partition4
21990 0000668D 66837E1600          <1>       cmp    word [esi+BPB_FATSz16], 0
21991 00006692 770F                <1>       ja     short loc_hd_FAT16_BPB
21992 00006694 807E4229            <1>       cmp    byte [esi+BS_FAT32_BootSig], 29h
21993 00006698 7507                <1>       jne    short loc_not_a_valid_fat_partition4
21994                              <1> loc_hd_FAT32_BPB:
21995 0000669A B92D000000          <1>       mov    ecx, 45
21996 0000669F EB0D                <1>       jmp    short loc_hd_move_FAT_BPB
21997                              <1>       ;
21998                              <1> loc_not_a_valid_fat_partition4:
21999 000066A1 F9                  <1>       stc
22000 000066A2 C3                  <1>       retn
22001                              <1>       ;
22002                              <1> loc_hd_FAT16_BPB:
22003 000066A3 807E2629            <1>       cmp    byte [esi+BS_BootSig], 29h
22004 000066A7 75F8                <1>       jne    short loc_not_a_valid_fat_partition4
22005 000066A9 B920000000          <1>       mov    ecx, 32
22006                              <1> loc_hd_move_FAT_BPB:
22007 000066AE 89EF                <1>       mov    edi, ebp
22008                              <1>       ;mov   esi, ebx ; Boot sector
22009 000066B0 57                  <1>       push   edi
22010 000066B1 83C706              <1>       add    edi, LD_BPB
22011 000066B4 F366A5              <1>       rep    movsw
22012 000066B7 5E                  <1>       pop    esi
22013 000066B8 0FB74614            <1>       movzx  eax, word [esi+LD_BPB+BPB_RsvdSecCnt]
22014 000066BC 03466C              <1>       add    eax, [esi+LD_StartSector]
22015 000066BF 894660              <1>       mov    [esi+LD_FATBegin], eax
22016 000066C2 807E0303            <1>       cmp    byte [esi+LD_FATType], 3
22017 000066C6 7224                <1>       jb     short loc_set_FAT16_RootDirLoc
22018                              <1> loc_set_FAT32_RootDirLoc:
22019 000066C8 8B462A              <1>       mov    eax, [esi+LD_BPB+BPB_FATSz32]
22020 000066CB 0FB65E16            <1>         movzx      ebx, byte [esi+LD_BPB+BPB_NumFATs]
22021 000066CF F7E3                <1>       mul    ebx
22022 000066D1 034660              <1>       add    eax, [esi+LD_FATBegin]
22023                              <1> loc_set_FAT32_data_begin:
22024 000066D4 894668              <1>       mov    [esi+LD_DATABegin], eax
22025 000066D7 894664              <1>       mov    [esi+LD_ROOTBegin], eax
22026                              <1>       ; If Root Directory Cluster <> 2 then
```

```
22027                                  <1>       ; change the beginning sector value
22028                                  <1>       ; of the root dir by adding sector offset.
22029 000066DA 8B4632                  <1>       mov    eax, [esi+LD_BPB+BPB_RootClus]
22030 000066DD 83E802                  <1>       sub    eax, 2
22031 000066E0 7442                    <1>       jz     short short loc_set_32bit_FAT_total_sectors
22032                                  <1>       ;movzx ebx, byte [esi+LD_BPB+BPB_SecPerClust]
22033 000066E2 8A5E13                  <1>       mov    bl, byte [esi+LD_BPB+BPB_SecPerClust]
22034 000066E5 F7E3                    <1>       mul    ebx
22035 000066E7 014664                  <1>       add    [esi+LD_ROOTBegin], eax
22036 000066EA EB38                    <1>       jmp    short loc_set_32bit_FAT_total_sectors
22037                                  <1>       ;
22038                                  <1> loc_set_FAT16_RootDirLoc:
22039 000066EC 0FB64616                <1>       movzx  eax, byte [esi+LD_BPB+BPB_NumFATs]
22040 000066F0 0FB7561C                <1>       movzx  edx, word [esi+LD_BPB+BPB_FATSz16]
22041 000066F4 F7E2                    <1>       mul    edx
22042 000066F6 034660                  <1>       add    eax, [esi+LD_FATBegin]
22043 000066F9 894664                  <1>       mov    [esi+LD_ROOTBegin], eax
22044                                  <1> loc_set_FAT16_data_begin:
22045 000066FC 894668                  <1>       mov    [esi+LD_DATABegin], eax
22046 000066FF B820000000              <1>       mov    eax, 20h  ; Size of a directory entry
22047                                  <1>       ;movzx edx, word [esi+LD_BPB+BPB_RootEntCnt]
22048 00006704 668B5617                <1>         mov     dx, [esi+LD_BPB+BPB_RootEntCnt]
22049 00006708 F7E2                    <1>         mul  edx
22050                                  <1>       ;mov   ecx, 511
22051 0000670A 66B9FF01                <1>       mov    cx, 511
22052 0000670E 01C8                    <1>       add    eax, ecx
22053 00006710 41                      <1>       inc    ecx ; 512
22054 00006711 F7F1                    <1>       div    ecx
22055 00006713 014668                  <1>       add    [esi+LD_DATABegin], eax
22056 00006716 0FB74619                <1>       movzx  eax, word [esi+LD_BPB+BPB_TotalSec16]
22057 0000671A 6685C0                  <1>       test   ax, ax
22058 0000671D 7405                    <1>       jz     short loc_set_32bit_FAT_total_sectors
22059                                  <1> loc_set_16bit_FAT_total_sectors:
22060 0000671F 894670                  <1>       mov    [esi+LD_TotalSectors], eax
22061 00006722 EB06                    <1>       jmp    short loc_set_hd_FAT_cluster_count
22062                                  <1> loc_set_32bit_FAT_total_sectors:
22063 00006724 8B4626                  <1>       mov    eax, [esi+LD_BPB+BPB_TotalSec32]
22064 00006727 894670                  <1>       mov    [esi+LD_TotalSectors], eax
22065                                  <1> loc_set_hd_FAT_cluster_count:
22066 0000672A 8B5668                  <1>       mov    edx, [esi+LD_DATABegin]
22067 0000672D 2B566C                  <1>       sub    edx, [esi+LD_StartSector]
22068 00006730 29D0                    <1>       sub    eax, edx
22069 00006732 31D2                    <1>       xor    edx, edx ; 0
22070 00006734 0FB64E13                <1>         movzx   ecx, byte [esi+LD_BPB+BPB_SecPerClust]
22071 00006738 F7F1                    <1>         div  ecx
22072 0000673A 894678                  <1>       mov    [esi+LD_Clusters], eax
22073                                  <1>       ; Maximum Valid Cluster Number= EAX +1
22074                                  <1>       ; with 2 reserved clusters= EAX +2
22075                                  <1> loc_set_hd_FAT_fs_free_sectors:
22076                                  <1>       ;mov   dword [esi+LD_FreeSectors], 0
22077 0000673D E859010000              <1>       call   get_free_FAT_sectors
22078 00006742 7207                    <1>       jc     short loc_validate_hd_FAT_partition_retn
22079 00006744 894674                  <1>       mov    [esi+LD_FreeSectors], eax
22080 00006747 C6467E06                <1>       mov    byte [esi+LD_MediaChanged], 6  ; Volume Name Reset
22081                                  <1>       ;mov   cl, [Last_DOS_DiskNo]
22082                                  <1>       ;add   cl, 'A'
22083                                  <1>       ;mov   [esi+LD_FS_Name], cl
22084                                  <1>
22085                                  <1> loc_validate_hd_FAT_partition_retn:
22086 0000674B C3                      <1>       retn
22087                                  <1>
22088                                  <1> validate_hd_fs_partition:
22089                                  <1>       ; 13/02/2016
22090                                  <1>       ; 10/01/2016 (TRDOS 386 = TRDOS v2.0)
22091                                  <1>       ; 29/01/2011
22092                                  <1>       ; 23/07/2011
22093                                  <1>       ; Input
22094                                  <1>       ;   DL = Hard/Fixed Disk Drive Number
22095                                  <1>       ;   ESI = PartitionTable offset
22096                                  <1>       ;   byte [Last_DOS_DiskNo]
22097                                  <1>       ; Output
22098                                  <1>       ;  cf=0 -> Validated
22099                                  <1>       ;   ESI = Logical dos drv desc. table
22100                                  <1>       ;   EBX = Singlix FS boot sector buffer
22101                                  <1>       ;   byte [Last_DOS_DiskNo]
22102                                  <1>       ;  cf=1 -> Not a valid 'Singlix FS' partition
22103                                  <1>       ; EAX, EDX, ECX, EDI -> changed
22104                                  <1>
22105                                  <1>       ;mov   esi, PartitionTable
22106 0000674C 8A6604                  <1>       mov    ah, [esi+ptFileSystemID]
22107 0000674F 80FCA1                  <1>       cmp    ah, 0A1h ; SINGLIX FS1 (trfs1) partition
22108 00006752 7549                    <1>       jne    short loc_validate_hd_fs_partition_stc_retn
22109                                  <1> loc_set_valid_hd_fs_partition_params:
22110 00006754 FE05[97020100]          <1>       inc    byte [Last_DOS_DiskNo] ; > 1
22111 0000675A 30C0                    <1>       xor    al, al ; mov al, 0
22112                                  <1>       ;mov   [drv], dl
22113 0000675C 29DB                    <1>       sub    ebx, ebx ; 0
22114 0000675E 8A3D[97020100]          <1>       mov    bh, [Last_DOS_DiskNo]
22115 00006764 81C300010900            <1>       add    ebx, Logical_DOSDisks
22116 0000676A C6430102                <1>       mov    byte [ebx+LD_DiskType], 2
22117 0000676E 885302                  <1>       mov    [ebx+LD_PhyDrvNo], dl
22118                                  <1>       ;mov   [ebx+LD_FATType], al ; 0
22119                                  <1>       ;mov   [ebx+LD_FSType], ah
22120 00006771 66894303                <1>       mov    [ebx+LD_FATType], ax
22121                                  <1>       ;mov   eax, [esi+ptStartSector]
22122                                  <1>       ;mov   [ebx+LD_StartSector], eax
22123                                  <1> loc_hd_fs_logical_drv_init:
22124 00006775 89DD                    <1>       mov    ebp, ebx ; 10/01/2016
22125                                  <1>       ;mov   dl, [ebx+LD_PhyDrvNo]
22126 00006777 A0[8F540100]            <1>       mov    al, [HD_LBA_yes] ; 10/01/2016
22127 0000677C 884305                  <1>       mov    [ebx+LD_LBAYes], al
22128 0000677F 89DE                    <1>       mov    esi, ebx
```

```
22129 00006781 BB[A2540100]        <1>       mov    ebx, DOSBootSectorBuff ; buffer address
22130 00006786 08C0                <1>       or     al, al
22131 00006788 7515                <1>       jnz    short loc_hd_fs_drv_init_load_bs_lba
22132                              <1> loc_hd_fs_drv_init_load_bs_chs:
22133 0000678A 8A7601              <1>       mov    dh, [esi+ptBeginHead]
22134 0000678D 668B4E02            <1>       mov    cx, [esi+ptBeginSector]
22135 00006791 66B80102            <1>       mov    ax, 0201h ; Read 1 sector
22136                              <1>       ;mov   ebx, DOSBootSectorBuff
22137 00006795 E86CDAFFFF          <1>       call   int13h
22138 0000679A 7311                <1>       jnc    short loc_hd_drv_fs_boot_validation
22139                              <1> loc_validate_hd_fs_partition_err_retn:
22140 0000679C C3                  <1>       retn
22141                              <1> loc_validate_hd_fs_partition_stc_retn:
22142 0000679D F9                  <1>       stc
22143 0000679E C3                  <1>       retn
22144                              <1> loc_hd_fs_drv_init_load_bs_lba:
22145                              <1>       ; DL = Physical drive number
22146                              <1>       ;mov   esi, ebx
22147 0000679F 8B4E08              <1>       mov    ecx, [esi+ptStartSector] ; sector number
22148                              <1>       ;mov   ebx, DOSBootSectorBuff ; buffer address
22149                              <1>       ; LBA read/write (with private LBA function)
22150                              <1>       ;((Retro UNIX 386 v1 - DISK I/O code by Erdogan Tan))
22151                              <1>       ; dl = physical drive number (0,1, 80h, 81h, 82h, 83h)
22152 000067A2 B41B                <1>       mov    ah, 1Bh ; LBA read
22153 000067A4 B001                <1>       mov    al, 1 ; sector count
22154 000067A6 E85BDAFFFF          <1>       call   int13h
22155 000067AB 72EF                <1>       jc     short loc_validate_hd_fs_partition_err_retn
22156                              <1> loc_hd_drv_fs_boot_validation:
22157                              <1>       ;mov   esi, DOSBootSectorBuff
22158 000067AD 89DE                <1>       mov    esi, ebx ; Boot sector buffer
22159 000067AF 6681BEFE01000055AA  <1>       cmp    word [esi+BS_Validation], 0AA55h
22160 000067B8 75E3                <1>       jne    short loc_validate_hd_fs_partition_stc_retn
22161                              <1>         ;
22162                              <1>       ;Singlix FS Extensions to TR-DOS (7/6/2009)
22163 000067BA 66817E035346        <1>       cmp    word [esi+bs_FS_Identifier], 'SF'
22164 000067C0 75DB                <1>       jne    short loc_validate_hd_fs_partition_stc_retn
22165                              <1>         ;'A1h' check is not necessary
22166                              <1>       ;  if 'FS' check is passed as OK/Yes.
22167 000067C2 807E09A1            <1>       cmp    byte [esi+bs_FS_PartitionID], 0A1h
22168 000067C6 75D5                <1>       jne    short loc_validate_hd_fs_partition_stc_retn
22169                              <1>       ;
22170 000067C8 89EF                <1>       mov    edi, ebp ; 10/01/2016
22171                              <1>       ;
22172 000067CA 8A462D              <1>       mov    al, byte [esi+bs_FS_LBA_Ready]
22173 000067CD 884705              <1>       mov    [edi+LD_FS_LBAYes], al
22174                              <1>       ;
22175                              <1>       ; 03/01/2010 CHS -> DOS FAT/BPB compatibility fix
22176 000067D0 8A4608              <1>       mov    al, [esi+bs_FS_MediaAttrib]
22177 000067D3 884706              <1>       mov    byte [edi+LD_FS_MediaAttrib], al
22178                              <1>       ;
22179 000067D6 8A460A              <1>       mov    al, [esi+bs_FS_VersionMaj]
22180 000067D9 884707              <1>       mov    [edi+LD_FS_VersionMajor], al
22181                              <1>       ;
22182 000067DC 668B4606            <1>       mov    ax, [esi+bs_FS_BytesPerSec]
22183 000067E0 66894711            <1>       mov    [edi+LD_FS_BytesPerSec], ax
22184 000067E4 8A462E              <1>       mov    al, [esi+bs_FS_SecPerTrack]
22185 000067E7 6698                <1>       cbw
22186 000067E9 6689471E            <1>       mov    [edi+LD_FS_SecPerTrack], ax
22187 000067ED 8A462F              <1>       mov    al, [esi+bs_FS_Heads]
22188                              <1>       ;cbw
22189 000067F0 66894720            <1>       mov    [edi+LD_FS_NumHeads], ax
22190                              <1>       ;
22191 000067F4 8B4628              <1>       mov    eax, [esi+bs_FS_UnDelDirD]
22192 000067F7 894722              <1>       mov    [edi+LD_FS_UnDelDirD], eax
22193 000067FA 8B5618              <1>       mov    edx, [esi+bs_FS_MATLocation]
22194 000067FD 89570C              <1>       mov    [edi+LD_FS_MATLocation], edx
22195 00006800 8B461C              <1>       mov    eax, [esi+bs_FS_RootDirD]
22196 00006803 894708              <1>       mov    [edi+LD_FS_RootDirD], eax
22197 00006806 8B460C              <1>       mov    eax, [esi+bs_FS_BeginSector]
22198 00006809 89476C              <1>       mov    [edi+LD_FS_BeginSector], eax
22199 0000680C 8B4710              <1>       mov    eax, [edi+bs_FS_VolumeSize]
22200 0000680F 894770              <1>       mov    [edi+LD_FS_VolumeSize], eax
22201                              <1>       ;
22202 00006812 89D0                <1>       mov    eax, edx ; [edi+LD_FS_MATLocation]
22203 00006814 03476C              <1>       add    eax, [edi+LD_FS_BeginSector]
22204 00006817 89FE                <1>       mov    esi, edi
22205                              <1> mread_hd_fs_MAT_sector:
22206                              <1>        ;mov  ebx, DOSBootSectorBuff
22207 00006819 B901000000          <1>       mov    ecx, 1
22208 0000681E E87B890000          <1>       call   disk_read
22209 00006823 7248                <1>       jc     short loc_validate_hd_fs_partition_retn
22210                              <1>       ; EDI will not be changed
22211 00006825 89DE                <1>       mov    esi, ebx
22212                              <1> use_hdfs_mat_sector_params:
22213 00006827 8B460C              <1>       mov    eax, [esi+FS_MAT_DATLocation]
22214 0000682A 894714              <1>       mov    [edi+LD_FS_DATLocation], eax
22215 0000682D 8B4610              <1>       mov    eax, [esi+FS_MAT_DATScount]
22216 00006830 894718              <1>       mov    [edi+LD_FS_DATSectors], eax
22217 00006833 8B4614              <1>       mov    eax, [esi+FS_MAT_FreeSectors]
22218 00006836 894774              <1>         mov     [edi+LD_FS_FreeSectors], eax
22219 00006839 8B4618              <1>       mov    eax, [esi+FS_MAT_FirstFreeSector]
22220 0000683C 894778              <1>       mov    [edi+LD_FS_FirstFreeSector], eax
22221 0000683F 8B4708              <1>       mov    eax, [edi+LD_FS_RootDirD]
22222 00006842 03476C              <1>       add    eax, [edi+LD_FS_BeginSector]
22223 00006845 89FE                <1>       mov    esi, edi
22224                              <1> read_hd_fs_RDT_sector:
22225 00006847 BB[A2540100]        <1>       mov    ebx, DOSBootSectorBuff
22226                              <1>       ;mov   ecx, 1
22227 0000684C B101                <1>       mov    cl, 1
22228 0000684E E84B890000          <1>       call   disk_read
22229 00006853 7218                <1>       jc     short loc_validate_hd_fs_partition_retn
22230                              <1>       ; EDI will not be changed
```

```
22231 00006855 89DE              <1>       mov    esi, ebx
22232                            <1> use_hdfs_RDT_sector_params:
22233 00006857 8B461C            <1>       mov    eax, [esi+FS_RDT_VolumeSerialNo]
22234 0000685A 894728            <1>       mov    [edi+LD_FS_VolumeSerial], eax
22235 0000685D 57                <1>       push   edi
22236                            <1>       ;mov   ecx, 16
22237 0000685E B110              <1>       mov    cl, 16
22238 00006860 83C640            <1>       add    esi, FS_RDT_VolumeName
22239 00006863 83C72C            <1>       add    edi, LD_FS_VolumeName
22240 00006866 F3A5              <1>       rep    movsd ; 64 bytes
22241 00006868 5E                <1>       pop    esi
22242                            <1>              ; Volume Name Reset
22243 00006869 C6467E06          <1>         mov     byte [esi+LD_FS_MediaChanged], 6
22244                            <1>       ;
22245                            <1>         ;mov cl, [Last_DOS_DiskNo]
22246                            <1>       ;add   cl, 'A'
22247                            <1>       ;mov   [esi+LD_FS_Name], cl
22248                            <1>
22249                            <1> loc_validate_hd_fs_partition_retn:
22250 0000686D C3                <1>       retn
22251                            <1>
22252                            <1> load_masterboot:
22253                            <1>       ; 10/01/2016 (TRDOS 386 = TRDOS v2.0)
22254                            <1>       ; 2005 - 2011
22255                            <1>       ; input -> DL = drive number
22256 0000686E B40D              <1>       mov    ah, 0Dh ; Alternate disk reset
22257 00006870 E891D9FFFF        <1>       call   int13h
22258 00006875 7301              <1>       jnc    short pass_reset_error
22259                            <1> harddisk_error:
22260 00006877 C3                <1>       retn
22261                            <1> pass_reset_error:
22262 00006878 BB[8E500100]      <1>       mov    ebx, MasterBootBuff
22263 0000687D 66B80102          <1>       mov    ax, 0201h
22264 00006881 66B90100          <1>       mov    cx, 1
22265 00006885 30F6              <1>       xor    dh, dh
22266 00006887 E87AD9FFFF        <1>       call   int13h
22267 0000688C 72E9              <1>       jc     short harddisk_error
22268                            <1>       ;
22269 0000688E 66813D[8C520100]55- <1>     cmp    word [MBIDCode], 0AA55h
22270 00006896 AA                <1>
22271 00006897 7401              <1>       je     short load_masterboot_ok
22272 00006899 F9                <1>       stc
22273                            <1> load_masterboot_ok:
22274 0000689A C3                <1>       retn
22275                            <1>
22276                            <1> get_free_FAT_sectors:
22277                            <1>       ; 29/02/2016
22278                            <1>       ; 13/02/2016
22279                            <1>       ; 04/02/2016
22280                            <1>       ; 07/01/2016 (TRDOS 386 = TRDOS v2.0)
22281                            <1>       ; 11/07/2010
22282                            <1>       ; 21/06/2009
22283                            <1>       ; INPUT: ESI = Logical DOS Drive Description Table address
22284                            <1>       ; OUTPUT: STC => Error
22285                            <1>         ;    cf = 0 and EAX = Free FAT sectors
22286                            <1>       ; Also, related parameters and FAT buffer will be reset and updated
22287                            <1>
22288 0000689B 31C0              <1>       xor    eax, eax
22289                            <1>       ;mov   [esi+LD_FreeSectors], eax ; Reset
22290                            <1>
22291 0000689D 807E0302          <1>         cmp     byte [esi+LD_FATType], 2
22292 000068A1 7650              <1>       jna    short loc_gfc_get_fat_free_clusters
22293                            <1>
22294                            <1>       ; 29/02/2016
22295 000068A3 48                <1>       dec    eax ; 0FFFFFFFFh
22296 000068A4 89463A            <1>       mov    [esi+LD_BPB+BPB_Reserved], eax ; Free cluster count (reset)
22297 000068A7 89463E            <1>       mov    [esi+LD_BPB+BPB_Reserved+4], eax ; First Free Cluster (reset)
22298 000068AA 40                <1>       inc    eax ; 0
22299                            <1>       ;
22300 000068AB 668B4636          <1>       mov    ax, [esi+LD_BPB+BPB_FSInfo]
22301 000068AF 03466C            <1>       add    eax, [esi+LD_StartSector]
22302                            <1>
22303 000068B2 BB[A2540100]      <1>       mov    ebx, DOSBootSectorBuff
22304 000068B7 B901000000        <1>       mov    ecx, 1
22305 000068BC E8DD880000        <1>       call   disk_read
22306 000068C1 7301              <1>       jnc    short loc_gfc_check_fsinfo_signs
22307                            <1> retn_gfc_get_fsinfo_sec:
22308 000068C3 C3                <1>       retn
22309                            <1>
22310                            <1> loc_gfc_check_fsinfo_signs:
22311 000068C4 BB[A2540100]      <1>       mov    ebx, DOSBootSectorBuff ; 13/02/2016
22312 000068C9 813B52526141      <1>         cmp     dword [ebx], 41615252h
22313 000068CF 7520              <1>       jne    short retn_gfc_get_fsinfo_stc
22314                            <1>       ;add   ebx, 484
22315                            <1>       ;cmp   dword [ebx], 61417272h
22316 000068D1 81BBE4010000727241- <1>     cmp    dword [ebx+484], 61417272h
22317 000068DA 61                <1>
22318 000068DB 7514              <1>       jne    short retn_gfc_get_fsinfo_stc
22319                            <1>       ;add   ebx, 4
22320                            <1>       ;mov   eax, [ebx]
22321 000068DD 8B83E8010000      <1>       mov    eax, [ebx+488]
22322                            <1>       ; 29/02/2016
22323 000068E3 89463A            <1>       mov    [esi+LD_BPB+BPB_Reserved], eax ; Free cluster count
22324 000068E6 8B93EC010000      <1>       mov    edx,  [ebx+492]
22325 000068EC 89463E            <1>       mov    [esi+LD_BPB+BPB_Reserved+4], eax ; First Free Cluster
22326                            <1>       ;
22327 000068EF EB12              <1>       jmp    short retn_from_get_free_fat32_clusters
22328                            <1>
22329                            <1> retn_gfc_get_fsinfo_stc:
22330 000068F1 F9                <1>       stc
22331 000068F2 C3                <1>       retn
22332                            <1>
```

```
22333                                  <1> loc_gfc_get_fat_free_clusters:
22334                                  <1>       ;mov   eax, 2
22335 000068F3 B002                     <1>       mov    al, 2
22336                                  <1>       ;mov   [FAT_CurrentCluster], eax
22337                                  <1> loc_gfc_loop_get_next_cluster:
22338 000068F5 E823500000               <1>       call   get_next_cluster
22339 000068FA 730E                     <1>       jnc    short loc_gfc_free_fat_clusters_cont
22340 000068FC 21C0                     <1>       and    eax, eax
22341 000068FE 7411                     <1>       jz     short loc_gfc_pass_inc_free_cluster_count
22342                                  <1>
22343                                  <1> retn_from_get_free_fat_clusters:
22344 00006900 8B4674                   <1>       mov    eax, [esi+LD_FreeSectors] ; Free clusters !
22345                                  <1> retn_from_get_free_fat32_clusters:
22346 00006903 0FB65E13                 <1>         movzx      ebx, byte [esi+LD_BPB+BPB_SecPerClust]
22347 00006907 F7E3                     <1>             mul    ebx
22348                                  <1>       ;mov   [esi+LD_FreeSectors], eax ; Free sectors
22349                                  <1> retn_get_free_sectors_calc:
22350 00006909 C3                       <1>       retn
22351                                  <1>
22352                                  <1> loc_gfc_free_fat_clusters_cont:
22353 0000690A 09C0                     <1>       or     eax, eax
22354 0000690C 7503                     <1>       jnz    short loc_gfc_pass_inc_free_cluster_count
22355 0000690E FF4674                   <1>       inc    dword [esi+LD_FreeSectors] ; Free clusters !
22356                                  <1>
22357                                  <1> loc_gfc_pass_inc_free_cluster_count:
22358                                  <1>       ;mov   eax, [FAT_CurrentCluster]
22359 00006911 89C8                     <1>       mov    eax, ecx ; [FAT_CurrentCluster]
22360 00006913 3B4678                   <1>       cmp    eax, [esi+LD_Clusters]
22361 00006916 77E8                     <1>       ja     short retn_from_get_free_fat_clusters
22362 00006918 40                       <1>       inc    eax
22363                                  <1>       ;mov   [FAT_CurrentCluster], eax
22364 00006919 EBDA                     <1>       jmp    short loc_gfc_loop_get_next_cluster
22365                                  <1>
22366                                  <1> floppy_drv_init:
22367                                  <1>       ; 06/07/2016
22368                                  <1>       ; 10/01/2016 (TRDOS 386 = TRDOS v2.0)
22369                                  <1>       ; 24/07/2011
22370                                  <1>       ; 04/07/2009
22371                                  <1>       ; INPUT ->
22372                                  <1>       ;      DL = Drive number (0,1)
22373                                  <1>       ; OUTPUT ->
22374                                  <1>       ;      BL = drive name
22375                                  <1>       ;      BH = drive number
22376                                  <1>       ;      ESI = Logical DOS drv description table
22377                                  <1>       ;      EAX = Volume serial number
22378                                  <1>
22379 0000691B BE[F65C0000]             <1>       mov    esi, fd0_type ; 10/01/2016
22380 00006920 BF00010900               <1>       mov    edi, Logical_DOSDisks
22381 00006925 08D2                     <1>       or     dl, dl
22382 00006927 7407                     <1>       jz     short loc_drv_init_fd0_fd1
22383 00006929 81C700010000             <1>       add    edi, 100h
22384 0000692F 46                       <1>       inc    esi ; fd1_type ; 10/01/2016
22385                                  <1> loc_drv_init_fd0_fd1:
22386 00006930 C6477E00                 <1>       mov    byte [edi+LD_MediaChanged], 0
22387 00006934 803E01                   <1>       cmp    byte [esi], 1 ; type (>0 if it is existing)
22388                                  <1>              ; 4 = 1.44 MB, 80 track, 3 1/2"
22389 00006937 7221                     <1>       jb     short read_fd_boot_sector_retn
22390 00006939 885702                   <1>       mov    [edi+LD_PhyDrvNo], dl
22391                                  <1> read_fd_boot_sector:
22392 0000693C 30F6                     <1>       xor    dh, dh
22393 0000693E B904000000               <1>       mov    ecx, 4 ; Retry Count
22394                                  <1> read_fd_boot_sector_again:
22395 00006943 51                       <1>       push   ecx
22396                                  <1>       ;mov   cx, 1
22397 00006944 B101                     <1>       mov    cl, 1
22398 00006946 66B80102                 <1>       mov    ax, 0201h ; Read 1 sector
22399 0000694A BB[A2540100]             <1>       mov    ebx, DOSBootSectorBuff
22400 0000694F E8B2D8FFFF               <1>       call   int13h
22401 00006954 59                       <1>       pop    ecx
22402 00006955 7304                     <1>       jnc    short use_fd_boot_sector_params
22403 00006957 E2EA                     <1>       loop   read_fd_boot_sector_again
22404                                  <1>
22405                                  <1> read_fd_boot_sector_stc_retn:
22406 00006959 F9                       <1>       stc
22407                                  <1> read_fd_boot_sector_retn:
22408 0000695A C3                       <1>       retn
22409                                  <1>
22410                                  <1> use_fd_boot_sector_params:
22411                                  <1>       ;mov   esi, DOSBootSectorBuff
22412 0000695B 89DE                     <1>       mov    esi, ebx
22413 0000695D 6681BEFE01000055AA       <1>       cmp    word [esi+BS_Validation], 0AA55h
22414 00006966 75F1                     <1>       jne    short read_fd_boot_sector_stc_retn
22415 00006968 66817E035346             <1>         cmp     word [esi+bs_FS_Identifier], 'SF'
22416 0000696E 0F85A2000000             <1>         jne     use_fd_fatfs_boot_sector_params
22417                                  <1>       ;
22418 00006974 8A462D                   <1>       mov    al, [esi+bs_FS_LBA_Ready]
22419 00006977 884705                   <1>       mov    [edi+LD_FS_LBAYes], al
22420                                  <1>       ;
22421                                  <1>       ; 03/01/2010 CHS -> DOS FAT/BPB compatibility fix
22422 0000697A 8A4608                   <1>       mov    al, [esi+bs_FS_MediaAttrib]
22423 0000697D 884706                   <1>       mov    [edi+LD_FS_MediaAttrib], al
22424                                  <1>       ;
22425 00006980 8A460A                   <1>         mov  al, [esi+bs_FS_VersionMaj]
22426 00006983 884707                   <1>       mov    byte [edi+LD_FS_VersionMajor], al
22427 00006986 668B4606                 <1>       mov    ax, [esi+bs_FS_BytesPerSec]
22428 0000698A 66894711                 <1>       mov    [edi+LD_FS_BytesPerSec], ax
22429 0000698E 8A462E                   <1>       mov    al, [esi+bs_FS_SecPerTrack]
22430 00006991 6698                     <1>       cbw
22431 00006993 6689471E                 <1>       mov    [edi+LD_FS_SecPerTrack], ax
22432 00006997 8A462F                   <1>       mov    al, [esi+bs_FS_Heads]
22433                                  <1>       ;cbw
22434 0000699A 66894720                 <1>       mov    [edi+LD_FS_NumHeads], ax
```

```
22435                                  <1>       ;
22436 0000699E 8B4628                  <1>       mov    eax, [esi+bs_FS_UnDelDirD]
22437 000069A1 894722                  <1>       mov    [edi+LD_FS_UnDelDirD], eax
22438 000069A4 8B4618                  <1>       mov    eax, [esi+bs_FS_MATLocation]
22439 000069A7 89470C                  <1>       mov    [edi+LD_FS_MATLocation], eax
22440 000069AA 8B461C                  <1>       mov    eax, [esi+bs_FS_RootDirD]
22441 000069AD 894708                  <1>       mov    [edi+LD_FS_RootDirD], eax
22442 000069B0 8B460C                  <1>       mov    eax, [esi+bs_FS_BeginSector]
22443 000069B3 89476C                  <1>       mov    [edi+LD_FS_BeginSector], eax
22444 000069B6 8B4610                  <1>       mov    eax, [esi+bs_FS_VolumeSize]
22445 000069B9 894770                  <1>       mov    [edi+LD_FS_VolumeSize], eax
22446                                  <1>       ;
22447 000069BC 89FE                    <1>       mov    esi, edi
22448 000069BE 8B460C                  <1>       mov    eax, [esi+LD_FS_MATLocation]
22449                                  <1>       ;add   eax, [edi+LD_FS_BeginSector]
22450                                  <1> read_fd_MAT_sector_again:
22451                                  <1>       ;mov   ebx, DOSBootSectorBuff
22452                                  <1>       ;mov   ecx, 1
22453 000069C1 B101                    <1>       mov    cl, 1
22454 000069C3 E8DC870000              <1>       call   chs_read
22455 000069C8 89DE                    <1>       mov    esi, ebx
22456 000069CA 7301                    <1>       jnc    short use_fdfs_mat_sector_params
22457                                  <1>       ;jmp   short read_fd_boot_sector_retn
22458 000069CC C3                      <1>       retn
22459                                  <1> use_fdfs_mat_sector_params:
22460 000069CD 8B460C                  <1>       mov    eax, [esi+FS_MAT_DATLocation]
22461 000069D0 894714                  <1>       mov    [edi+LD_FS_DATLocation], eax
22462 000069D3 8B4610                  <1>       mov    eax, [esi+FS_MAT_DATScount]
22463 000069D6 894718                  <1>       mov    [edi+LD_FS_DATSectors], eax
22464 000069D9 8B4714                  <1>       mov    eax, [edi+FS_MAT_FreeSectors]
22465 000069DC 894774                  <1>       mov    [edi+LD_FS_FreeSectors], eax
22466 000069DF 8B4618                  <1>       mov    eax, [esi+FS_MAT_FirstFreeSector]
22467 000069E2 894778                  <1>       mov    [edi+LD_FS_FirstFreeSector], eax
22468                                  <1>       ;
22469 000069E5 89FE                    <1>       mov    esi, edi
22470 000069E7 8B4608                  <1>       mov    eax, [esi+LD_FS_RootDirD]
22471                                  <1> read_fd_RDT_sector_again:
22472                                  <1>       ;mov   ebx, DOSBootSectorBuff
22473                                  <1>       ;mov   cx, 1
22474 000069EA B101                    <1>       mov    cl, 1
22475 000069EC E8B3870000              <1>       call   chs_read
22476 000069F1 89DE                    <1>       mov    esi, ebx
22477 000069F3 7220                    <1>       jc     short read_fd_RDT_sector_retn
22478                                  <1> use_fdfs_RDT_sector_params:
22479 000069F5 8B461C                  <1>       mov    eax, [esi+FS_RDT_VolumeSerialNo]
22480 000069F8 894728                  <1>       mov    [edi+LD_FS_VolumeSerial], eax
22481 000069FB 57                      <1>       push   edi
22482                                  <1>       ;mov   ecx, 16
22483 000069FC B110                    <1>       mov    cl, 16
22484 000069FE 83C640                  <1>       add    esi, FS_RDT_VolumeName
22485 00006A01 83C72C                  <1>       add    edi, LD_FS_VolumeName
22486 00006A04 F3A5                    <1>       rep    movsd ; 64 bytes
22487 00006A06 5E                      <1>       pop    esi
22488 00006A07 C6460300                <1>       mov    byte [esi+LD_FATType], 0
22489 00006A0B C64604A1                <1>       mov    byte [esi+LD_FSType], 0A1h
22490 00006A0F E9A5000000              <1>         jmp     loc_cont_use_fd_boot_sector_params
22491                                  <1>
22492                                  <1> read_fd_RDT_sector_stc_retn:
22493 00006A14 F9                      <1>       stc
22494                                  <1> read_fd_RDT_sector_retn:
22495 00006A15 C3                      <1>       retn
22496                                  <1>
22497                                  <1> use_fd_fatfs_boot_sector_params:
22498 00006A16 807E2629                <1>       cmp    byte [esi+BS_BootSig], 29h
22499 00006A1A 75F8                    <1>       jne    short read_fd_RDT_sector_stc_retn
22500 00006A1C 807E15F0                <1>       cmp    byte [esi+BPB_Media], 0F0h
22501 00006A20 72F3                    <1>       jb     short read_fd_RDT_sector_retn
22502 00006A22 57                      <1>       push   edi
22503 00006A23 83C706                  <1>       add    edi, LD_BPB
22504                                  <1>       ;mov   ecx, 16
22505 00006A26 B110                    <1>       mov    cl, 16
22506 00006A28 F3A5                    <1>       rep    movsd ; 64 bytes
22507 00006A2A 5E                      <1>       pop    esi
22508 00006A2B 31C0                    <1>       xor    eax, eax
22509 00006A2D 89466C                  <1>       mov    [esi+LD_StartSector], eax ; 0
22510 00006A30 668B461C                <1>       mov    ax, [esi+LD_BPB+BPB_FATSz16]
22511 00006A34 8A4E16                  <1>       mov    cl, [esi+LD_BPB+BPB_NumFATs]
22512 00006A37 F7E1                    <1>       mul    ecx
22513                                  <1>       ; edx = 0 !
22514 00006A39 668B5614                <1>       mov    dx, [esi+LD_BPB+BPB_RsvdSecCnt]
22515 00006A3D 66895660                <1>       mov    [esi+LD_FATBegin], dx
22516                                  <1>       ;add   eax, edx
22517 00006A41 6601D0                  <1>       add    ax, dx
22518 00006A44 894664                  <1>       mov    [esi+LD_ROOTBegin], eax
22519 00006A47 894668                  <1>       mov    [esi+LD_DATABegin], eax
22520 00006A4A 668B5617                <1>       mov    dx, [esi+LD_BPB+BPB_RootEntCnt]
22521                                  <1>       ;;shl  edx, 5 ; * 32 (Size of a directory entry)
22522                                  <1>       ;shl   dx, 5
22523                                  <1>       ;;add  edx, 511
22524                                  <1>       ;add   dx, 511
22525                                  <1>       ;;shr  edx, 9 ; edx = ((edx*32)+511) / 512
22526                                  <1>       ;shr   dx, 9
22527 00006A4E 6683C20F               <1>       add    dx, 15 ; 06/07/2016 (+(512/32)-1)
22528 00006A52 66C1EA04                <1>       shr    dx, 4 ; / 16 (==16 entries per sector)
22529 00006A56 015668                  <1>       add    [esi+LD_DATABegin], edx ; + rd sectors
22530                                  <1>       ;movzx eax, word [esi+LD_BPB+BPB_TotalSec16]
22531 00006A59 668B4619                <1>       mov    ax, [esi+LD_BPB+BPB_TotalSec16]
22532 00006A5D 894670                  <1>       mov    [esi+LD_TotalSectors], eax
22533 00006A60 2B4668                  <1>       sub    eax, [esi+LD_DATABegin]
22534                                  <1>       ;movzx ecx, byte [esi+LD_BPB+BPB_SecPerClust]
22535 00006A63 8A4E13                  <1>       mov    cl, [esi+LD_BPB+BPB_SecPerClust]
22536 00006A66 80F901                  <1>       cmp    cl, 1
```

```
22537 00006A69 7605                <1>       jna    short save_fd_fatfs_cluster_count
22538                              <1>       ;sub   edx, edx
22539 00006A6B 6629D2              <1>       sub    dx, dx ; 0
22540                              <1>       ;sub   dl, dl ; 06/07/2016
22541 00006A6E F7F1                <1>       div    ecx
22542                              <1> save_fd_fatfs_cluster_count:
22543 00006A70 894678              <1>       mov    [esi+LD_Clusters], eax
22544                              <1>
22545                              <1>       ; Maximum Valid Cluster Number = EAX +1
22546                              <1>       ; with 2 reserved clusters= EAX +2
22547                              <1>
22548                              <1> reset_FAT_buffer_decriptors:
22549 00006A73 29C0                <1>       sub    eax, eax ; 0
22550 00006A75 A2[A6560100]        <1>       mov    [FAT_BuffValidData], al ; 0
22551 00006A7A A2[A7560100]        <1>       mov    [FAT_BuffDrvName], al ; 0
22552 00006A7F A3[AA560100]        <1>       mov    [FAT_BuffSector], eax ; 0
22553                              <1>
22554                              <1> read_fd_FAT_sectors:
22555 00006A84 BB001C0900          <1>       mov    ebx, FAT_Buffer
22556 00006A89 668B4614            <1>       mov    ax, [esi+LD_BPB+BPB_RsvdSecCnt]
22557                              <1>       ;mov   ecx, 3
22558 00006A8D B103                <1>       mov    cl, 3 ; 3 sectors
22559 00006A8F E810870000          <1>       call   chs_read
22560 00006A94 7240                <1>       jc     short read_fd_FAT_sectors_retn
22561                              <1> use_fd_FAT_sectors:
22562 00006A96 8A4602              <1>       mov    al, [esi+LD_PhyDrvNo]
22563 00006A99 0441                <1>       add    al, 'A'
22564 00006A9B A2[A7560100]        <1>       mov    [FAT_BuffDrvName], al
22565 00006AA0 C605[A6560100]01    <1>       mov    byte [FAT_BuffValidData], 1
22566 00006AA7 E82B000000          <1>       call   fd_init_calculate_free_clusters
22567 00006AAC 7228                <1>       jc     short read_fd_FAT_sectors_retn
22568                              <1>
22569                              <1> loc_use_fd_boot_sector_params_FAT:
22570 00006AAE C6460301            <1>       mov    byte [esi+LD_FATType], 1 ; FAT 12
22571 00006AB2 C6460401            <1>       mov    byte [esi+LD_FSType], 1
22572 00006AB6 8B462D              <1>         mov     eax, [esi+LD_BPB+VolumeID]
22573                              <1> loc_cont_use_fd_boot_sector_params:
22574 00006AB9 8A7E02              <1>       mov    bh, [esi+LD_PhyDrvNo]
22575 00006ABC 887E7D              <1>       mov    [esi+LD_DParamEntry], bh
22576 00006ABF 88FB                <1>       mov    bl, bh
22577 00006AC1 80C341              <1>       add    bl, 'A'
22578 00006AC4 881E                <1>       mov    byte [esi+LD_Name], bl
22579 00006AC6 C6460101            <1>       mov    byte [esi+LD_DiskType], 1
22580 00006ACA C6460500            <1>       mov    byte [esi+LD_LBAYes], 0
22581 00006ACE C6467C00            <1>       mov    byte [esi+LD_PartitionEntry], 0
22582 00006AD2 C6467E06            <1>       mov    byte [esi+LD_MediaChanged], 6 ; Volume Name Reset
22583                              <1>
22584                              <1> read_fd_FAT_sectors_retn:
22585 00006AD6 C3                  <1>       retn
22586                              <1>
22587                              <1> fd_init_calculate_free_clusters:
22588                              <1>       ; 10/01/2016 (TRDOS 386 = TRDOS v2.0)
22589                              <1>       ; 04/07/2009
22590                              <1>       ; INPUT ->
22591                              <1>       ;     ESI = Logical DOS drive description table address
22592                              <1>       ; OUTPUT ->
22593                              <1>       ;    [ESI+LD_FreeSectors] will be set
22594                              <1>
22595 00006AD7 29C0                <1>       sub    eax, eax
22596 00006AD9 894674              <1>       mov    [esi+LD_FreeSectors], eax ; 0
22597 00006ADC B002                <1>       mov    al, 2 ; eax = 2
22598                              <1>
22599                              <1> fd_init_loop_get_next_cluster:
22600 00006ADE E830000000          <1>       call   fd_init_get_next_cluster
22601 00006AE3 722D                <1>       jc     short fd_init_calculate_free_clusters_retn
22602                              <1>
22603                              <1> fd_init_free_fat_clusters:
22604                              <1>       ;cmp   eax, 0
22605                              <1>       ;ja    short fd_init_pass_inc_free_cluster_count
22606                              <1>       ;and   eax, eax
22607                              <1>       ;jnz   short fd_init_pass_inc_free_cluster_count
22608 00006AE5 6621C0              <1>       and    ax, ax
22609 00006AE8 7504                <1>       jnz    short fd_init_pass_inc_free_cluster_count
22610                              <1>       ;inc   dword [esi+LD_FreeSectors]
22611 00006AEA 66FF4674            <1>         inc  word [esi+LD_FreeSectors]
22612                              <1>
22613                              <1> fd_init_pass_inc_free_cluster_count:
22614                              <1>       ;mov   eax, [FAT_CurrentCluster]
22615 00006AEE 66A1[A2560100]      <1>       mov    ax, [FAT_CurrentCluster]
22616                              <1>       ;cmp   eax, [esi+LD_Clusters]
22617 00006AF4 663B4678            <1>       cmp    ax, [esi+LD_Clusters]
22618 00006AF8 7704                <1>       ja     short short retn_from_fd_init_calculate_free_clusters
22619                              <1>       ;inc   eax
22620 00006AFA 6640                <1>       inc    ax
22621 00006AFC EBE0                <1>       jmp    short fd_init_loop_get_next_cluster
22622                              <1>
22623                              <1> retn_from_fd_init_calculate_free_clusters:
22624 00006AFE 8A4613              <1>       mov    al, [esi+LD_BPB+BPB_SecPerClust]
22625 00006B01 3C01                <1>       cmp    al, 1
22626 00006B03 760D                <1>       jna    short fd_init_calculate_free_clusters_retn
22627                              <1>       ;movzx eax, al
22628 00006B05 6698                <1>       cbw
22629                              <1>       ;mov   ecx, [esi+LD_FreeSectors]
22630 00006B07 668B4E74            <1>       mov    cx, [esi+LD_FreeSectors] ; Count of free clusters
22631                              <1>       ;mul   ecx
22632 00006B0B 66F7E1              <1>       mul    cx
22633                              <1>       ;mov   [esi+LD_FreeSectors], eax
22634 00006B0E 66894674            <1>       mov    [esi+LD_FreeSectors], ax
22635                              <1> fd_init_calculate_free_clusters_retn:
22636 00006B12 C3                  <1>       retn
22637                              <1>
22638                              <1> fd_init_get_next_cluster:
```

```
22639                                  <1>       ; 04/02/2016
22640                                  <1>       ; 02/02/2016
22641                                  <1>       ; 10/01/2016 (TRDOS 386 = TRDOS v2.0)
22642                                  <1>       ; 04/07/2009
22643                                  <1>       ; INPUT ->
22644                                  <1>       ;    EAX = Current cluster
22645                                  <1>       ;    ESI = Logical DOS drive description table address
22646                                  <1>       ;    EDX = 0
22647                                  <1>       ; OUTPUT ->
22648                                  <1>       ;    EAX = Next cluster
22649                                  <1>
22650 00006B13 A3[A2560100]           <1>       mov    [FAT_CurrentCluster], eax
22651                                  <1> fd_init_get_next_cluster_readnext:
22652 00006B18 29D2                   <1>       sub    edx, edx ; 0
22653 00006B1A BB00040000             <1>       mov    ebx, 1024 ; 400h
22654 00006B1F F7F3                   <1>       div    ebx
22655                                  <1>       ; EAX = Count of 3 FAT sectors
22656                                  <1>       ; EDX = Buffer entry index
22657 00006B21 89C1                   <1>       mov    ecx, eax
22658                                  <1>       ;mov   eax, 3
22659 00006B23 B003                   <1>       mov    al, 3
22660 00006B25 F7E2                   <1>       mul    edx ; Multiply by 3
22661 00006B27 66D1E8                 <1>       shr    ax, 1 ; Divide by 2
22662 00006B2A 89C3                   <1>       mov    ebx, eax ; Buffer byte offset
22663 00006B2C 81C3001C0900           <1>       add    ebx, FAT_Buffer
22664 00006B32 89C8                   <1>       mov    eax, ecx
22665                                  <1>       ;mov   edx, 3
22666 00006B34 66BA0300               <1>       mov    dx, 3
22667 00006B38 F7E2                   <1>       mul    edx
22668                                  <1>       ; EAX = FAT Beginning Sector
22669                                  <1>       ; EDX = 0
22670 00006B3A 8A0E                   <1>       mov    cl, [esi+LD_Name]
22671                                  <1>       ;cmp   byte [FAT_BuffValidData], 0
22672                                  <1>       ;jna   short fd_init_load_FAT_sectors0
22673 00006B3C 3A0D[A7560100]         <1>       cmp    cl, [FAT_BuffDrvName]
22674 00006B42 751E                   <1>       jne    short fd_init_load_FAT_sectors0
22675 00006B44 3B05[AA560100]         <1>       cmp    eax, [FAT_BuffSector]
22676 00006B4A 751C                   <1>       jne    short fd_init_load_FAT_sectors1
22677                                  <1>       ;mov   eax, [FAT_CurrentCluster]
22678 00006B4C A0[A2560100]           <1>       mov    al, [FAT_CurrentCluster]
22679                                  <1>       ;shr   eax, 1
22680 00006B51 D0E8                   <1>       shr    al, 1
22681 00006B53 668B03                 <1>       mov    ax, [ebx]
22682 00006B56 7306                   <1>       jnc    short fd_init_gnc_even
22683 00006B58 66C1E804               <1>       shr    ax, 4
22684                                  <1> fd_init_gnc_clc_retn:
22685 00006B5C F8                     <1>       clc
22686 00006B5D C3                     <1>       retn
22687                                  <1>
22688                                  <1> fd_init_gnc_even:
22689 00006B5E 80E40F                 <1>       and    ah, 0Fh
22690 00006B61 C3                     <1>       retn
22691                                  <1>
22692                                  <1> fd_init_load_FAT_sectors0:
22693 00006B62 880D[A7560100]         <1>       mov    [FAT_BuffDrvName], cl
22694                                  <1> fd_init_load_FAT_sectors1:
22695 00006B68 C605[A6560100]00       <1>       mov    byte [FAT_BuffValidData], 0
22696 00006B6F A3[AA560100]           <1>       mov    [FAT_BuffSector], eax
22697 00006B74 034660                 <1>       add    eax, [esi+LD_FATBegin]
22698 00006B77 BB001C0900             <1>       mov    ebx, FAT_Buffer
22699                                  <1>       ;movzx ecx, word [esi+LD_BPB+BPB_FATSz16]
22700 00006B7C 668B4E1C               <1>       mov    cx, [esi+LD_BPB+BPB_FATSz16]
22701 00006B80 662B0D[AA560100]       <1>       sub    cx, [FAT_BuffSector]
22702                                  <1>         ;cmp ecx, 3
22703 00006B87 6683F903               <1>       cmp    cx, 3
22704 00006B8B 7605                   <1>       jna    short fdinit_pass_fix_sector_count_3
22705                                  <1>       ;mov   ecx, 3
22706 00006B8D B903000000             <1>       mov    ecx, 3
22707                                  <1> fdinit_pass_fix_sector_count_3:
22708 00006B92 E80D860000             <1>       call   chs_read
22709 00006B97 730D                   <1>       jnc    short fd_init_FAT_sectors_no_load_error
22710 00006B99 C605[A6560100]00       <1>       mov    byte [FAT_BuffValidData], 0
22711                                  <1>              ; Drv not ready or read Error !
22712 00006BA0 B80F000000             <1>       mov    eax, ERR_DRV_NOT_RDY ; 15
22713                                  <1>       ;xor   edx, edx
22714 00006BA5 C3                     <1>       retn
22715                                  <1>
22716                                  <1> fd_init_FAT_sectors_no_load_error:
22717 00006BA6 C605[A6560100]01       <1>       mov    byte [FAT_BuffValidData], 1
22718 00006BAD A1[A2560100]           <1>       mov    eax, [FAT_CurrentCluster]
22719 00006BB2 E961FFFFFF             <1>         jmp     fd_init_get_next_cluster_readnext
22720                                  <1>
22721                                  <1> get_FAT_volume_name:
22722                                  <1>       ; 10/01/2016 (TRDOS 386 = TRDOS v2.0)
22723                                  <1>       ; 12/09/2009
22724                                  <1>       ; INPUT ->
22725                                  <1>       ;      BH = Logical DOS drive number (0,1,2,3,4 ...)
22726                                  <1>       ;       BL = 0
22727                                  <1>       ; OUTPUT ->
22728                                  <1>       ;      CF = 0 -> ESI = Volume name address
22729                                  <1>       ;      CF = 1 -> Root volume name not found
22730                                  <1>
22731                                  <1>       ;mov   ah, 0FFh
22732                                  <1>       ;mov   al, [Last_Dos_DiskNo]
22733                                  <1>       ;cmp   al, bh
22734                                  <1>       ;jb     short loc_gfvn_dir_load_err
22735                                  <1>
22736 00006BB7 89DE                   <1>       mov    esi, ebx
22737 00006BB9 81E600FF0000           <1>       and    esi, 0FF00h ; esi = bh
22738 00006BBF 81C600010900           <1>       add    esi, Logical_DOSDisks
22739 00006BC5 8A06                   <1>       mov     al, [esi+LD_Name]
22740 00006BC7 8A6603                 <1>       mov     ah, [esi+LD_FATType]
```

```
22741 00006BCA 80FC01              <1>       cmp     ah, 1
22742 00006BCD 7210                <1>       jb    short loc_gfvn_dir_load_err
22743 00006BCF 3C41                <1>       cmp   al, 'A'
22744 00006BD1 720C                <1>       jb      short loc_gfvn_dir_load_err
22745 00006BD3 80FC02              <1>       cmp   ah, 2
22746 00006BD6 7708                <1>       ja      short get_FAT32_root_cluster
22747                              <1>
22748 00006BD8 E89B4E0000          <1>       call    load_FAT_root_directory
22749 00006BDD 730B                <1>       jnc     short loc_get_volume_name
22750                              <1>
22751                              <1> loc_gfvn_dir_load_err:
22752 00006BDF C3                  <1>       retn
22753                              <1>
22754                              <1> get_FAT32_root_cluster:
22755 00006BE0 8B4632              <1>       mov   eax, [esi+LD_BPB+BPB_RootClus]
22756 00006BE3 E81B4F0000          <1>       call    load_FAT_sub_directory
22757 00006BE8 7224                <1>       jc    short loc_get_volume_name_retn
22758                              <1>
22759                              <1> loc_get_volume_name:
22760 00006BEA BE00000800          <1>         mov     esi, Directory_Buffer
22761 00006BEF 6631C9              <1>       xor   cx, cx ; 0
22762                              <1> check_root_volume_name:
22763 00006BF2 8A06                <1>       mov   al, [esi]
22764 00006BF4 08C0                <1>       or      al, al
22765 00006BF6 7416                <1>       jz      short loc_get_volume_name_retn
22766 00006BF8 807E0B08            <1>       cmp     byte [esi+0Bh], 08h
22767 00006BFC 7410                <1>       je      short loc_get_volume_name_retn
22768 00006BFE 663B0D[BB560100]    <1>       cmp     cx, [DirBuff_LastEntry]
22769 00006C05 7308                <1>       jnb     short pass_check_root_volume_name
22770 00006C07 6641                <1>       inc     cx
22771 00006C09 83C620              <1>       add     esi, 32
22772 00006C0C EBE4                <1>       jmp     short check_root_volume_name
22773                              <1>
22774                              <1> loc_get_volume_name_retn:
22775 00006C0E C3                  <1>       retn
22776                              <1>
22777                              <1> pass_check_root_volume_name:
22778 00006C0F 803D[B7560100]03    <1>       cmp   byte [DirBuff_FATType], 3
22779 00006C16 7230                <1>       jb    short loc_get_volume_name_retn_xor
22780                              <1>
22781 00006C18 BB001C0900          <1>       mov   ebx, FAT_Buffer
22782 00006C1D BE00010900          <1>       mov   esi, Logical_DOSDisks
22783 00006C22 31C0                <1>       xor   eax, eax
22784 00006C24 8A25[B6560100]      <1>       mov   ah, [DirBuff_DRV]
22785 00006C2A 80EC41              <1>       sub   ah, 'A'
22786 00006C2D 01C6                <1>       add   esi, eax
22787 00006C2F A1[BD560100]        <1>       mov   eax, [DirBuff_Cluster]
22788 00006C34 E8E44C0000          <1>       call  get_next_cluster
22789 00006C39 7305                <1>       jnc   short loc_gfvn_load_FAT32_dir_cluster
22790                              <1>
22791 00006C3B 83F801              <1>       cmp     eax, 1
22792 00006C3E F5                  <1>       cmc
22793 00006C3F C3                  <1>       retn
22794                              <1>
22795                              <1> loc_gfvn_load_FAT32_dir_cluster:
22796 00006C40 E8BE4E0000          <1>       call  load_FAT_sub_directory
22797 00006C45 73A3                <1>       jnc   short loc_get_volume_name
22798 00006C47 C3                  <1>       retn
22799                              <1>
22800                              <1> loc_get_volume_name_retn_xor:
22801 00006C48 31C0                <1>       xor   eax, eax
22802 00006C4A C3                  <1>       retn
22803                              <1>
22804                              <1> get_media_change_status:
22805                              <1>       ; 10/01/2016 (TRDOS 386 = TRDOS v2.0)
22806                              <1>       ; 09/09/2009
22807                              <1>       ; INPUT:
22808                              <1>       ;     DL = Drive number (physical)
22809                              <1>       ; OUTPUT: clc & AH = 6 media changed
22810                              <1>       ;     clc & AH = 0 media not changed
22811                              <1>       ;     stc -> Drive not ready or an error
22812                              <1>
22813 00006C4B B416                <1>       mov   ah, 16h
22814 00006C4D E8B4D5FFFF          <1>       call  int13h
22815 00006C52 80FC06              <1>       cmp   ah, 06h
22816 00006C55 7405                <1>       je    short loc_gmc_status_retn
22817 00006C57 08E4                <1>       or    ah, ah
22818 00006C59 7401                <1>       jz    short loc_gmc_status_retn
22819                              <1> loc_gmc_status_stc_retn:
22820 00006C5B F9                  <1>       stc
22821                              <1> loc_gmc_status_retn:
22822 00006C5C C3                  <1>       retn
22823                                  %include 'trdosk3.s' ; 06/01/2016
22824                              <1> ; ****************************************************************************
22825                              <1> ; TRDOS386.ASM (TRDOS 386 Kernel - v2.0.0) - MAIN PROGRAM : trdosk3.s
22826                              <1> ; ----------------------------------------------------------------------------
22827                              <1> ; Last Update: 07/01/2017
22828                              <1> ; ----------------------------------------------------------------------------
22829                              <1> ; Beginning: 06/01/2016
22830                              <1> ; ----------------------------------------------------------------------------
22831                              <1> ; Assembler: NASM version 2.11 (trdos386.s)
22832                              <1> ; ----------------------------------------------------------------------------
22833                              <1> ; Derived from TRDOS Operating System v1.0 (8086) source code by Erdogan Tan
22834                              <1> ; MAINPROG.ASM (09/11/2011)
22835                              <1> ; ****************************************************************************
22836                              <1> ; MAINPROG.ASM [ TRDOS KERNEL - COMMAND EXECUTER SECTION - MAIN PROGRAM ]
22837                              <1> ; (c) 2004-2011  Erdogan TAN  [ 17/01/2004 ]  Last Update: 09/11/2011
22838                              <1> ; CMD_INTR.ASM [ TRDOS Command Interpreter Procedure ] Last Update: 09/11/2011
22839                              <1> ; DIR.ASM [ DIRECTORY FUNCTIONS ] Last Update: 09/10/2011
22840                              <1> ; FILE.ASM [ FILE FUNCTIONS ] Last Update: 09/10/2011
22841                              <1>
22842                              <1> change_current_drive:
```

```
22843                                  <1>       ; 16/10/2016
22844                                  <1>       ; 02/02/2016
22845                                  <1>       ; 15/01/2016 (TRDOS 386 = TRDOS v2.0)
22846                                  <1>       ; 18/08/2011
22847                                  <1>       ; 09/09/2009
22848                                  <1>       ; INPUT:
22849                                  <1>       ;   DL = Logical DOS Drive Number
22850                                  <1>       ; OUTPUT:
22851                                  <1>       ;  cf=1 -> Not successful
22852                                  <1>       ;   EAX = Error code
22853                                  <1>       ;  cf=0 ->
22854                                  <1>       ;   EAX = 0 (successful)
22855                                  <1>
22856 00006C5D 31DB                   <1>       xor    ebx, ebx
22857 00006C5F 88D7                   <1>       mov    bh, dl
22858                                  <1>
22859                                  <1>       ;cmp   dl, 1
22860                                  <1>       ;jna   short loc_ccdrv_initial_media_change_check
22861                                  <1>       ;cmp   bh, [Last_Dos_DiskNo]
22862                                  <1>       ;ja    short loc_ccdrv_drive_not_ready_err
22863                                  <1>
22864                                  <1> loc_ccdrv_initial_media_change_check:
22865 00006C61 BE00010900             <1>       mov    esi, Logical_DOSDisks
22866 00006C66 01DE                   <1>       add    esi, ebx
22867                                  <1> loc_ccdrv_dos_drive_name_check:
22868 00006C68 80FA02                 <1>       cmp    dl, 2
22869 00006C6B 720F                   <1>       jb     short loc_ccdrv_dos_drive_name_check_ok
22870                                  <1>
22871 00006C6D 8A06                   <1>       mov    al, [esi+LD_Name]
22872 00006C6F 2C41                   <1>       sub    al, 'A'
22873 00006C71 38D0                   <1>       cmp    al, dl
22874 00006C73 7407                   <1>       je     short loc_ccdrv_dos_drive_name_check_ok
22875                                  <1>
22876                                  <1> loc_ccdrv_drive_not_ready_err:
22877                                  <1>       ; 16/10/2016 (15h -> 15)
22878 00006C75 B80F000000             <1>       mov    eax, 15 ; Drive not ready
22879                                  <1> loc_change_current_drive_stc_retn:
22880 00006C7A F9                     <1>       stc
22881 00006C7B C3                     <1>       retn
22882                                  <1>
22883                                  <1> loc_ccdrv_dos_drive_name_check_ok:
22884 00006C7C 8A667E                 <1>       mov    ah, [esi+LD_MediaChanged]
22885 00006C7F 80FC06                 <1>       cmp    ah, 6  ; VOLUME NAME CHECK/MOVE SIGN
22886 00006C82 7455                   <1>       je     short loc_ccdrv_get_FAT_volume_name_0
22887                                  <1>
22888 00006C84 80FA01                 <1>       cmp    dl, 1
22889 00006C87 777D                   <1>       ja     short loc_gmcs_init_drv_hd
22890                                  <1>
22891                                  <1> loc_gmcs_init_drv_fd:
22892 00006C89 08E4                   <1>       or     ah, ah
22893                                  <1>       ; AH = 1 is initialization sign (invalid_fd_parameter)
22894 00006C8B 7517                   <1>       jnz    short loc_ccdrv_call_fd_init
22895                                  <1>
22896 00006C8D E8B9FFFFFF             <1>       call   get_media_change_status
22897 00006C92 72E1                   <1>       jc     short loc_ccdrv_drive_not_ready_err
22898                                  <1>
22899 00006C94 20E4                   <1>       and    ah, ah
22900 00006C96 7476                   <1>       jz     short loc_change_current_drv3
22901                                  <1>
22902 00006C98 80F406                 <1>       xor    ah, 6
22903 00006C9B 75D8                   <1>       jnz    short loc_ccdrv_drive_not_ready_err
22904                                  <1>
22905                                  <1> loc_ccdrv_call_fd_init_check_vol_id:
22906 00006C9D E8490A0000             <1>       call   get_volume_serial_number
22907 00006CA2 730D                   <1>       jnc    short loc_ccdrv_check_vol_serial
22908                                  <1>
22909                                  <1> loc_ccdrv_call_fd_init:
22910 00006CA4 E872FCFFFF             <1>       call   floppy_drv_init
22911 00006CA9 731A                   <1>       jnc    short loc_reset_drv_fd_current_dir
22912                                  <1>
22913                                  <1> loc_ccdrv_fdinit_fail_retn:
22914                                  <1>       ; 16/10/2016
22915 00006CAB B80F000000             <1>       mov    eax, 15 ; Drive not ready
22916 00006CB0 C3                     <1>       retn
22917                                  <1>
22918                                  <1> loc_ccdrv_check_vol_serial:
22919 00006CB1 A3[844E0100]           <1>       mov    [Current_VolSerial], eax
22920                                  <1>       ;mov   dl, bh
22921 00006CB6 E860FCFFFF             <1>       call   floppy_drv_init
22922 00006CBB 72EE                   <1>       jc     short loc_ccdrv_fdinit_fail_retn
22923                                  <1>
22924 00006CBD 3B05[844E0100]         <1>       cmp    eax, [Current_VolSerial]
22925 00006CC3 7445                   <1>       je     short loc_change_current_drv2
22926                                  <1>
22927                                  <1> loc_reset_drv_fd_current_dir:
22928 00006CC5 31C0                   <1>       xor    eax, eax
22929 00006CC7 88467F                 <1>         mov  [esi+LD_CDirLevel], al
22930 00006CCA 89F7                   <1>       mov    edi, esi
22931 00006CCC 81C780000000           <1>       add    edi, LD_CurrentDirectory
22932 00006CD2 B920000000             <1>       mov    ecx, 32
22933 00006CD7 F3AB                   <1>       rep    stosd
22934                                  <1>
22935                                  <1> loc_ccdrv_get_FAT_volume_name_0:
22936 00006CD9 8A4603                 <1>       mov    al, [esi+LD_FATType]
22937 00006CDC 08C0                   <1>       or     al, al
22938 00006CDE 742A                   <1>       jz     short loc_change_current_drv2
22939                                  <1>
22940 00006CE0 56                     <1>       push   esi
22941 00006CE1 3C02                   <1>       cmp    al, 2
22942 00006CE3 7705                   <1>       ja     short loc_ccdrv_get_FAT32_vol_name
22943                                  <1>
22944                                  <1> loc_ccdrv_get_FAT2_16_vol_name:
```

```
22945 00006CE5 83C631              <1>       add    esi, LD_BPB + VolumeLabel
22946 00006CE8 EB03                <1>       jmp    short loc_ccdrv_get_FAT_volume_name_1
22947                              <1>
22948                              <1> loc_ccdrv_get_FAT32_vol_name:
22949 00006CEA 83C64D              <1>       add    esi, LD_BPB + FAT32_VolLab
22950                              <1> loc_ccdrv_get_FAT_volume_name_1:
22951 00006CED 53                  <1>       push   ebx
22952 00006CEE 56                  <1>       push   esi
22953 00006CEF E8C3FEFFFF          <1>       call   get_FAT_volume_name
22954 00006CF4 5F                  <1>       pop    edi
22955 00006CF5 5B                  <1>       pop    ebx
22956                              <1>       ; BL = 0
22957 00006CF6 720B                <1>       jc     short loc_change_current_drv1
22958 00006CF8 20C0                <1>       and    al, al
22959 00006CFA 7407                <1>       jz     short loc_change_current_drv1
22960                              <1>
22961                              <1> loc_ccdrv_move_FAT_volume_name:
22962 00006CFC B90B000000          <1>       mov    ecx, 11
22963 00006D01 F3A4                <1>       rep    movsb
22964                              <1>
22965                              <1> loc_change_current_drv1:
22966 00006D03 5E                  <1>       pop    esi
22967 00006D04 EB04                <1>       jmp    short loc_change_current_drv2
22968                              <1>
22969                              <1> loc_gmcs_init_drv_hd:
22970 00006D06 08E4                <1>       or     ah, ah
22971 00006D08 7404                <1>       jz     short loc_change_current_drv3
22972                              <1>       ; BL = 0, BH = Logical DOS drive number
22973                              <1> loc_change_current_drv2:
22974 00006D0A C6467E00            <1>       mov    byte [esi+LD_MediaChanged], 0
22975                              <1> loc_change_current_drv3:
22976 00006D0E 883D[8E4E0100]      <1>       mov    [Current_Drv], bh
22977                              <1>
22978                              <1>       ;call  restore_current_directory
22979                              <1>       ;retn
22980                              <1>
22981                              <1> restore_current_directory:
22982                              <1>       ; 11/02/2016
22983                              <1>       ; 15/01/2016 (TRDOS 386 = TRDOS v2.0)
22984                              <1>       ; 25/01/2010
22985                              <1>       ; 12/10/2009
22986                              <1>       ;
22987                              <1>       ; INPUT:
22988                              <1>       ;   ESI = Logical DOS Drive Description Table
22989                              <1>       ;
22990                              <1>       ; OUTPUT:
22991                              <1>       ;   ESI = Logical DOS Drive Description Table
22992                              <1>       ;   EDI = offset Current_Dir_Drv
22993                              <1>
22994 00006D14 8A4603              <1>       mov    al, [esi+LD_FATType]
22995 00006D17 A2[8D4E0100]        <1>       mov    [Current_FATType], al
22996                              <1>
22997 00006D1C 8A26                <1>       mov    ah, [esi+LD_Name]
22998 00006D1E 8825[8F4E0100]      <1>       mov    [Current_Dir_Drv], ah
22999                              <1>
23000 00006D24 20C0                <1>       and    al, al
23001 00006D26 741D                <1>       jz     short loc_restore_FS_current_directory
23002                              <1>
23003                              <1> loc_restore_FAT_current_directory:
23004 00006D28 8A667F              <1>       mov    ah, [esi+LD_CDirLevel]
23005 00006D2B 8825[8C4E0100]      <1>       mov    [Current_Dir_Level], ah
23006 00006D31 08E4                <1>       or     ah, ah
23007 00006D33 7416                <1>         jz   short loc_ccdrv_reset_cdir_FAT_12_16_32_fcluster
23008                              <1>
23009 00006D35 0FB6D4              <1>       movzx  edx, ah
23010 00006D38 C0E204              <1>       shl    dl, 4 ; * 16
23011 00006D3B 01F2                <1>         add  edx, esi
23012 00006D3D 8B828C000000        <1>       mov    eax, [edx+LD_CurrentDirectory+12]
23013 00006D43 EB2C                <1>       jmp    short loc_ccdrv_reset_cdir_FAT_fcluster
23014                              <1>
23015                              <1> loc_restore_FS_current_directory:
23016 00006D45 E8F44D0000          <1>       call   load_current_FS_directory
23017 00006D4A C3                  <1>       retn
23018                              <1>
23019                              <1> loc_ccdrv_reset_cdir_FAT_12_16_32_fcluster:
23020 00006D4B 3C03                <1>       cmp    al, 3
23021 00006D4D 7205                <1>       jb     short loc_ccdrv_reset_cdir_FAT_12_16_fcluster
23022                              <1> loc_ccdrv_reset_cdir_FAT32_fcluster:
23023 00006D4F 8B4632              <1>       mov    eax, [esi+LD_BPB+FAT32_RootFClust]
23024 00006D52 EB04                <1>       jmp    short loc_ccdrv_check_rootdir_sign
23025                              <1> loc_ccdrv_reset_cdir_FAT_12_16_fcluster:
23026 00006D54 30C0                <1>       xor    al, al  ; xor eax, eax
23027 00006D56 31D2                <1>       xor    edx, edx
23028                              <1> loc_ccdrv_check_rootdir_sign:
23029 00006D58 80BE8000000000      <1>       cmp    byte [esi+LD_CurrentDirectory], 0
23030 00006D5F 7510                <1>       jne    short loc_ccdrv_reset_cdir_FAT_fcluster
23031                              <1> loc_ccdrv_set_rootdir_FAT_fcluster:
23032 00006D61 89868C000000        <1>         mov     [esi+LD_CurrentDirectory+12], eax
23033 00006D67 C78680000000524F4F- <1>       mov    dword [esi+LD_CurrentDirectory], 'ROOT'
23034 00006D70 54                  <1>
23035                              <1>
23036                              <1> loc_ccdrv_reset_cdir_FAT_fcluster:
23037 00006D71 A3[884E0100]        <1>       mov    [Current_Dir_FCluster], eax
23038                              <1>
23039 00006D76 BF[EF560100]        <1>       mov    edi, PATH_Array
23040 00006D7B 89F2                <1>       mov    edx, esi
23041 00006D7D 81C680000000        <1>       add    esi, LD_CurrentDirectory
23042 00006D83 B920000000          <1>       mov    ecx, 32
23043 00006D88 F3A5                <1>       rep    movsd
23044                              <1>
23045 00006D8A E8832D0000          <1>       call   change_prompt_dir_string
23046                              <1>
```

```
23047 00006D8F 89D6                <1>       mov    esi, edx
23048                              <1>
23049 00006D91 29C0                <1>         sub  eax, eax
23050                              <1>        ;sub  edx, edx
23051 00006D93 BF[8F4E0100]        <1>       mov    edi, Current_Dir_Drv
23052                              <1>
23053 00006D98 A2[98020100]        <1>       mov    [Restore_CDIR], al ; 0
23054 00006D9D C3                  <1>       retn
23055                              <1>
23056                              <1> dos_prompt:
23057                              <1>       ; 06/05/2016
23058                              <1>       ; 30/01/2016
23059                              <1>       ; 29/01/2016
23060                              <1>       ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
23061                              <1>       ; 15/09/2011
23062                              <1>       ; 13/09/2009
23063                              <1>       ; 2004-2005
23064                              <1>
23065                              <1>       ; 06/05/2016
23066 00006D9E C705[4C5B0100]-     <1>       mov    dword [mainprog_return_addr], return_from_cmd_interpreter
23067 00006DA4 [526E0000]          <1>
23068                              <1>
23069                              <1> loc_TRDOS_prompt:
23070 00006DA8 BF[8E4F0100]        <1>       mov    edi, TextBuffer
23071 00006DAD C6075B              <1>       mov    byte [edi], "["
23072 00006DB0 47                  <1>       inc    edi
23073 00006DB1 BE[EB020100]        <1>       mov    esi, TRDOSPromptLabel
23074                              <1> get_next_prompt_label_char:
23075 00006DB6 803E20              <1>       cmp    byte [esi], 20h
23076 00006DB9 7203                <1>       jb     short pass_prompt_label
23077 00006DBB A4                  <1>       movsb
23078 00006DBC EBF8                <1>       jmp    short get_next_prompt_label_char
23079                              <1> pass_prompt_label:
23080 00006DBE C6075D              <1>       mov    byte [edi], "]"
23081 00006DC1 47                  <1>       inc    edi
23082 00006DC2 C60720              <1>       mov    byte [edi], 20h
23083 00006DC5 47                  <1>       inc    edi
23084 00006DC6 BE[8F4E0100]        <1>       mov    esi, Current_Dir_Drv
23085 00006DCB 66A5                <1>       movsw
23086 00006DCD A4                  <1>       movsb
23087                              <1> loc_prompt_current_directory:
23088 00006DCE 803E20              <1>       cmp    byte [esi], 20h
23089 00006DD1 7203                <1>       jb     short pass_prompt_current_directory
23090 00006DD3 A4                  <1>       movsb
23091 00006DD4 EBF8                <1>       jmp    short loc_prompt_current_directory
23092                              <1> pass_prompt_current_directory:
23093 00006DD6 C6073E              <1>       mov    byte [edi], '>'
23094 00006DD9 47                  <1>       inc    edi
23095 00006DDA C60700              <1>       mov    byte [edi], 0
23096 00006DDD BE[8E4F0100]        <1>       mov    esi, TextBuffer
23097 00006DE2 E876F5FFFF          <1>       call   print_msg
23098                              <1>
23099                              <1>       ;sub   bh, bh ; video page = 0
23100                              <1>       ;call  get_cpos ; get cursor position
23101 00006DE7 668B15[E64D0100]    <1>       mov    dx, [CURSOR_POSN] ; video page 0
23102 00006DEE 8815[EE4E0100]      <1>       mov    [CursorColumn], dl
23103                              <1>
23104                              <1>       ; 30/01/2016 (to show cursor on the row, again)
23105                              <1>       ; (Initial color attributes of video page 0 is 0)
23106                              <1>       ; (see: 'StartPMP' in trdos386.s)
23107                              <1>       ;
23108                              <1>       ;mov   edi, 0B8000h ; start of video page 0
23109                              <1>       ;movzx ecx, dl ; column
23110                              <1>       ;mov   al, 80
23111                              <1>       ;mul   dh
23112                              <1>       ;add   ax, cx
23113                              <1>       ;shl   ax, 1 ; character + attribute
23114                              <1>       ;add   di, ax ; (2*80*row) + (2*column)
23115                              <1>       ;neg   cl
23116                              <1>       ;add   cl, 80
23117                              <1>       ;mov   ax, 700h ;  ah = 7 (color attribute)
23118                              <1>       ;rep   stosw
23119                              <1>
23120                              <1> loc_rw_char:
23121 00006DF4 E899000000          <1>       call   rw_char
23122                              <1> loc_move_command:
23123 00006DF9 BE[3E4F0100]        <1>       mov    esi, CommandBuffer
23124 00006DFE 89F7                <1>       mov    edi, esi
23125 00006E00 31C9                <1>       xor    ecx, ecx
23126                              <1> first_command_char:
23127 00006E02 AC                  <1>       lodsb
23128 00006E03 3C20                <1>       cmp    al, 20h
23129 00006E05 772E                <1>       ja     short pass_space_control
23130 00006E07 7241                <1>       jb     short loc_move_cmd_arguments_ok
23131 00006E09 81FE[8D4F0100]      <1>       cmp    esi, CommandBuffer + 79
23132 00006E0F 72F1                <1>       jb     short first_command_char
23133 00006E11 EB37                <1>       jmp    short loc_move_cmd_arguments_ok
23134                              <1>
23135                              <1> next_command_char:
23136 00006E13 AC                  <1>       lodsb
23137 00006E14 3C20                <1>       cmp    al, 20h
23138 00006E16 771D                <1>       ja     short pass_space_control
23139 00006E18 7230                <1>       jb     short loc_move_cmd_arguments_ok
23140                              <1>
23141                              <1> loc_1st_cmd_arg: ; 30/01/2016
23142 00006E1A AC                  <1>       lodsb
23143 00006E1B 3C20                <1>       cmp    al, 20h
23144 00006E1D 74FB                <1>       je     short loc_1st_cmd_arg
23145 00006E1F 7229                <1>       jb     short loc_move_cmd_arguments_ok
23146                              <1>
23147 00006E21 C60700              <1>         mov     byte [edi], 0
23148 00006E24 47                  <1>       inc    edi
```

```
23149                                   <1>
23150                                   <1> loc_move_cmd_arguments:
23151 00006E25 AA                       <1>       stosb
23152 00006E26 81FE[8D4F0100]           <1>       cmp    esi, CommandBuffer + 79
23153 00006E2C 731C                     <1>       jnb    short loc_move_cmd_arguments_ok
23154 00006E2E AC                       <1>         lodsb
23155 00006E2F 3C20                     <1>       cmp    al, 20h
23156 00006E31 73F2                     <1>       jnb    short loc_move_cmd_arguments
23157 00006E33 EB15                     <1>       jmp    short loc_move_cmd_arguments_ok
23158                                   <1>
23159                                   <1> pass_space_control:
23160 00006E35 3C61                     <1>       cmp    al, 61h
23161 00006E37 7206                     <1>       jb     short pass_capitalize
23162 00006E39 3C7A                     <1>       cmp    al, 7Ah
23163 00006E3B 7702                     <1>       ja     short pass_capitalize
23164 00006E3D 24DF                     <1>       and    al, 0DFh
23165                                   <1> pass_capitalize:
23166 00006E3F AA                       <1>       stosb
23167 00006E40 FEC1                     <1>       inc     cl
23168 00006E42 81FE[8D4F0100]           <1>         cmp     esi, CommandBuffer + 79
23169 00006E48 72C9                     <1>       jb      short next_command_char
23170                                   <1>
23171                                   <1> loc_move_cmd_arguments_ok:
23172 00006E4A C60700                   <1>         mov     byte [edi], 0
23173                                   <1>
23174                                   <1> call_command_interpreter:
23175 00006E4D E8D4080000               <1>       call    command_interpreter
23176                                   <1>
23177                                   <1> return_from_cmd_interpreter:
23178 00006E52 B950000000               <1>         mov  ecx, 80
23179                                   <1>       ;mov   cx, 80
23180 00006E57 BF[3E4F0100]             <1>       mov    edi, CommandBuffer
23181 00006E5C 30C0                     <1>       xor    al, al
23182 00006E5E F3AA                     <1>       rep    stosb
23183                                   <1>       ;cmp   byte [Program_Exit], 0
23184                                   <1>       ;ja    short loc_terminate_trdos
23185                                   <1>
23186                                   <1>       ; 16/01/2016
23187 00006E60 803D[C25E0000]03         <1>       cmp    byte [CRT_MODE], 3 ; 80*25 color
23188 00006E67 741D                     <1>       je     short pass_set_txt_mode
23189                                   <1>
23190 00006E69 E8F5A6FFFF               <1>       call   set_txt_mode ; set vide mode to 03h
23191                                   <1>       ; 07/01/2017
23192 00006E6E 30C0                     <1>       xor    al, al
23193                                   <1>
23194                                   <1> loc_check_active_page:
23195                                   <1>       ;xor   al, al
23196 00006E70 3805[F64D0100]           <1>       cmp    [ACTIVE_PAGE], al ; 0
23197 00006E76 0F842CFFFFFF             <1>         je      loc_TRDOS_prompt
23198                                   <1>       ; AL = 0 = video page 0
23199 00006E7C E8FBAAFFFF               <1>       call   set_active_page
23200 00006E81 E922FFFFFF               <1>         jmp     loc_TRDOS_prompt ; infinitive loop
23201                                   <1>
23202                                   <1> pass_set_txt_mode:
23203 00006E86 BE[030F0100]             <1>       mov    esi, nextline
23204 00006E8B E8CDF4FFFF               <1>       call   print_msg
23205 00006E90 EBDE                     <1>       jmp     short loc_check_active_page
23206                                   <1>
23207                                   <1> rw_char:
23208                                   <1>       ; 13/05/2016
23209                                   <1>       ; 30/01/2016
23210                                   <1>       ; 29/01/2016
23211                                   <1>       ; 17/01/2016 (TRDOS 386 = TRDOS v2.0)
23212                                   <1>       ; 2004-2005
23213                                   <1>
23214                                   <1>       ; DH = cursor row, DL = cursor column
23215                                   <1>       ; BH = 0 = video page number (active page)
23216                                   <1>
23217                                   <1>       ;xor   bh, bh ; 0 = video page 0
23218                                   <1>
23219                                   <1> readnextchar:
23220 00006E92 30E4                     <1>       xor     ah, ah
23221 00006E94 E87D9DFFFF               <1>       call   int16h
23222 00006E99 20C0                     <1>       and    al, al
23223 00006E9B 7434                     <1>       jz     short loc_arrow
23224 00006E9D 3CE0                     <1>       cmp    al, 0E0h
23225 00006E9F 7430                     <1>       je     short loc_arrow
23226 00006EA1 3C08                     <1>       cmp    al, 08h
23227 00006EA3 7544                     <1>       jne    short char_return
23228                                   <1> loc_back:
23229 00006EA5 3A15[EE4E0100]           <1>       cmp    dl, [CursorColumn]
23230 00006EAB 76E5                     <1>       jna     short readnextchar
23231                                   <1> prev_column:
23232 00006EAD FECA                     <1>       dec    dl
23233                                   <1> set_cursor_pos:
23234 00006EAF 6652                     <1>       push   dx
23235                                   <1>       ;xor   bh, bh ; 0 = video page 0
23236                                   <1>       ; DH = Row, DL = Column
23237 00006EB1 E892AEFFFF               <1>       call   _set_cpos ; 17/01/2016
23238 00006EB6 665A                     <1>         pop  dx
23239                                   <1>       ;movzx ebx, dl
23240 00006EB8 88D3                     <1>       mov    bl, dl
23241 00006EBA 2A1D[EE4E0100]           <1>       sub    bl, [CursorColumn]
23242 00006EC0 B020                     <1>       mov    al, 20h
23243 00006EC2 8883[3E4F0100]           <1>       mov    [CommandBuffer+ebx], al
23244                                   <1>       ;sub   bh, bh ; video page 0
23245                                   <1>       ;mov   cx, 1
23246 00006EC8 B307                     <1>       mov    bl, 7 ; color attribute
23247 00006ECA E86AADFFFF               <1>       call   _write_c_current ; 17/01/2016
23248                                   <1>       ;mov   dx, [CURSOR_POSN]
23249 00006ECF EBC1                     <1>       jmp    short readnextchar
23250                                   <1> loc_arrow:
```

```
23251 00006ED1 80FC4B              <1>       cmp    ah, 4Bh
23252 00006ED4 74CF                <1>       je     short loc_back
23253 00006ED6 80FC53              <1>       cmp    ah, 53h
23254 00006ED9 74CA                <1>       je      short loc_back
23255 00006EDB 80FC4D              <1>       cmp    ah, 4Dh
23256 00006EDE 75B2                <1>       jne    short readnextchar
23257 00006EE0 80FA4F              <1>       cmp    dl, 79
23258 00006EE3 73AD                <1>       jnb    short readnextchar
23259 00006EE5 FEC2                <1>       inc    dl
23260 00006EE7 EBC6                <1>       jmp    short set_cursor_pos
23261                              <1> char_return:
23262 00006EE9 0FB6DA              <1>       movzx  ebx, dl
23263 00006EEC 2A1D[EE4E0100]      <1>       sub    bl, [CursorColumn]
23264 00006EF2 3C20                <1>       cmp    al, 20h
23265 00006EF4 7220                <1>       jb     short loc_escape
23266 00006EF6 8883[3E4F0100]      <1>       mov    [CommandBuffer+ebx], al
23267 00006EFC 80FA4F              <1>       cmp    dl, 79
23268 00006EFF 7391                <1>       jnb    short readnextchar
23269 00006F01 66BB0700            <1>       mov    bx, 7 ; color attribute
23270 00006F05 E8A8ADFFFF          <1>       call   _write_tty
23271 00006F0A 668B15[E64D0100]    <1>       mov    dx, [CURSOR_POSN] ; video page 0
23272 00006F11 E97CFFFFFF          <1>         jmp     readnextchar
23273                              <1> loc_escape:
23274 00006F16 3C1B                <1>       cmp    al, 1Bh
23275 00006F18 7418                <1>       je     short rw_char_retn
23276                              <1>       ;
23277 00006F1A 3C0D                <1>       cmp    al, 0Dh ; CR
23278 00006F1C 0F8570FFFFFF        <1>         jne     readnextchar
23279                              <1>       ; 13/05/2016
23280 00006F22 66BB0700            <1>       mov    bx, 7 ; attribute/color (bl)
23281                              <1>                    ; video page 0 (bh=0)
23282 00006F26 E887ADFFFF          <1>       call   _write_tty
23283                              <1>       ;mov   bx, 7  ; attribute/color
23284                              <1>                     ; video page 0 (bh=0)
23285 00006F2B B00A                <1>       mov    al, 0Ah ; LF
23286 00006F2D E880ADFFFF          <1>       call   _write_tty
23287                              <1> rw_char_retn:
23288 00006F32 C3                  <1>       retn
23289                              <1>
23290                              <1> show_date:
23291                              <1>       ; 18/01/2016 (TRDOS 386 = TRDOS v2.0)
23292                              <1>         ; 2004-2005
23293                              <1>
23294                              <1>       ;mov   ah, 04h
23295                              <1>       ;call  int1Ah
23296 00006F33 E826EBFFFF          <1>       call   RTC_40 ; GET RTC DATE
23297                              <1>
23298 00006F38 88D0                <1>       mov    al, dl
23299 00006F3A E8CE9CFFFF          <1>       call   bcd_to_ascii
23300 00006F3F 66A3[D7030100]      <1>       mov    [Day], ax
23301                              <1>
23302 00006F45 88F0                <1>       mov    al, dh
23303 00006F47 E8C19CFFFF          <1>       call   bcd_to_ascii
23304 00006F4C 66A3[DA030100]      <1>       mov    [Month], ax
23305                              <1>
23306 00006F52 88E8                <1>       mov    al, ch
23307 00006F54 E8B49CFFFF          <1>       call   bcd_to_ascii
23308 00006F59 66A3[DD030100]      <1>       mov    [Century], ax
23309                              <1>
23310 00006F5F 88C8                <1>       mov    al, cl
23311 00006F61 E8A79CFFFF          <1>       call   bcd_to_ascii
23312 00006F66 66A3[DF030100]      <1>       mov    word [Year], ax
23313                              <1>
23314 00006F6C BE[C7030100]        <1>       mov    esi, Msg_Show_Date
23315 00006F71 E8E7F3FFFF          <1>       call   print_msg
23316                              <1>
23317 00006F76 C3                  <1>       retn
23318                              <1>
23319                              <1> set_date:
23320                              <1>       ; 13/05/2016
23321                              <1>       ; 18/01/2016 (TRDOS 386 = TRDOS v2.0)
23322                              <1>         ; 2004-2005
23323                              <1>
23324 00006F77 BE[AB030100]        <1>       mov    esi, Msg_Enter_Date
23325 00006F7C E8DCF3FFFF          <1>       call   print_msg
23326                              <1>
23327                              <1> loc_enter_day_1:
23328 00006F81 30E4                <1>       xor     ah, ah
23329 00006F83 E88E9CFFFF          <1>       call   int16h
23330                              <1>       ; AL = ASCII Code of the Character
23331 00006F88 3C0D                <1>       cmp    al, 13
23332 00006F8A 0F84B7010000        <1>       je     loc_set_date_retn
23333 00006F90 3C1B                <1>       cmp    al, 27
23334 00006F92 0F84AF010000        <1>       je     loc_set_date_retn
23335 00006F98 A2[D7030100]        <1>       mov    [Day], al
23336 00006F9D 3C30                <1>       cmp    al, '0'
23337 00006F9F 0F82AD010000        <1>       jb     loc_set_date_stc_0
23338 00006FA5 3C33                <1>       cmp    al, '3'
23339 00006FA7 0F87A5010000        <1>       ja     loc_set_date_stc_0
23340                              <1>       ; 13/05/2016
23341                              <1>       ;mov   bx, 7 ; attribute/color (bl)
23342                              <1>                    ; video page 0 (bh)
23343 00006FAD B307                <1>       mov    bl, 7
23344 00006FAF E8FEACFFFF          <1>       call   _write_tty
23345                              <1> loc_enter_day_2:
23346 00006FB4 30E4                <1>       xor     ah, ah
23347 00006FB6 E85B9CFFFF          <1>       call   int16h
23348                              <1>       ; AL = ASCII Code of the Character
23349 00006FBB 3C1B                <1>       cmp    al, 27
23350 00006FBD 0F8484010000        <1>         je      loc_set_date_retn
23351 00006FC3 A2[D8030100]        <1>       mov    [Day+1], al
23352 00006FC8 3C30                <1>       cmp    al, '0'
```

```
23353 00006FCA 0F828C010000        <1>         jb      loc_set_date_stc_1
23354 00006FD0 3C39                <1>       cmp    al, '9'
23355 00006FD2 0F8784010000        <1>         ja      loc_set_date_stc_1
23356 00006FD8 803D[D7030100]33    <1>       cmp    byte [Day], '3'
23357 00006FDF 7208                <1>       jb     short pass_set_day_31
23358 00006FE1 3C31                <1>       cmp    al, '1'
23359 00006FE3 0F8773010000        <1>         ja      loc_set_date_stc_1
23360                              <1> pass_set_day_31:
23361                              <1>       ; 13/05/2016
23362                              <1>       ;mov   bx, 7 ; attribute/color (bl)
23363                              <1>                    ; video page 0 (bh)
23364 00006FE9 B307                <1>       mov    bl, 7
23365 00006FEB E8C2ACFFFF          <1>       call   _write_tty
23366                              <1> loc_enter_separator_1:
23367 00006FF0 28E4                <1>       sub     ah, ah ; 0
23368 00006FF2 E81F9CFFFF          <1>       call   int16h
23369                              <1>       ; AL = ASCII Code of the Character
23370 00006FF7 3C1B                <1>       cmp    al, 27
23371 00006FF9 0F8448010000        <1>         je      loc_set_date_retn
23372 00006FFF 3C2D                <1>       cmp    al, '-'
23373 00007001 7408                <1>       je     short pass_set_date_separator_1
23374 00007003 3C2F                <1>       cmp    al, '/'
23375 00007005 0F856C010000        <1>         jne     loc_set_date_stc_2
23376                              <1> pass_set_date_separator_1:
23377                              <1>       ; 13/05/2016
23378                              <1>       ;mov   bx, 7 ; attribute/color (bl)
23379                              <1>                    ; video page 0 (bh)
23380 0000700B B307                <1>       mov    bl, 7
23381 0000700D E8A0ACFFFF          <1>       call   _write_tty
23382                              <1> loc_enter_month_1:
23383 00007012 30E4                <1>       xor     ah, ah ; 0
23384 00007014 E8FD9BFFFF          <1>       call   int16h
23385                              <1>       ; AL = ASCII Code of the Character
23386 00007019 3C1B                <1>       cmp    al, 27
23387 0000701B 0F8426010000        <1>         je      loc_set_date_retn
23388 00007021 A2[DA030100]        <1>       mov    [Month], al
23389 00007026 3C30                <1>       cmp    al, '0'
23390 00007028 0F8264010000        <1>         jb      loc_set_date_stc_3
23391 0000702E 3C31                <1>       cmp    al, '1'
23392 00007030 0F875C010000        <1>         ja      loc_set_date_stc_3
23393                              <1>       ; 13/05/2016
23394                              <1>       ;mov   bx, 7 ; attribute/color (bl)
23395                              <1>                    ; video page 0 (bh)
23396 00007036 B307                <1>       mov    bl, 7
23397 00007038 E875ACFFFF          <1>       call   _write_tty
23398                              <1> loc_enter_month_2:
23399 0000703D 30E4                <1>       xor     ah, ah
23400 0000703F E8D29BFFFF          <1>       call   int16h
23401                              <1>       ; AL = ASCII Code of the Character
23402 00007044 3C1B                <1>       cmp    al, 27
23403 00007046 0F84FB000000        <1>         je      loc_set_date_retn
23404 0000704C A2[DB030100]        <1>       mov    [Month+1], al
23405 00007051 3C30                <1>       cmp    al, '0'
23406 00007053 0F8254010000        <1>         jb      loc_set_date_stc_4
23407 00007059 3C39                <1>       cmp    al, '9'
23408 0000705B 0F874C010000        <1>         ja      loc_set_date_stc_4
23409 00007061 803D[DA030100]31    <1>       cmp    byte [Month], '1'
23410 00007068 7208                <1>       jb     short pass_set_month_12
23411 0000706A 3C32                <1>       cmp    al, '2'
23412 0000706C 0F873B010000        <1>         ja      loc_set_date_stc_4
23413                              <1> pass_set_month_12:
23414                              <1>       ; 13/05/2016
23415                              <1>       ;mov   bx, 7 ; attribute/color (bl)
23416                              <1>                    ; video page 0 (bh)
23417 00007072 B307                <1>       mov    bl, 7
23418 00007074 E839ACFFFF          <1>       call   _write_tty
23419                              <1> loc_enter_separator_2:
23420 00007079 28E4                <1>       sub     ah, ah
23421 0000707B E8969BFFFF          <1>       call   int16h
23422                              <1>       ; AL = ASCII Code of the Character
23423 00007080 3C1B                <1>       cmp    al, 27
23424 00007082 0F84BF000000        <1>         je      loc_set_date_retn
23425 00007088 3C2D                <1>       cmp    al, '-'
23426 0000708A 7408                <1>       je     short pass_set_date_separator_2
23427 0000708C 3C2F                <1>       cmp    al, '/'
23428 0000708E 0F8534010000        <1>         jne     loc_set_date_stc_5
23429                              <1> pass_set_date_separator_2:
23430                              <1>       ; 13/05/2016
23431                              <1>       ;mov   bx, 7 ; attribute/color (bl)
23432                              <1>                    ; video page 0 (bh)
23433 00007094 B307                <1>       mov    bl, 7
23434 00007096 E817ACFFFF          <1>       call   _write_tty
23435                              <1> loc_enter_year_1:
23436 0000709B 30E4                <1>       xor    ah, ah
23437 0000709D E8749BFFFF          <1>       call   int16h
23438                              <1>       ; AL = ASCII Code of the Character
23439 000070A2 3C1B                <1>       cmp    al, 27
23440 000070A4 0F849D000000        <1>         je      loc_set_date_retn
23441 000070AA A2[DF030100]        <1>       mov    [Year], al
23442 000070AF 3C30                <1>       cmp    al, '0'
23443 000070B1 0F822C010000        <1>         jb      loc_set_date_stc_6
23444 000070B7 3C39                <1>       cmp    al, '9'
23445 000070B9 0F8724010000        <1>         ja      loc_set_date_stc_6
23446                              <1>       ; 13/05/2016
23447                              <1>       ;mov   bx, 7 ; attribute/color (bl)
23448                              <1>                    ; video page 0 (bh)
23449 000070BF B307                <1>       mov    bl, 7
23450 000070C1 E8ECABFFFF          <1>       call   _write_tty
23451                              <1> loc_enter_year_2:
23452 000070C6 30E4                <1>       xor    ah, ah
23453 000070C8 E8499BFFFF          <1>       call   int16h
23454                              <1>       ; AL = ASCII Code of the Character
```

```
23455 000070CD 3C1B              <1>       cmp    al, 27
23456 000070CF 7476              <1>       je     short loc_set_date_retn
23457 000070D1 A2[E0030100]      <1>       mov    byte [Year+1], al
23458 000070D6 3C30              <1>       cmp    al, '0'
23459 000070D8 0F8220010000      <1>         jb      loc_set_date_stc_7
23460 000070DE 3C39              <1>       cmp    al, '9'
23461 000070E0 0F8718010000      <1>         ja      loc_set_date_stc_7
23462                            <1>       ; 13/05/2016
23463                            <1>       ;mov   bx, 7 ; attribute/color (bl)
23464                            <1>                    ; video page 0 (bh)
23465 000070E6 B307              <1>       mov    bl, 7
23466 000070E8 E8C5ABFFFF        <1>       call   _write_tty
23467                            <1> loc_set_date_get_lchar_again:
23468 000070ED 28E4              <1>       sub    ah, ah ; 0
23469 000070EF E8229BFFFF        <1>       call   int16h
23470                            <1>       ; AL = ASCII Code of the Character
23471 000070F4 3C0D              <1>       cmp    al, 13 ; ENTER key
23472 000070F6 7412              <1>       je     short loc_set_date_progress
23473 000070F8 3C1B              <1>       cmp    al, 27 ; ESC key
23474 000070FA 744B              <1>       je     short loc_set_date_retn
23475                            <1>       ;
23476 000070FC E82A010000        <1>       call   check_for_backspace
23477 00007101 75EA              <1>       jne    short loc_set_date_get_lchar_again
23478                            <1>
23479                            <1> loc_set_date_bs_8:
23480 00007103 E811010000        <1>       call   write_backspace
23481 00007108 EBBC              <1>       jmp    short loc_enter_year_2
23482                            <1>
23483                            <1> loc_set_date_progress:
23484                            <1>       ; Get Current Date
23485                            <1>       ;mov   ah, 04h
23486                            <1>       ;call  int1Ah
23487 0000710A E84FE9FFFF        <1>       call   RTC_40 ; GET RTC DATE
23488                            <1>       ; CH = century (in BCD)
23489                            <1>
23490 0000710F 66A1[DF030100]    <1>       mov    ax, [Year]
23491 00007115 662D3030          <1>       sub    ax, '00'
23492 00007119 C0E004            <1>       shl    al, 4 ; * 16
23493 0000711C 88C1              <1>       mov    cl, al
23494 0000711E 00E1              <1>       add    cl, ah
23495 00007120 66A1[DA030100]    <1>       mov    ax, [Month]
23496 00007126 662D3030          <1>       sub    ax, '00'
23497 0000712A C0E004            <1>       shl    al, 4 ; * 16
23498 0000712D 88C6              <1>       mov    dh, al
23499 0000712F 00E6              <1>       add    dh, ah
23500 00007131 66A1[D7030100]    <1>       mov    ax, [Day]
23501 00007137 662D3030          <1>       sub    ax, '00'
23502 0000713B C0E004            <1>       shl    al, 4 ; * 16
23503 0000713E 88C2              <1>       mov    dl, al
23504 00007140 00E2              <1>       add    dl, ah
23505                            <1>
23506                            <1>       ;mov   ah, 05h
23507                            <1>       ;call  int1Ah
23508 00007142 E844E9FFFF        <1>       call   RTC_50 ; SET RTC DATE
23509                            <1>
23510                            <1> loc_set_date_retn:
23511 00007147 BE[030F0100]      <1>       mov    esi, nextline
23512 0000714C E80CF2FFFF        <1>       call   print_msg
23513 00007151 C3                <1>       retn
23514                            <1>
23515                            <1> loc_set_date_stc_0:
23516                            <1>       ;xor   bh, bh ; video page 0
23517 00007152 E83BACFFFF        <1>       call   beeper ; BEEP !
23518 00007157 E925FEFFFF        <1>         jmp     loc_enter_day_1
23519                            <1> loc_set_date_stc_1:
23520 0000715C E8CA000000        <1>       call   check_for_backspace
23521 00007161 740A              <1>       je     short loc_set_date_bs_1
23522                            <1>       ;xor   bh, bh ; video page 0
23523 00007163 E82AACFFFF        <1>       call   beeper ; BEEP !
23524 00007168 E947FEFFFF        <1>         jmp     loc_enter_day_2
23525                            <1> loc_set_date_bs_1:
23526 0000716D E8A7000000        <1>       call   write_backspace
23527 00007172 E90AFEFFFF        <1>         jmp     loc_enter_day_1
23528                            <1> loc_set_date_stc_2:
23529 00007177 E8AF000000        <1>       call   check_for_backspace
23530 0000717C 740A              <1>       je     short loc_set_date_bs_2
23531                            <1>       ;xor   bh, bh ; video page 0
23532 0000717E E80FACFFFF        <1>       call   beeper ; BEEP !
23533 00007183 E968FEFFFF        <1>         jmp     loc_enter_separator_1
23534                            <1> loc_set_date_bs_2:
23535 00007188 E88C000000        <1>       call   write_backspace
23536 0000718D E922FEFFFF        <1>         jmp     loc_enter_day_2
23537                            <1> loc_set_date_stc_3:
23538 00007192 E894000000        <1>       call   check_for_backspace
23539 00007197 740A              <1>       je short loc_set_date_bs_3
23540                            <1>       ;xor   bh, bh ; video page 0
23541 00007199 E8F4ABFFFF        <1>       call   beeper ; BEEP !
23542 0000719E E96FFEFFFF        <1>         jmp     loc_enter_month_1
23543                            <1> loc_set_date_bs_3:
23544 000071A3 E871000000        <1>       call   write_backspace
23545 000071A8 E943FEFFFF        <1>         jmp     loc_enter_separator_1
23546                            <1> loc_set_date_stc_4:
23547 000071AD E879000000        <1>       call   check_for_backspace
23548 000071B2 740A              <1>       je     short loc_set_date_bs_4
23549                            <1>       ;xor   bh, bh ; video page 0
23550 000071B4 E8D9ABFFFF        <1>       call   beeper ; BEEP !
23551 000071B9 E97FFEFFFF        <1>         jmp     loc_enter_month_2
23552                            <1> loc_set_date_bs_4:
23553 000071BE E856000000        <1>       call   write_backspace
23554 000071C3 E94AFEFFFF        <1>         jmp     loc_enter_month_1
23555                            <1> loc_set_date_stc_5:
23556 000071C8 E85E000000        <1>       call   check_for_backspace
```

```
23557 000071CD 740A              <1>       je     short loc_set_date_bs_5
23558                            <1>       ;xor   bh, bh ; video page 0
23559 000071CF E8BEABFFFF        <1>       call   beeper ; BEEP !
23560 000071D4 E9A0FEFFFF        <1>         jmp     loc_enter_separator_2
23561                            <1> loc_set_date_bs_5:
23562 000071D9 E83B000000        <1>       call   write_backspace
23563 000071DE E95AFEFFFF        <1>         jmp     loc_enter_month_2
23564                            <1> loc_set_date_stc_6:
23565 000071E3 E843000000        <1>       call   check_for_backspace
23566 000071E8 740A              <1>         je      short  loc_set_date_bs_6
23567                            <1>       ;xor   bh, bh ; video page 0
23568 000071EA E8A3ABFFFF        <1>       call   beeper ; BEEP !
23569 000071EF E9A7FEFFFF        <1>         jmp     loc_enter_year_1
23570                            <1> loc_set_date_bs_6:
23571 000071F4 E820000000        <1>       call   write_backspace
23572 000071F9 E97BFEFFFF        <1>         jmp     loc_enter_separator_2
23573                            <1> loc_set_date_stc_7:
23574 000071FE E828000000        <1>       call   check_for_backspace
23575 00007203 740A              <1>       je     short loc_set_date_bs_7
23576                            <1>       ;xor   bh, bh ; video page 0
23577 00007205 E888ABFFFF        <1>       call   beeper ; BEEP !
23578 0000720A E9B7FEFFFF        <1>         jmp     loc_enter_year_2
23579                            <1> loc_set_date_bs_7:
23580 0000720F E805000000        <1>       call   write_backspace
23581 00007214 E982FEFFFF        <1>         jmp     loc_enter_year_1
23582                            <1>
23583                            <1> write_backspace:
23584                            <1>       ; 18/01/2016 (TRDOS 386 = TRDOS v2.0)
23585 00007219 B008              <1>       mov    al, 08h ; BACKSPACE
23586                            <1>       ; 13/05/2016
23587 0000721B 66BB0700          <1>       mov    bx, 7 ; bl = attribute/color
23588                            <1>                    ; bh = video page = 0
23589 0000721F E88EAAFFFF        <1>       call   _write_tty
23590 00007224 B020              <1>       mov    al, 20h ; BLANK/SPACE char
23591                            <1>       ;mov   bx, 7 ; attribute/color
23592                            <1>       ;call  _write_c_current
23593                            <1>       ;retn
23594 00007226 E90EAAFFFF        <1>       jmp    _write_c_current
23595                            <1>
23596                            <1> check_for_backspace:
23597                            <1>       ; 18/01/2016 (TRDOS 386 = TRDOS v2.0)
23598 0000722B 663D080E          <1>       cmp    ax, 0E08h
23599 0000722F 7410              <1>       je     short cfbs_retn
23600 00007231 663DE04B          <1>       cmp    ax, 4BE0h
23601 00007235 740A              <1>       je     short cfbs_retn
23602 00007237 663D004B          <1>       cmp    ax, 4B00h
23603 0000723B 7404              <1>       je     short cfbs_retn
23604 0000723D 663DE053          <1>       cmp    ax, 53E0h
23605                            <1> cfbs_retn:
23606 00007241 C3                <1>       retn
23607                            <1>
23608                            <1> show_time:
23609                            <1>       ; 18/01/2016 (TRDOS 386 = TRDOS v2.0)
23610                            <1>         ; 2004-2005
23611                            <1>
23612                            <1>       ;mov   ah, 02h
23613                            <1>       ;call  int1Ah
23614 00007242 E8A6E7FFFF        <1>       call   RTC_20 ; GET RTC TIME
23615                            <1>
23616 00007247 88E8              <1>       mov    al, ch
23617 00007249 E8BF99FFFF        <1>       call   bcd_to_ascii
23618 0000724E 66A3[05040100]    <1>       mov    [Hour], ax
23619                            <1>
23620 00007254 88C8              <1>       mov    al, cl
23621 00007256 E8B299FFFF        <1>       call   bcd_to_ascii
23622 0000725B 66A3[08040100]    <1>       mov    [Minute], ax
23623                            <1>
23624 00007261 88F0              <1>       mov    al, dh
23625 00007263 E8A599FFFF        <1>       call   bcd_to_ascii
23626 00007268 66A3[0B040100]    <1>       mov    [Second], ax
23627                            <1>
23628 0000726E BE[F5030100]      <1>       mov    esi, Msg_Show_Time
23629 00007273 E8E5F0FFFF        <1>       call   print_msg
23630 00007278 C3                <1>       retn
23631                            <1>
23632                            <1> set_time:
23633                            <1>       ; 13/05/2016
23634                            <1>       ; 18/01/2016 (TRDOS 386 = TRDOS v2.0)
23635                            <1>         ; 2004-2005
23636                            <1>
23637 00007279 BE[E4030100]      <1>       mov    esi, Msg_Enter_Time
23638 0000727E E8DAF0FFFF        <1>       call   print_msg
23639                            <1>
23640                            <1> loc_enter_hour_1:
23641 00007283 30E4              <1>       xor     ah, ah
23642 00007285 E88C99FFFF        <1>       call   int16h
23643                            <1>       ; AL = ASCII Code of the Character
23644 0000728A 3C0D              <1>       cmp    al, 13 ; ENTER key
23645 0000728C 0F84AE010000      <1>         je      loc_set_time_retn
23646 00007292 3C1B              <1>       cmp    al, 27 ; ESC key
23647 00007294 0F84A6010000      <1>         je      loc_set_time_retn
23648 0000729A A2[05040100]      <1>       mov    [Hour], al
23649 0000729F 3C30              <1>       cmp    al, '0'
23650 000072A1 0F82A4010000      <1>         jb      loc_set_time_stc_0
23651 000072A7 3C32              <1>       cmp    al, '2'
23652 000072A9 0F879C010000      <1>         ja      loc_set_time_stc_0
23653                            <1>       ; 13/05/2016
23654                            <1>       ;mov   bx, 7 ; attribute/color (bl)
23655                            <1>                    ; video page 0 (bh)
23656 000072AF B307              <1>       mov    bl, 7
23657 000072B1 E8FCA9FFFF        <1>       call   _write_tty
23658                            <1> loc_enter_hour_2:
```

```
23659 000072B6 30E4                <1>       xor     ah, ah
23660 000072B8 E85999FFFF          <1>       call   int16h
23661                              <1>       ; AL = ASCII Code of the Character
23662 000072BD 3C1B                <1>       cmp    al, 27
23663 000072BF 0F847B010000        <1>         je      loc_set_time_retn
23664 000072C5 A2[06040100]        <1>       mov    [Hour+1], al
23665 000072CA 3C30                <1>       cmp    al, '0'
23666 000072CC 0F8283010000        <1>         jb      loc_set_time_stc_1
23667 000072D2 3C39                <1>       cmp    al, '9'
23668 000072D4 0F877B010000        <1>       ja     loc_set_time_stc_1
23669 000072DA 803D[05040100]32    <1>         cmp     byte [Hour], '2'
23670 000072E1 7208                <1>       jb     short pass_set_time_24
23671 000072E3 3C34                <1>       cmp    al, '4'
23672 000072E5 0F876A010000        <1>         ja      loc_set_time_stc_1
23673                              <1> pass_set_time_24:
23674                              <1>       ; 13/05/2016
23675                              <1>       ;mov   bx, 7 ; attribute/color (bl)
23676                              <1>                    ; video page 0 (bh)
23677 000072EB B307                <1>       mov    bl, 7
23678 000072ED E8C0A9FFFF          <1>       call   _write_tty
23679                              <1> loc_enter_time_separator_1:
23680 000072F2 28E4                <1>       sub    ah, ah ; 0
23681 000072F4 E81D99FFFF          <1>       call   int16h
23682                              <1>       ; AL = ASCII Code of the Character
23683 000072F9 3C1B                <1>       cmp    al, 27
23684 000072FB 0F843F010000        <1>         je      loc_set_time_retn
23685 00007301 3C3A                <1>       cmp    al, ':'
23686 00007303 0F8567010000        <1>         jne     loc_set_time_stc_2
23687                              <1>       ; 13/05/2016
23688                              <1>       ;mov   bx, 7 ; attribute/color (bl)
23689                              <1>                    ; video page 0 (bh)
23690 00007309 B307                <1>       mov    bl, 7
23691 0000730B E8A2A9FFFF          <1>       call   _write_tty
23692                              <1> loc_enter_minute_1:
23693 00007310 30E4                <1>       xor     ah, ah
23694 00007312 E8FF98FFFF          <1>       call   int16h
23695                              <1>       ; AL = ASCII Code of the Character
23696 00007317 3C1B                <1>       cmp    al, 27
23697 00007319 0F8421010000        <1>         je      loc_set_time_retn
23698 0000731F A2[08040100]        <1>       mov    [Minute], al
23699 00007324 3C30                <1>       cmp    al, '0'
23700 00007326 0F825F010000        <1>         jb      loc_set_time_stc_3
23701 0000732C 3C35                <1>       cmp    al, '5'
23702 0000732E 0F8757010000        <1>         ja      loc_set_time_stc_3
23703                              <1>       ; 13/05/2016
23704                              <1>       ;mov   bx, 7 ; attribute/color (bl)
23705                              <1>                    ; video page 0 (bh)
23706 00007334 B307                <1>       mov    bl, 7
23707 00007336 E877A9FFFF          <1>       call   _write_tty
23708                              <1> loc_enter_minute_2:
23709 0000733B 30E4                <1>       xor     ah, ah
23710 0000733D E8D498FFFF          <1>       call   int16h
23711                              <1>       ; AL = ASCII Code of the Character
23712 00007342 3C1B                <1>       cmp    al, 27
23713 00007344 0F84F6000000        <1>         je      loc_set_time_retn
23714 0000734A A2[09040100]        <1>       mov    [Minute+1], al
23715 0000734F 3C30                <1>       cmp    al, '0'
23716 00007351 0F824F010000        <1>         jb      loc_set_time_stc_4
23717 00007357 3C39                <1>       cmp    al, '9'
23718 00007359 0F8747010000        <1>         ja      loc_set_time_stc_4
23719                              <1>       ; 13/05/2016
23720                              <1>       ;mov   bx, 7 ; attribute/color (bl)
23721                              <1>                    ; video page 0 (bh)
23722 0000735F B307                <1>       mov    bl, 7
23723 00007361 E84CA9FFFF          <1>       call   _write_tty
23724                              <1> loc_enter_time_separator_2:
23725 00007366 66C705[0B040100]30- <1>       mov    word [Second], 3030h
23726 0000736E 30                  <1>
23727 0000736F 28E4                <1>       sub     ah, ah
23728 00007371 E8A098FFFF          <1>       call   int16h
23729                              <1>       ; AL = ASCII Code of the Character
23730 00007376 3C0D                <1>       cmp    al, 13
23731 00007378 0F8485000000        <1>         je      loc_set_time_progress
23732 0000737E 3C1B                <1>       cmp    al, 27
23733 00007380 0F84BA000000        <1>         je      loc_set_time_retn
23734 00007386 3C3A                <1>       cmp    al, ':'
23735 00007388 0F8533010000        <1>         jne     loc_set_time_stc_5
23736                              <1>       ; 13/05/2016
23737                              <1>       ;mov   bx, 7 ; attribute/color (bl)
23738                              <1>                    ; video page 0 (bh)
23739 0000738E B307                <1>       mov    bl, 7
23740 00007390 E81DA9FFFF          <1>       call   _write_tty
23741                              <1> loc_enter_second_1:
23742 00007395 30E4                <1>       xor     ah, ah
23743 00007397 E87A98FFFF          <1>       call   int16h
23744                              <1>       ; AL = ASCII Code of the Character
23745 0000739C 3C0D                <1>       cmp    al, 13
23746 0000739E 7463                <1>       je     short loc_set_time_progress
23747 000073A0 3C1B                <1>       cmp    al, 27
23748 000073A2 0F8498000000        <1>         je      loc_set_time_retn
23749 000073A8 A2[0B040100]        <1>       mov    [Second], al
23750 000073AD 3C30                <1>       cmp    al, '0'
23751 000073AF 0F8227010000        <1>         jb      loc_set_time_stc_6
23752 000073B5 3C35                <1>       cmp    al, '5'
23753 000073B7 0F871F010000        <1>         ja      loc_set_time_stc_6
23754                              <1>       ; 13/05/2016
23755                              <1>       ;mov   bx, 7 ; attribute/color (bl)
23756                              <1>                    ; video page 0 (bh)
23757 000073BD B307                <1>       mov    bl, 7
23758 000073BF E8EEA8FFFF          <1>       call   _write_tty
23759                              <1> loc_enter_second_2:
23760 000073C4 30E4                <1>       xor     ah, ah
```

```
23761 000073C6 E84B98FFFF          <1>       call   int16h
23762                              <1>       ; AL = ASCII Code of the Character
23763 000073CB 3C1B                <1>       cmp    al, 27
23764 000073CD 7471                <1>       je     short loc_set_time_retn
23765 000073CF 3C30                <1>       cmp    al, '0'
23766 000073D1 0F8229010000        <1>         jb      loc_set_time_stc_7
23767 000073D7 3C39                <1>       cmp    al, '9'
23768 000073D9 0F8721010000        <1>         ja      loc_set_time_stc_7
23769                              <1>       ; 13/05/2016
23770                              <1>       ;mov   bx, 7 ; attribute/color (bl)
23771                              <1>                    ; video page 0 (bh)
23772 000073DF B307                <1>       mov    bl, 7
23773 000073E1 E8CCA8FFFF          <1>       call   _write_tty
23774                              <1> loc_set_time_get_lchar_again:
23775 000073E6 28E4                <1>       sub    ah, ah ; 0
23776 000073E8 E82998FFFF          <1>       call   int16h
23777                              <1>       ; AL = ASCII Code of the Character
23778 000073ED 3C0D                <1>       cmp    al, 13
23779 000073EF 7412                <1>       je     short loc_set_time_progress
23780 000073F1 3C1B                <1>       cmp    al, 27
23781 000073F3 744B                <1>       je     short loc_set_time_retn
23782                              <1>       ;
23783 000073F5 E831FEFFFF          <1>       call   check_for_backspace
23784 000073FA 75EA                <1>       jne    short loc_set_time_get_lchar_again
23785                              <1>
23786                              <1> loc_set_time_bs_8:
23787 000073FC E818FEFFFF          <1>       call   write_backspace
23788 00007401 EBC1                <1>       jmp    short loc_enter_second_2
23789                              <1>
23790                              <1> loc_set_time_progress:
23791                              <1>       ; Get Current Time
23792                              <1>       ;mov   ah, 02h
23793                              <1>       ;call  int1Ah
23794 00007403 E8E5E5FFFF          <1>       call   RTC_20 ; GET RTC TIME
23795                              <1>       ;DL = Daylight Savings Enable option (0-1)
23796                              <1>
23797 00007408 66A1[05040100]      <1>       mov    ax, [Hour]
23798 0000740E 662D3030            <1>       sub    ax, '00'
23799 00007412 C0E004              <1>       shl    al, 4 ; * 16
23800 00007415 88C5                <1>       mov    ch, al
23801 00007417 00E5                <1>       add    ch, ah
23802 00007419 66A1[08040100]      <1>       mov    ax, [Minute]
23803 0000741F 662D3030            <1>       sub    ax, '00'
23804 00007423 C0E004              <1>       shl    al, 4 ; * 16
23805 00007426 88C1                <1>       mov    cl, al
23806 00007428 00E1                <1>       add    cl, ah
23807 0000742A 66A1[0B040100]      <1>       mov    ax, [Second]
23808 00007430 662D3030            <1>       sub    ax, '00'
23809 00007434 C0E004              <1>       shl    al, 4 ; * 16
23810 00007437 88C6                <1>       mov    dh, al
23811 00007439 00E6                <1>       add    dh, ah
23812                              <1>
23813                              <1>       ;mov   ah, 03h
23814                              <1>       ;call  int1Ah
23815 0000743B E8DCE5FFFF          <1>       call   RTC_30 ; SET RTC TIME
23816                              <1>
23817                              <1> loc_set_time_retn:
23818 00007440 BE[030F0100]        <1>       mov    esi, nextline
23819 00007445 E813EFFFFF          <1>       call   print_msg
23820 0000744A C3                  <1>       retn
23821                              <1>
23822                              <1> loc_set_time_stc_0:
23823                              <1>       ;xor   bh, bh ; video page 0
23824 0000744B E842A9FFFF          <1>       call   beeper ; BEEP !
23825 00007450 E92EFEFFFF          <1>         jmp     loc_enter_hour_1
23826                              <1> loc_set_time_stc_1:
23827 00007455 E8D1FDFFFF          <1>       call   check_for_backspace
23828 0000745A 740A                <1>       je     short loc_set_time_bs_1
23829                              <1>       ;xor   bh, bh ; video page 0
23830 0000745C E831A9FFFF          <1>       call   beeper ; BEEP !
23831 00007461 E950FEFFFF          <1>         jmp     loc_enter_hour_2
23832                              <1> loc_set_time_bs_1:
23833 00007466 E8AEFDFFFF          <1>       call   write_backspace
23834 0000746B E913FEFFFF          <1>         jmp     loc_enter_hour_1
23835                              <1> loc_set_time_stc_2:
23836 00007470 E8B6FDFFFF          <1>       call   check_for_backspace
23837 00007475 740A                <1>       je     short loc_set_time_bs_2
23838                              <1>       ;xor   bh, bh ; video page 0
23839 00007477 E816A9FFFF          <1>       call   beeper ; BEEP !
23840 0000747C E971FEFFFF          <1>         jmp     loc_enter_time_separator_1
23841                              <1> loc_set_time_bs_2:
23842 00007481 E893FDFFFF          <1>       call   write_backspace
23843 00007486 E92BFEFFFF          <1>         jmp     loc_enter_hour_2
23844                              <1> loc_set_time_stc_3:
23845 0000748B E89BFDFFFF          <1>       call   check_for_backspace
23846 00007490 740A                <1>       je     short loc_set_time_bs_3
23847                              <1>       ;xor   bh, bh ; video page 0
23848 00007492 E8FBA8FFFF          <1>       call   beeper ; BEEP !6
23849 00007497 E974FEFFFF          <1>         jmp     loc_enter_minute_1
23850                              <1> loc_set_time_bs_3:
23851 0000749C E878FDFFFF          <1>       call   write_backspace
23852 000074A1 E94CFEFFFF          <1>         jmp     loc_enter_time_separator_1
23853                              <1> loc_set_time_stc_4:
23854 000074A6 E880FDFFFF          <1>       call   check_for_backspace
23855 000074AB 740A                <1>       je     short loc_set_time_bs_4
23856                              <1>       ;xor   bh, bh ; video page 0
23857 000074AD E8E0A8FFFF          <1>       call   beeper ; BEEP !
23858 000074B2 E984FEFFFF          <1>         jmp     loc_enter_minute_2
23859                              <1> loc_set_time_bs_4:
23860 000074B7 E85DFDFFFF          <1>       call   write_backspace
23861 000074BC E94FFEFFFF          <1>         jmp     loc_enter_minute_1
23862                              <1> loc_set_time_stc_5:
```

```
23863 000074C1 E865FDFFFF          <1>       call   check_for_backspace
23864 000074C6 740A                <1>       je     short loc_set_time_bs_5
23865                              <1>       ;xor   bh, bh ; video page 0
23866 000074C8 E8C5A8FFFF          <1>       call   beeper ; BEEP !
23867 000074CD E994FEFFFF          <1>         jmp     loc_enter_time_separator_2
23868                              <1> loc_set_time_bs_5:
23869 000074D2 E842FDFFFF          <1>       call   write_backspace
23870 000074D7 E95FFEFFFF          <1>         jmp     loc_enter_minute_2
23871                              <1> loc_set_time_stc_6:
23872 000074DC E84AFDFFFF          <1>       call   check_for_backspace
23873 000074E1 7413                <1>       je     short loc_set_time_bs_6
23874                              <1>       ;xor   bh, bh ; video page 0
23875 000074E3 E8AAA8FFFF          <1>       call   beeper ; BEEP !
23876 000074E8 66C705[0B040100]30- <1>       mov    word [Second], 3030h
23877 000074F0 30                  <1>
23878 000074F1 E99FFEFFFF          <1>         jmp     loc_enter_second_1
23879                              <1> loc_set_time_bs_6:
23880 000074F6 E81EFDFFFF          <1>       call   write_backspace
23881 000074FB E966FEFFFF          <1>         jmp     loc_enter_time_separator_2
23882                              <1> loc_set_time_stc_7:
23883 00007500 E826FDFFFF          <1>       call   check_for_backspace
23884 00007505 740A                <1>       je     short loc_set_time_bs_7
23885                              <1>       ;xor   bh, bh ; video page 0
23886 00007507 E886A8FFFF          <1>       call   beeper ; BEEP !
23887 0000750C E9B3FEFFFF          <1>         jmp     loc_enter_second_2
23888                              <1> loc_set_time_bs_7:
23889 00007511 E803FDFFFF          <1>       call   write_backspace
23890 00007516 E97AFEFFFF          <1>         jmp     loc_enter_second_1
23891                              <1>
23892                              <1> print_volume_info:
23893                              <1>       ; 01/03/2016
23894                              <1>       ; 08/02/2016
23895                              <1>       ; 06/02/2016
23896                              <1>       ; 04/02/2016
23897                              <1>       ; 18/01/2016 (TRDOS 386 = TRDOS v2.0)
23898                              <1>       ; 25/10/2009
23899                              <1>       ;
23900                              <1>       ; "Volume Serial No: "
23901                              <1>       ;
23902                              <1>       ; INPUT  : AL = DOS Drive Number
23903                              <1>       ; OUTPUT : AH = FS Type
23904                              <1>       ;          AL = DOS Drive Name
23905                              <1>       ; CF = 0 -> OK
23906                              <1>       ; CF = 1 -> Drive not ready
23907                              <1>
23908 0000751B 88C4                <1>       mov    ah, al
23909 0000751D 28C0                <1>       sub    al, al
23910 0000751F 0FB7F0              <1>       movzx  esi, ax
23911 00007522 81C600010900        <1>       add    esi, Logical_DOSDisks
23912 00007528 8A06                <1>       mov    al, [esi]
23913 0000752A 3C41                <1>       cmp    al, 'A'
23914 0000752C 7304                <1>       jnb    short loc_pvi_set_vol_name
23915 0000752E 8A6604              <1>       mov    ah, [esi+LD_FSType]
23916 00007531 C3                  <1>       retn
23917                              <1>
23918                              <1> loc_pvi_set_vol_name:
23919 00007532 A2[3F040100]        <1>       mov    [Vol_Drv_Name], al
23920 00007537 56                  <1>       push   esi
23921 00007538 E858010000          <1>       call   move_volume_name_and_serial_no ;;;
23922 0000753D 7302                <1>       jnc    short loc_pvi_mvn_ok
23923 0000753F 5E                  <1>       pop    esi
23924 00007540 C3                  <1>       retn
23925                              <1>
23926                              <1> loc_pvi_mvn_ok:
23927 00007541 8B3424              <1>       mov    esi, [esp]
23928 00007544 807E04A1            <1>       cmp    byte [esi+LD_FSType], 0A1h
23929 00007548 7509                <1>       jne    short loc_pvi_fat_vol_size
23930 0000754A 8B4670              <1>       mov    eax, [esi+LD_FS_VolumeSize]
23931 0000754D 0FB75E11            <1>       movzx  ebx, word [esi+LD_FS_BytesPerSec]
23932 00007551 EB07                <1>       jmp    short loc_vol_size_mul32
23933                              <1> loc_pvi_fat_vol_size:
23934 00007553 8B4670              <1>       mov    eax, [esi+LD_TotalSectors]
23935 00007556 0FB75E11            <1>       movzx  ebx, word [esi+LD_BPB+BPB_BytsPerSec]
23936                              <1> loc_vol_size_mul32:
23937 0000755A F7E3                <1>       mul    ebx
23938 0000755C 09D2                <1>       or     edx, edx
23939 0000755E 7507                <1>       jnz    short loc_vol_size_in_kbytes
23940                              <1> loc_vol_size_in_bytes:
23941 00007560 B9[1D040100]        <1>       mov    ecx, VolSize_Bytes
23942 00007565 EB0D                <1>       jmp    short loc_write_vol_size_str
23943                              <1> loc_vol_size_in_kbytes:
23944 00007567 66BB0004            <1>       mov    bx, 1024
23945 0000756B F7F3                <1>       div    ebx
23946 0000756D B9[10040100]        <1>       mov    ecx, VolSize_KiloBytes
23947 00007572 31D2                <1>       xor    edx, edx ; 0
23948                              <1> loc_write_vol_size_str:
23949 00007574 890D[C7560100]      <1>       mov    [VolSize_Unit1], ecx
23950                              <1>       ;
23951 0000757A BF[DD560100]        <1>       mov    edi, Vol_Tot_Sec_Str_End
23952                              <1>         ;mov byte [edi], 0
23953 0000757F B90A000000          <1>       mov    ecx, 10
23954                              <1> loc_write_vol_size_chr:
23955 00007584 F7F1                <1>       div    ecx
23956 00007586 80C230              <1>       add    dl, '0'
23957 00007589 4F                  <1>       dec    edi
23958 0000758A 8817                <1>       mov    [edi], dl
23959 0000758C 85C0                <1>       test   eax, eax
23960 0000758E 7404                <1>       jz     short loc_write_vol_size_str_ok
23961 00007590 28D2                <1>       sub    dl, dl ; 0
23962 00007592 EBF0                <1>       jmp    short loc_write_vol_size_chr
23963                              <1>
23964                              <1> loc_write_vol_size_str_ok:
```

```
23965 00007594 893D[CF560100]      <1>       mov    [Vol_Tot_Sec_Str_Start], edi
23966                              <1>       ;
23967 0000759A BF[28040100]        <1>       mov    edi, Vol_FS_Name
23968 0000759F 8A4E03              <1>       mov    cl, [esi+LD_FATType]
23969 000075A2 20C9                <1>       and    cl, cl ; 0 ?
23970 000075A4 7515                <1>       jnz    short loc_write_vol_FAT_str_1
23971 000075A6 66C7075452          <1>       mov    word [edi], 'TR'
23972 000075AB C7470420465331      <1>       mov    dword [edi+4], ' FS1'
23973                              <1>       ;movzx ebx, word [esi+LD_FS_BytesPerSec]
23974 000075B2 668B5E11            <1>       mov    bx, [esi+LD_FS_BytesPerSec]
23975 000075B6 8B4674              <1>       mov    eax, [esi+LD_FS_FreeSectors]
23976 000075B9 EB36                <1>       jmp    short loc_vol_freespace_mul32
23977                              <1>
23978                              <1> loc_write_vol_FAT_str_1:
23979 000075BB 66B83332            <1>       mov    ax, '32' ; FAT32
23980 000075BF 80F902              <1>       cmp    cl, 2 ; [esi+LD_FATType]
23981 000075C2 7708                <1>       ja     short loc_write_vol_FAT_str_2
23982 000075C4 66B83132            <1>       mov    ax, '12' ; FAT12
23983 000075C8 7202                <1>       jb     short loc_write_vol_FAT_str_2
23984 000075CA B436                <1>       mov    ah, '6'  ; FAT16
23985                              <1> loc_write_vol_FAT_str_2:
23986 000075CC C70746415420        <1>       mov    dword [edi], 'FAT '
23987 000075D2 66894704            <1>       mov    word [edi+4], ax
23988                              <1>       ;
23989                              <1>       ;movzx ebx, word [esi+LD_BPB+BPB_BytsPerSec]
23990 000075D6 668B5E11            <1>       mov    bx, [esi+LD_BPB+BPB_BytsPerSec]
23991 000075DA 8B4674              <1>       mov    eax, [esi+LD_FreeSectors]
23992                              <1>
23993                              <1> loc_vol_freespace_recalc0:
23994                              <1>       ; 01/03/2016
23995 000075DD 83F8FF              <1>       cmp    eax, 0FFFFFFFFh
23996 000075E0 720F                <1>       jb     short loc_vol_freespace_mul32
23997                              <1>       ;inc   eax ; 0
23998 000075E2 20C9                <1>       and    cl, cl ; byte [esi+LD_FATType]
23999 000075E4 740B                <1>       jz     short loc_vol_freespace_mul32
24000 000075E6 53                  <1>       push   ebx
24001 000075E7 66BB00FF            <1>       mov    bx, 0FF00h ; recalculate free sectors
24002 000075EB E8A9490000          <1>       call   calculate_fat_freespace
24003 000075F0 5B                  <1>       pop    ebx
24004                              <1>
24005                              <1> loc_vol_freespace_mul32:
24006 000075F1 F7E3                <1>       mul    ebx
24007 000075F3 09D2                <1>       or     edx, edx
24008 000075F5 7507                <1>       jnz    short loc_vol_fspace_in_kbytes
24009                              <1> loc_vol_fspace_in_bytes:
24010 000075F7 B9[1D040100]        <1>       mov    ecx, VolSize_Bytes
24011 000075FC EB0D                <1>       jmp    short loc_write_vol_fspace_str
24012                              <1> loc_vol_fspace_in_kbytes:
24013 000075FE 66BB0004            <1>       mov    bx, 1024
24014 00007602 F7F3                <1>       div    ebx
24015 00007604 B9[10040100]        <1>       mov    ecx, VolSize_KiloBytes
24016 00007609 31D2                <1>       xor    edx, edx ; 0
24017                              <1> loc_write_vol_fspace_str:
24018 0000760B 890D[CB560100]      <1>       mov    [VolSize_Unit2], ecx
24019                              <1>       ;
24020 00007611 BF[ED560100]        <1>       mov    edi, Vol_Free_Sectors_Str_End
24021                              <1>         ;mov byte [edi], 0
24022 00007616 B90A000000          <1>       mov    ecx, 10
24023                              <1> loc_write_vol_fspace_chr:
24024 0000761B F7F1                <1>       div    ecx
24025 0000761D 80C230              <1>       add    dl, '0'
24026 00007620 4F                  <1>       dec    edi
24027 00007621 8817                <1>       mov    [edi], dl
24028 00007623 85C0                <1>       test   eax, eax
24029 00007625 7404                <1>       jz     short loc_write_vol_fspace_str_ok
24030 00007627 28D2                <1>       sub    dl, dl ; 0
24031 00007629 EBF0                <1>       jmp    short loc_write_vol_fspace_chr
24032                              <1>
24033                              <1> loc_write_vol_fspace_str_ok:
24034 0000762B 893D[DF560100]      <1>       mov    [Vol_Free_Sectors_Str_Start], edi
24035                              <1>       ;
24036 00007631 BE[26040100]        <1>       mov    esi, Volume_in_drive
24037 00007636 E822EDFFFF          <1>       call   print_msg
24038 0000763B BE[66040100]        <1>       mov    esi, Vol_Name
24039 00007640 E818EDFFFF          <1>       call   print_msg
24040 00007645 BE[030F0100]        <1>       mov    esi, nextline
24041 0000764A E80EEDFFFF          <1>       call   print_msg
24042                              <1>       ;
24043 0000764F BE[C7040100]        <1>       mov    esi, Vol_Total_Sector_Header
24044 00007654 E804EDFFFF          <1>       call   print_msg
24045 00007659 8B35[CF560100]      <1>       mov    esi, [Vol_Tot_Sec_Str_Start]
24046 0000765F E8F9ECFFFF          <1>       call   print_msg
24047 00007664 8B35[C7560100]      <1>       mov    esi, [VolSize_Unit1]
24048 0000766A E8EEECFFFF          <1>       call   print_msg
24049                              <1>       ;
24050 0000766F BE[D8040100]        <1>       mov    esi, Vol_Free_Sectors_Header
24051 00007674 E8E4ECFFFF          <1>       call   print_msg
24052 00007679 8B35[DF560100]      <1>       mov    esi, [Vol_Free_Sectors_Str_Start]
24053 0000767F E8D9ECFFFF          <1>       call   print_msg
24054 00007684 8B35[CB560100]      <1>       mov    esi, [VolSize_Unit2]
24055 0000768A E8CEECFFFF          <1>       call   print_msg
24056                              <1>       ;
24057 0000768F 5E                  <1>       pop    esi
24058                              <1>
24059                              <1>       ;mov   ah, [esi+LD_FSType]
24060                              <1>       ;mov   al, [esi+LD_FATType]
24061 00007690 668B4603            <1>       mov    ax, [esi+LD_FATType]
24062                              <1>
24063 00007694 C3                  <1>       retn
24064                              <1>
24065                              <1> move_volume_name_and_serial_no:
24066                              <1>       ; 08/02/2016  (TRDOS 386 = TRDOS v2.0)
```

```
24067                                  <1>       ; this routine will be called by
24068                                  <1>       ; "print_volume_info" and "print_directory"
24069                                  <1>       ; INPUT ->
24070                                  <1>       ;     ESI = Logical DOS drv descripton table address
24071                                  <1>       ; OUTPUT ->
24072                                  <1>       ;     *Volume name will be moved to text area
24073                                  <1>       ;     *Volume serial number will be converted to
24074                                  <1>       ;      text and will be moved to text area
24075                                  <1>       ;   cf = 1 -> invalid/unknown dos drive
24076                                  <1>       ;   cf = 0 -> ecx = 0
24077                                  <1>       ;
24078                                  <1>       ; (eax, edx, ecx, esi, edi will be changed)
24079                                  <1>
24080 00007695 BF[66040100]           <1>       mov    edi, Vol_Name
24081                                  <1>
24082                                  <1>       ;mov   ah, [esi+LD_FSType]
24083                                  <1>       ;mov   al, [esi+LD_FATType]
24084 0000769A 668B4603              <1>       mov    ax, [esi+LD_FATType]
24085 0000769E 80FCA1                <1>       cmp    ah, 0A1h
24086 000076A1 7418                  <1>       je     short mvn_2
24087 000076A3 08E4                  <1>       or     ah, ah
24088 000076A5 7404                  <1>       jz     short mvn_0
24089 000076A7 08C0                  <1>       or     al, al
24090 000076A9 7504                  <1>       jnz    short mvn_1
24091                                  <1> mvn_0:
24092 000076AB 8A06                  <1>       mov    al, [esi]
24093 000076AD F9                    <1>       stc
24094 000076AE C3                    <1>       retn
24095                                  <1> mvn_1:
24096 000076AF 3C02                  <1>       cmp    al, 2
24097 000076B1 7717                  <1>       ja     short mvn_3
24098                                  <1>       ;or    al, al
24099                                  <1>       ;jz    short mvn_2
24100 000076B3 8B462D                <1>       mov    eax, [esi+LD_BPB+VolumeID]
24101 000076B6 83C631                <1>       add    esi, LD_BPB+VolumeLabel
24102 000076B9 EB15                  <1>       jmp    short mvn_4
24103                                  <1> mvn_2:
24104 000076BB 8B4628                <1>       mov    eax, [esi+LD_FS_VolumeSerial]
24105 000076BE 83C62C                <1>       add    esi, LD_FS_VolumeName
24106 000076C1 B910000000            <1>       mov    ecx, 16
24107 000076C6 F3A5                  <1>       rep    movsd
24108 000076C8 EB10                  <1>       jmp    short mvn_5
24109                                  <1> mvn_3:
24110 000076CA 8B4649                <1>       mov    eax, [esi+LD_BPB+FAT32_VolID]
24111 000076CD 83C64D                <1>       add    esi, LD_BPB+FAT32_VolLab
24112                                  <1> mvn_4:
24113 000076D0 B90B000000            <1>       mov    ecx, 11
24114 000076D5 F3A4                  <1>       rep    movsb
24115 000076D7 C60700                <1>       mov    byte [edi], 0
24116                                  <1> mvn_5:
24117                                  <1>       ;mov   [Current_VolSerial], eax
24118 000076DA E82ABCFFFF            <1>       call   dwordtohex
24119 000076DF 8915[BB040100]        <1>       mov    [Vol_Serial1], edx
24120 000076E5 A3[C0040100]          <1>       mov    [Vol_Serial2], eax
24121                                  <1>       ; ecx = 0
24122 000076EA C3                    <1>       retn
24123                                  <1>
24124                                  <1> get_volume_serial_number:
24125                                  <1>       ; 19/01/2016 (TRDOS 386 = TRDOS v2.0)
24126                                  <1>       ; 08/08/2010
24127                                  <1>       ;
24128                                  <1>       ; INPUT -> DL = Logical DOS Drive number
24129                                  <1>       ; OUTPUT -> EAX = Volume serial number
24130                                  <1>       ;          BL= FAT Type
24131                                  <1>       ;          BH = Logical DOS drv Number (DL input)
24132                                  <1>       ; cf = 1 -> Drive not ready
24133                                  <1>
24134 000076EB 31DB                  <1>       xor    ebx, ebx
24135 000076ED 88D7                  <1>       mov    bh, dl
24136 000076EF 3815[97020100]        <1>       cmp    [Last_DOS_DiskNo], dl
24137 000076F5 7304                  <1>       jnb    short loc_gvsn_start
24138                                  <1> loc_gvsn_stc_retn:
24139 000076F7 31C0                  <1>       xor    eax, eax
24140 000076F9 F9                    <1>       stc
24141 000076FA C3                    <1>         retn
24142                                  <1> loc_gvsn_start:
24143 000076FB 56                    <1>       push   esi
24144 000076FC BE00010900            <1>       mov    esi, Logical_DOSDisks
24145 00007701 01DE                  <1>       add    esi, ebx
24146 00007703 8A5E03                <1>       mov    bl, [esi+LD_FATType]
24147 00007706 20DB                  <1>       and    bl, bl
24148 00007708 740F                  <1>       jz     short loc_gvsn_fs
24149 0000770A 80FB02                <1>       cmp    bl, 2
24150 0000770D 7705                  <1>       ja     short loc_gvsn_fat32
24151                                  <1> loc_gvsn_fat:
24152 0000770F 83C62D                <1>       add    esi, LD_BPB + VolumeID
24153 00007712 EB0E                  <1>       jmp    short loc_gvsn_return
24154                                  <1> loc_gvsn_fat32:
24155 00007714 83C649                <1>       add    esi, LD_BPB + FAT32_VolID
24156 00007717 EB09                  <1>       jmp    short loc_gvsn_return
24157                                  <1> loc_gvsn_fs:
24158 00007719 807E04A1              <1>       cmp    byte [esi+LD_FSType], 0A1h
24159 0000771D 75D8                  <1>       jne    short loc_gvsn_stc_retn
24160 0000771F 83C628                <1>       add    esi, LD_FS_VolumeSerial
24161                                  <1> loc_gvsn_return:
24162 00007722 8B06                  <1>       mov    eax, [esi]
24163 00007724 5E                    <1>       pop    esi
24164 00007725 C3                    <1>       retn
24165                                  <1>
24166                                  <1> ; CMD_INTR.ASM [ TRDOS Command Interpreter Procedure ]
24167                                  <1> ; 09/11/2011
24168                                  <1> ; 29/01/2005
```

```
24169                                  <1>
24170                                  <1> command_interpreter:
24171                                  <1>       ; 16/10/2016
24172                                  <1>       ; 12/10/2016
24173                                  <1>       ; 13/05/2016
24174                                  <1>       ; 07/05/2016
24175                                  <1>       ; 04/03/2016
24176                                  <1>       ; 04/02/2016
24177                                  <1>       ; 03/02/2016
24178                                  <1>       ; 30/01/2016
24179                                  <1>       ; 29/01/2016 (TRDOS 386 = TRDOS 2.0)
24180                                  <1>       ; 15/09/2011
24181                                  <1>       ; 29/01/2005
24182                                  <1>
24183                                  <1>       ; Input: ecx = command word length (CL)
24184                                  <1>       ;      CommandBuffer = Command string offset
24185                                  <1>
24186 00007726 C605[80570100]00        <1>       mov    byte [Program_Exit],0
24187 0000772D 80F904                  <1>       cmp    cl, 4
24188 00007730 0F87AE020000            <1>         ja      c_6
24189 00007736 0F822E010000            <1>         jb      c_2
24190                                  <1> c_4:
24191                                  <1>
24192                                  <1> cmp_cmd_exit:
24193 0000773C BF[05030100]            <1>       mov    edi, Cmd_Exit
24194 00007741 E8BA030000              <1>       call   cmp_cmd
24195 00007746 7208                    <1>       jc     short cmp_cmd_date
24196                                  <1>
24197 00007748 C605[80570100]01        <1>         mov     byte [Program_Exit], 1
24198 0000774F C3                      <1>         retn
24199                                  <1>
24200                                  <1> cmp_cmd_date:
24201 00007750 B104                    <1>       mov    cl, 4
24202 00007752 BF[21030100]            <1>       mov    edi, Cmd_Date
24203 00007757 E8A4030000              <1>       call   cmp_cmd
24204 0000775C 720B                    <1>         jc   short cmp_cmd_time
24205                                  <1>
24206 0000775E E8D0F7FFFF              <1>       call   show_date
24207 00007763 E80FF8FFFF              <1>       call   set_date
24208 00007768 C3                      <1>       retn
24209                                  <1>
24210                                  <1> cmp_cmd_time:
24211 00007769 B104                    <1>       mov    cl, 4
24212 0000776B BF[26030100]            <1>       mov    edi, Cmd_Time
24213 00007770 E88B030000              <1>       call   cmp_cmd
24214 00007775 720B                    <1>       jc     short cmp_cmd_show
24215                                  <1>
24216 00007777 E8C6FAFFFF              <1>       call   show_time
24217 0000777C E8F8FAFFFF              <1>       call   set_time
24218 00007781 C3                      <1>       retn
24219                                  <1>
24220                                  <1> cmp_cmd_show:
24221 00007782 B104                    <1>       mov    cl, 4
24222 00007784 BF[37030100]            <1>       mov    edi, Cmd_Show
24223 00007789 E872030000              <1>       call   cmp_cmd
24224 0000778E 0F83EF090000            <1>         jnc     show_file
24225                                  <1>
24226                                  <1> cmp_cmd_echo:
24227 00007794 B104                    <1>       mov    cl, 4
24228 00007796 BF[73030100]            <1>       mov    edi, Cmd_Echo
24229 0000779B E860030000              <1>       call   cmp_cmd
24230 000077A0 721B                    <1>       jc     short cmp_cmd_copy
24231                                  <1>
24232                                  <1>       ; 14/04/2016
24233 000077A2 56                      <1>       push   esi
24234                                  <1> cmd_echo_asciiz:
24235 000077A3 46                      <1>         inc  esi
24236 000077A4 8A06                    <1>       mov    al, [esi]
24237 000077A6 3C20                    <1>       cmp    al, 20h
24238 000077A8 73F9                    <1>       jnb    short cmd_echo_asciiz
24239 000077AA C60600                  <1>       mov    byte [esi], 0
24240 000077AD 5E                      <1>       pop    esi
24241 000077AE E8AAEBFFFF              <1>       call   print_msg
24242 000077B3 BE[4C0F0100]            <1>       mov    esi, NextLine
24243                                  <1>       ;call  print_msg
24244                                  <1>       ;retn
24245 000077B8 E9A0EBFFFF              <1>       jmp    print_msg
24246                                  <1>
24247                                  <1> cmp_cmd_copy:
24248 000077BD B104                    <1>       mov    cl, 4
24249 000077BF BF[5A030100]            <1>       mov    edi, Cmd_Copy
24250 000077C4 E837030000              <1>       call   cmp_cmd
24251 000077C9 0F8305180000            <1>       jnc    copy_file
24252                                  <1>
24253                                  <1> cmp_cmd_move:
24254 000077CF B104                    <1>       mov    cl, 4
24255 000077D1 BF[5F030100]            <1>       mov    edi, Cmd_Move
24256 000077D6 E825030000              <1>       call   cmp_cmd
24257 000077DB 0F8399160000            <1>       jnc    move_file
24258                                  <1>
24259                                  <1> cmp_cmd_path:
24260 000077E1 B104                    <1>       mov    cl, 4
24261 000077E3 BF[64030100]            <1>       mov    edi, Cmd_Path
24262 000077E8 E813030000              <1>       call   cmp_cmd
24263 000077ED 0F83391A0000            <1>       jnc    set_get_path
24264                                  <1>
24265                                  <1> cmp_cmd_beep:
24266 000077F3 B104                    <1>       mov    cl, 4
24267 000077F5 BF[91030100]            <1>       mov    edi, Cmd_Beep
24268 000077FA E801030000              <1>       call   cmp_cmd
24269 000077FF 720B                    <1>       jc     short cmp_cmd_find
24270                                  <1>       ; 13/05/2016
```

```
24271 00007801 8A3D[F64D0100]      <1>       mov    bh, [ptty] ; [ACTIVE_PAGE]
24272 00007807 E986A5FFFF          <1>       jmp    beeper
24273                              <1>
24274                              <1> cmp_cmd_find:
24275 0000780C B104                <1>       mov    cl, 4
24276 0000780E BF[6E030100]        <1>       mov    edi, Cmd_Find
24277 00007813 E8E8020000          <1>       call   cmp_cmd
24278 00007818 0F82C5020000        <1>         jc     cmp_cmd_external
24279                              <1>
24280                              <1>       ;call  find_and_list_files
24281 0000781E E9EA220000          <1>       jmp    find_and_list_files
24282                              <1>       ;retn
24283                              <1>
24284                              <1> c_1:
24285 00007823 AD                  <1>       lodsd
24286                              <1> cmp_cmd_help:
24287 00007824 3C3F                <1>       cmp    al, '?'
24288 00007826 751D                <1>         jne    short cmp_cmd_remark
24289                              <1>
24290 00007828 BE[F7020100]        <1>       mov    esi, Command_List
24291                              <1> cmd_help_next_w:
24292 0000782D E82BEBFFFF          <1>       call   print_msg
24293                              <1>
24294 00007832 803E20              <1>       cmp    byte [esi], 20h ; 0
24295 00007835 7232                <1>       jb     short cmd_help_retn
24296                              <1>
24297 00007837 56                  <1>       push   esi
24298 00007838 BE[030F0100]        <1>       mov    esi, nextline
24299 0000783D E81BEBFFFF          <1>       call   print_msg
24300 00007842 5E                  <1>       pop    esi
24301 00007843 EBE8                <1>       jmp    short cmd_help_next_w
24302                              <1>
24303                              <1> cmp_cmd_remark:
24304 00007845 3C2A                <1>       cmp    al, '*'
24305 00007847 0F8596020000        <1>         jne    cmp_cmd_external
24306 0000784D 46                  <1>       inc    esi
24307 0000784E BF[F04E0100]        <1>       mov    edi, Remark
24308 00007853 8A06                <1>       mov    al, [esi]
24309 00007855 3C20                <1>       cmp    al, 20h
24310 00007857 7707                <1>       ja     short cmd_remark_write
24311 00007859 89FE                <1>       mov    esi, edi ; Remark
24312 0000785B E9FDEAFFFF          <1>       jmp    print_msg
24313                              <1>
24314                              <1> cmd_remark_write:
24315 00007860 AA                  <1>       stosb
24316 00007861 AC                  <1>       lodsb
24317 00007862 3C20                <1>       cmp    al, 20h
24318 00007864 73FA                <1>       jnb    short cmd_remark_write
24319 00007866 C60700              <1>       mov    byte [edi], 0
24320                              <1>
24321                              <1> cmd_help_retn:
24322                              <1> cmd_remark_retn:
24323                              <1> cd_retn:
24324 00007869 C3                  <1>       retn
24325                              <1>
24326                              <1> c_2:
24327 0000786A 80F902              <1>       cmp    cl, 2
24328 0000786D 0F87B1000000        <1>         ja     c_3
24329 00007873 BE[3E4F0100]        <1>       mov    esi, CommandBuffer
24330 00007878 72A9                <1>       jb     short c_1
24331                              <1>
24332                              <1> cmp_cmd_cd:
24333 0000787A 66AD                <1>       lodsw
24334 0000787C 663D4344            <1>       cmp    ax, 'CD'
24335 00007880 7553                <1>       jne    short cmp_cmd_drive
24336 00007882 46                  <1>         inc  esi
24337                              <1> cd_0:
24338 00007883 668B06              <1>       mov    ax, [esi]
24339 00007886 3C20                <1>       cmp    al, 20h
24340 00007888 76DF                <1>       jna    short cd_retn
24341                              <1>       ; 10/02/2016
24342 0000788A 80FC3A              <1>       cmp    ah, ':'
24343 0000788D 7504                <1>       jne    short cd_1
24344 0000788F 46                  <1>       inc    esi
24345 00007890 46                  <1>       inc    esi
24346 00007891 EB4B                <1>       jmp    short cd_2
24347                              <1>
24348                              <1> cd_1: ; change current directory
24349                              <1>       ; 29/11/2009
24350                              <1>       ; AH = CDh   ; to separate 'CD' command from others
24351                              <1>                    ; for restoring current directory
24352                              <1>                    ; 0CDh sign is for saving cdir into
24353                              <1>                    ; DOS drv description table cdir area
24354                              <1>
24355 00007893 B4CD                <1>       mov    ah, 0CDh ; mov byte [CD_COMMAND], 0CDh
24356                              <1>
24357 00007895 E858230000          <1>       call   change_current_directory
24358 0000789A 0F8372220000        <1>         jnc    change_prompt_dir_string
24359                              <1>
24360                              <1> cd_error_messages:
24361 000078A0 3C03                <1>       cmp    al, 3
24362 000078A2 740C                <1>       je     short cd_path_not_found
24363                              <1>       ; 16/10/2016 (15h -> 15)
24364 000078A4 3C0F                <1>       cmp    al, 15 ; drive not ready error
24365 000078A6 745B                <1>       je     short cd_drive_not_ready
24366 000078A8 3C11                <1>       cmp    al, 17 ; read error
24367 000078AA 7457                <1>       je     short cd_drive_not_ready
24368 000078AC 3C13                <1>       cmp    al, 19 ; ; Bad directory/path name
24369 000078AE 7468                <1>       je     short cd_command_failed
24370                              <1>
24371                              <1> cd_path_not_found:
24372 000078B0 6650                <1>       push   ax
```

```
24373 000078B2 BE[9A050100]        <1>       mov    esi, Msg_Dir_Not_Found
24374 000078B7 E8A1EAFFFF          <1>       call   print_msg
24375 000078BC 6658                <1>       pop    ax
24376 000078BE 3A25[8C4E0100]      <1>       cmp    ah, [Current_Dir_Level]
24377 000078C4 0F8348220000        <1>         jnb     change_prompt_dir_string
24378 000078CA 8825[8C4E0100]      <1>       mov    [Current_Dir_Level], ah
24379 000078D0 E93D220000          <1>         jmp     change_prompt_dir_string
24380                              <1>
24381                              <1> cmp_cmd_drive: ; change current drive
24382                              <1>       ; C:, D:, E: etc.
24383 000078D5 80FC3A              <1>       cmp    ah, ':'
24384 000078D8 0F8505020000        <1>         jne     cmp_cmd_external
24385                              <1>
24386                              <1> cd_2: ; 'CD C:', 'CD D:' ...
24387 000078DE 803E20              <1>       cmp    byte [esi], 20h
24388 000078E1 0F8706020000        <1>         ja      loc_cmd_failed
24389                              <1>
24390 000078E7 24DF                <1>       and    al, 0DFh
24391 000078E9 2C41                <1>       sub    al, 'A'
24392 000078EB 0F82FC010000        <1>         jc      loc_cmd_failed
24393                              <1>
24394 000078F1 3A05[97020100]      <1>         cmp     al, [Last_DOS_DiskNo]
24395 000078F7 770A                <1>         ja   short cd_drive_not_ready
24396                              <1>
24397 000078F9 88C2                <1>       mov    dl, al
24398 000078FB E85DF3FFFF          <1>       call   change_current_drive
24399 00007900 7201                <1>       jc     short cd_drive_not_ready
24400 00007902 C3                  <1>       retn
24401                              <1>
24402                              <1> cd_drive_not_ready:
24403 00007903 BE[57050100]        <1>       mov    esi, Msg_Not_Ready_Read_Err
24404 00007908 E850EAFFFF          <1>       call   print_msg
24405                              <1>
24406                              <1> cd_fail_drive_restart:
24407 0000790D 8A15[8E4E0100]      <1>       mov    dl, [Current_Drv]
24408                              <1>       ;call  change_current_drive
24409 00007913 E945F3FFFF          <1>         jmp     change_current_drive
24410                              <1>       ;retn
24411                              <1>
24412                              <1> cd_command_failed:
24413 00007918 BE[38050100]        <1>       mov    esi, Msg_Bad_Command
24414 0000791D E83BEAFFFF          <1>       call   print_msg
24415 00007922 EBE9                <1>       jmp    short cd_fail_drive_restart
24416                              <1>
24417                              <1> c_3:
24418                              <1> cmp_cmd_dir:
24419 00007924 BF[F7020100]        <1>       mov    edi, Cmd_Dir
24420 00007929 E8D2010000          <1>       call   cmp_cmd
24421 0000792E 0F8371020000        <1>       jnc    print_directory_list
24422                              <1>
24423                              <1> cmp_cmd_cls:
24424 00007934 B103                <1>       mov    cl, 3
24425 00007936 BF[33030100]        <1>       mov    edi, Cmd_Cls
24426 0000793B E8C0010000          <1>       call   cmp_cmd
24427 00007940 0F832DEAFFFF        <1>         jnc  clear_screen
24428                              <1>
24429                              <1> cmp_cmd_ver:
24430 00007946 B103                <1>       mov    cl, 3
24431 00007948 BF[01030100]        <1>       mov    edi, Cmd_Ver
24432 0000794D E8AE010000          <1>       call   cmp_cmd
24433 00007952 720A                <1>       jc     short cmp_cmd_mem
24434                              <1>
24435 00007954 BE[9F020100]        <1>       mov    esi, mainprog_Version
24436                              <1>       ;call  print_msg
24437 00007959 E9FFE9FFFF          <1>       jmp    print_msg
24438                              <1>       ;retn
24439                              <1>
24440                              <1> cmp_cmd_mem:
24441 0000795E B103                <1>       mov    cl, 3
24442 00007960 BF[69030100]        <1>       mov    edi, Cmd_Mem
24443 00007965 E896010000          <1>       call   cmp_cmd
24444 0000796A 0F83E4B8FFFF        <1>       jnc    memory_info
24445                              <1>
24446                              <1> cmp_cmd_del:
24447 00007970 B103                <1>       mov    cl, 3
24448 00007972 BF[3C030100]        <1>       mov    edi, Cmd_Del
24449 00007977 E884010000          <1>       call   cmp_cmd
24450 0000797C 0F832D0F0000        <1>         jnc     delete_file
24451                              <1>
24452                              <1> cmp_cmd_set:
24453 00007982 B103                <1>       mov    cl, 3
24454 00007984 BF[2F030100]        <1>       mov    edi, Cmd_Set
24455 00007989 E872010000          <1>       call   cmp_cmd
24456 0000798E 0F8310180000        <1>         jnc     set_get_env
24457                              <1>
24458                              <1> cmp_cmd_run:
24459 00007994 B103                <1>       mov    cl, 3
24460 00007996 BF[2B030100]        <1>       mov    edi, Cmd_Run
24461 0000799B E860010000          <1>       call   cmp_cmd
24462                              <1>       ; 07/05/2016
24463 000079A0 0F823D010000        <1>         jc      cmp_cmd_external
24464 000079A6 E9471E0000          <1>       jmp    load_and_execute_file
24465                              <1> c_5:
24466                              <1> cmp_cmd_mkdir:
24467 000079AB BF[54030100]        <1>       mov    edi, Cmd_Mkdir
24468 000079B0 E84B010000          <1>       call   cmp_cmd
24469 000079B5 0F838C0A0000        <1>         jnc     make_directory
24470                              <1>
24471                              <1> cmp_cmd_rmdir:
24472 000079BB B105                <1>       mov    cl, 5
24473 000079BD BF[4E030100]        <1>       mov    edi, Cmd_Rmdir
24474 000079C2 E839010000          <1>       call   cmp_cmd
```

```
24475 000079C7 0F83990B0000        <1>         jnc     delete_directory
24476                              <1>
24477                              <1> cmp_cmd_chdir:
24478 000079CD B105                <1>         mov     cl, 5
24479 000079CF BF[8B030100]        <1>         mov     edi, Cmd_Chdir
24480 000079D4 E827010000          <1>         call    cmp_cmd
24481 000079D9 0F8204010000        <1>           jc      cmp_cmd_external
24482                              <1>
24483 000079DF E99FFEFFFF          <1>         jmp     cd_0
24484                              <1>
24485                              <1> c_6:
24486 000079E4 80F906              <1>         cmp     cl, 6
24487 000079E7 0F87DF000000        <1>           ja      c_8
24488 000079ED 72BC                <1>         jb      short c_5
24489                              <1> cmp_cmd_prompt:
24490 000079EF BF[0A030100]        <1>         mov     edi, Cmd_Prompt
24491 000079F4 E807010000          <1>         call    cmp_cmd
24492 000079F9 722E                <1>           jc    short cmp_cmd_volume
24493                              <1> get_prompt_name_fchar:
24494 000079FB AC                  <1>         lodsb
24495 000079FC 3C20                <1>         cmp     al, 20h
24496 000079FE 74FB                <1>         je      short get_prompt_name_fchar
24497 00007A00 7712                <1>         ja      short loc_change_prompt_label
24498 00007A02 BE[EB020100]        <1>         mov     esi, TRDOSPromptLabel
24499 00007A07 C7065452444F        <1>         mov     dword [esi], "TRDO"
24500 00007A0D 66C746045300        <1>                 mov     word [esi+4], "S"
24501                              <1> loc_cmd_prompt_return:
24502 00007A13 C3                  <1>         retn
24503                              <1> loc_change_prompt_label:
24504 00007A14 66B90B00            <1>         mov     cx, 11
24505 00007A18 BF[EB020100]        <1>         mov     edi, TRDOSPromptLabel
24506                              <1> put_char_new_prompt_label:
24507 00007A1D AA                  <1>         stosb
24508 00007A1E AC                  <1>         lodsb
24509 00007A1F 3C20                <1>         cmp     al, 20h
24510 00007A21 7202                <1>         jb      short pass_put_new_prompt_label
24511 00007A23 E2F8                <1>         loop    put_char_new_prompt_label
24512                              <1> pass_put_new_prompt_label:
24513 00007A25 C60700              <1>         mov     byte [edi], 0
24514 00007A28 C3                  <1>         retn
24515                              <1>
24516                              <1> cmp_cmd_volume:
24517 00007A29 B106                <1>         mov     cl, 6
24518 00007A2B BF[11030100]        <1>         mov     edi, Cmd_Volume
24519 00007A30 E8CB000000          <1>         call    cmp_cmd
24520 00007A35 7255                <1>           jc    short cmp_cmd_attrib
24521                              <1>
24522                              <1> cmd_vol1:
24523 00007A37 AC                  <1>         lodsb
24524 00007A38 3C20                <1>         cmp     al, 20h
24525 00007A3A 7707                <1>         ja      short cmd_vol2
24526 00007A3C A0[8E4E0100]        <1>         mov     al, [Current_Drv]
24527 00007A41 EB3D                <1>         jmp     short cmd_vol4
24528                              <1> cmd_vol2:
24529 00007A43 3C41                <1>         cmp     al, 'A'
24530 00007A45 0F82A2000000        <1>           jb      loc_cmd_failed
24531 00007A4B 3C7A                <1>         cmp     al, 'z'
24532 00007A4D 0F879A000000        <1>           ja      loc_cmd_failed
24533 00007A53 3C5A                <1>         cmp     al, 'Z'
24534 00007A55 760A                <1>         jna     short cmd_vol3
24535 00007A57 3C61                <1>         cmp     al, 'a'
24536 00007A59 0F828E000000        <1>           jb      loc_cmd_failed
24537 00007A5F 24DF                <1>         and     al, 0DFh
24538                              <1> cmd_vol3:
24539 00007A61 8A26                <1>         mov     ah, [esi]
24540 00007A63 80FC3A              <1>         cmp     ah, ':'
24541 00007A66 0F8581000000        <1>           jne     loc_cmd_failed
24542 00007A6C 2C41                <1>         sub     al, 'A'
24543 00007A6E 3A05[97020100]      <1>           cmp     al, [Last_DOS_DiskNo]
24544 00007A74 760A                <1>         jna     short cmd_vol4
24545                              <1>
24546 00007A76 BE[57050100]        <1>         mov     esi, Msg_Not_Ready_Read_Err
24547 00007A7B E9DDE8FFFF          <1>         jmp     print_msg
24548                              <1>
24549                              <1> cmd_vol4:
24550 00007A80 E896FAFFFF          <1>         call    print_volume_info
24551 00007A85 0F8278FEFFFF        <1>           jc      cd_drive_not_ready
24552 00007A8B C3                  <1>         retn
24553                              <1>
24554                              <1> cmp_cmd_attrib:
24555 00007A8C B106                <1>         mov     cl, 6
24556 00007A8E BF[40030100]        <1>         mov     edi, Cmd_Attrib
24557 00007A93 E868000000          <1>         call    cmp_cmd
24558 00007A98 0F83310F0000        <1>           jnc     set_file_attributes
24559                              <1>
24560                              <1> cmp_cmd_rename:
24561 00007A9E B106                <1>         mov     cl, 6
24562 00007AA0 BF[47030100]        <1>         mov     edi, Cmd_Rename
24563 00007AA5 E856000000          <1>         call    cmp_cmd
24564 00007AAA 0F8367110000        <1>           jnc     rename_file
24565                              <1>
24566                              <1> cmp_cmd_device:
24567 00007AB0 B106                <1>         mov     cl, 6
24568 00007AB2 BF[7C030100]        <1>         mov     edi, Cmd_Device
24569 00007AB7 E844000000          <1>         call    cmp_cmd
24570 00007ABC 7225                <1>           jc    short cmp_cmd_external
24571                              <1>
24572 00007ABE C3                  <1>         retn
24573                              <1>
24574                              <1> c_7:
24575                              <1> cmp_cmd_devlist:
24576 00007ABF BF[83030100]        <1>         mov     edi, Cmd_DevList
```

```
24577 00007AC4 E837000000          <1>       call   cmp_cmd
24578 00007AC9 7218                <1>         jc   short cmp_cmd_external
24579                              <1>
24580                              <1> loc_cmd_return:
24581 00007ACB C3                  <1>       retn
24582                              <1>
24583                              <1> c_8:
24584 00007ACC 80F908              <1>         cmp  cl, 8
24585 00007ACF 7712                <1>       ja     short cmp_cmd_external
24586 00007AD1 72EC                <1>       jb     short c_7
24587                              <1>
24588                              <1> cmp_cmd_longname:
24589 00007AD3 BF[18030100]        <1>       mov    edi, Cmd_LongName
24590 00007AD8 E823000000          <1>       call   cmp_cmd
24591 00007ADD 0F8342060000        <1>         jnc     get_and_print_longname
24592                              <1>
24593                              <1> cmp_cmd_external:
24594                              <1>       ; 07/05/2016
24595                              <1>       ; 22/04/2016
24596 00007AE3 BE[3E4F0100]        <1>       mov    esi, CommandBuffer
24597 00007AE8 E9051D0000          <1>       jmp    loc_run_check_filename
24598                              <1>
24599                              <1> loc_cmd_failed:
24600 00007AED 803D[3E4F0100]20    <1>       cmp    byte [CommandBuffer], 20h
24601 00007AF4 76D5                <1>       jna    short loc_cmd_return
24602 00007AF6 BE[38050100]        <1>       mov    esi, Msg_Bad_Command
24603                              <1> ;     call   print_msg
24604                              <1> ;loc_cmd_return:
24605                              <1> ;     retn
24606 00007AFB E95DE8FFFF          <1>       jmp    print_msg
24607                              <1>
24608                              <1> cmp_cmd:
24609                              <1>       ; 29/01/2016 (TRDOS 386 = TRDOS v2.0)
24610 00007B00 BE[3E4F0100]        <1>         mov esi, CommandBuffer
24611                              <1>         ; edi = internal command word (ASCIIZ)
24612                              <1>       ; ecx = command length (<=8)
24613                              <1> cmp_cmd_1:
24614 00007B05 AC                  <1>       lodsb
24615 00007B06 AE                  <1>       scasb
24616 00007B07 750D                <1>       jne    short cmp_cmd_3
24617 00007B09 E2FA                <1>       loop   cmp_cmd_1
24618 00007B0B AC                  <1>       lodsb
24619 00007B0C 3C20                <1>       cmp    al, 20h
24620 00007B0E 7703                <1>       ja     short cmp_cmd_2
24621 00007B10 30C0                <1>       xor    al, al
24622                              <1>       ; ZF = 1 -> internal command word matches
24623 00007B12 C3                  <1>       retn
24624                              <1> cmp_cmd_2:
24625                              <1>       ; ZF = 0 (CF = 0) -> external command word
24626 00007B13 58                  <1>       pop    eax ; no return to the caller from here
24627 00007B14 EBCD                <1>       jmp    cmp_cmd_external
24628                              <1> cmp_cmd_3:
24629 00007B16 F9                  <1>       stc
24630                              <1>       ; CF = 1 -> internal command word does not match
24631 00007B17 C3                  <1>       retn
24632                              <1>
24633                              <1> loc_run_cmd_failed:
24634                              <1>       ; 15/03/2016
24635                              <1>       ; 15/02/2016 (TRDOS 386 =  TRDOS v2.0)
24636                              <1>       ; 07/12/2009 (CMD_INTR.ASM)
24637                              <1>       ; 29/11/2009
24638                              <1>
24639 00007B18 E855000000          <1>       call   restore_cdir_after_cmd_fail
24640                              <1>
24641                              <1> loc_run_cmd_failed_cmp_al:
24642                              <1>       ; End of Restore_CDIR code (29/11/2009)
24643                              <1>
24644 00007B1D 3C01                <1>       cmp    al, 1 ; Bad command or file name
24645 00007B1F 74CC                <1>       je     loc_cmd_failed
24646                              <1> loc_run_dir_not_found:
24647 00007B21 3C03                <1>       cmp    al, 3
24648 00007B23 750A                <1>       jne    short loc_run_file_notfound_msg
24649                              <1>       ; Path not found (MS-DOS Error Code = 3)
24650 00007B25 BE[9A050100]        <1>       mov    esi, Msg_Dir_Not_Found
24651 00007B2A E92EE8FFFF          <1>       jmp    print_msg
24652                              <1>
24653                              <1> loc_run_file_notfound_msg:
24654 00007B2F 3C02                <1>       cmp    al, 2 ; File not found
24655 00007B31 750A                <1>       jne    short loc_run_file_drv_read_err
24656                              <1>
24657                              <1> loc_print_file_notfound_msg:
24658 00007B33 BE[B1050100]        <1>         mov     esi, Msg_File_Not_Found
24659                              <1>       ;call  proc_printmsg
24660                              <1>       ;retn
24661 00007B38 E920E8FFFF          <1>       jmp    print_msg
24662                              <1>
24663                              <1> loc_run_file_drv_read_err:
24664                              <1>       ; Err: 17 (Read fault)
24665 00007B3D 3C11                <1>       cmp    al, 17 ; Drive not ready or read error
24666 00007B3F 7404                <1>       je     short loc_run_file_print_drv_read_err
24667                              <1>       ;
24668 00007B41 3C0F                <1>       cmp    al, 15 ; Drive not ready (or read error)
24669 00007B43 750A                <1>       jne    short loc_run_file_toobig
24670                              <1>
24671                              <1> loc_run_file_print_drv_read_err:
24672 00007B45 BE[57050100]        <1>       mov    esi, Msg_Not_Ready_Read_Err
24673 00007B4A E90EE8FFFF          <1>       jmp    print_msg
24674                              <1>
24675                              <1> loc_run_file_toobig:
24676 00007B4F 3C08                <1>       cmp    al, 8 ; Not enough free memory to load&run file
24677 00007B51 750A                <1>       jne    short loc_run_misc_error
24678 00007B53 BE[FC050100]        <1>       mov    esi, Msg_Insufficient_Memory
```

```
24679 00007B58 E900E8FFFF          <1>       jmp    print_msg
24680                              <1>
24681                              <1>       ; 15/03/2016
24682                              <1> print_misc_error_msg:
24683                              <1> loc_run_misc_error:
24684                              <1>       ; AL = Error code
24685 00007B5D E867B7FFFF          <1>       call   bytetohex
24686 00007B62 66A3[30060100]      <1>         mov     [error_code_hex], ax
24687                              <1>
24688 00007B68 BE[13060100]        <1>       mov    esi, Msg_Error_Code
24689                              <1>       ;call  print_msg
24690                              <1>       ;retn
24691                              <1>
24692 00007B6D E9EBE7FFFF          <1>       jmp    print_msg
24693                              <1>
24694                              <1> restore_cdir_after_cmd_fail:
24695                              <1>       ; 15/02/2016 (TRDOS 386 =  TRDOS v2.0)
24696 00007B72 50                  <1>       push   eax
24697 00007B73 8A3D[EE560100]      <1>       mov    bh, [RUN_CDRV] ; it is set at the beginning
24698                              <1>                               ; of the 'run' command.
24699 00007B79 3A3D[8E4E0100]      <1>       cmp    bh, [Current_Drv]
24700 00007B7F 7409                <1>       je     short loc_run_restore_cdir
24701 00007B81 88FA                <1>       mov    dl, bh
24702 00007B83 E8D5F0FFFF          <1>       call   change_current_drive
24703 00007B88 EB19                <1>       jmp    short loc_run_err_pass_restore_cdir
24704                              <1>
24705                              <1> loc_run_restore_cdir:
24706 00007B8A 803D[98020100]00    <1>       cmp    byte [Restore_CDIR], 0
24707 00007B91 7610                <1>       jna    short loc_run_err_pass_restore_cdir
24708 00007B93 30DB                <1>       xor    bl, bl
24709 00007B95 0FB7F3              <1>       movzx  esi, bx
24710 00007B98 81C600010900        <1>       add    esi, Logical_DOSDisks
24711 00007B9E E871F1FFFF          <1>       call   restore_current_directory
24712                              <1>
24713                              <1> loc_run_err_pass_restore_cdir:
24714 00007BA3 58                  <1>       pop    eax
24715 00007BA4 C3                  <1>       retn
24716                              <1>
24717                              <1> print_directory_list:
24718                              <1>       ; 10/02/2016
24719                              <1>       ; 08/02/2016 (TRDOS 386 =  TRDOS v2.0)
24720                              <1>       ; 06/12/2009 ('cmp_cmd_dir')
24721                              <1>       ;
24722 00007BA5 66C705[30580100]00- <1>       mov    word [AttributesMask], 0800h ; ..except volume names..
24723 00007BAD 08                  <1>
24724 00007BAE A0[8E4E0100]        <1>       mov    al, [Current_Drv]
24725 00007BB3 A2[EE560100]        <1>       mov    [RUN_CDRV], al
24726                              <1> get_dfname_fchar:
24727 00007BB8 AC                  <1>       lodsb
24728 00007BB9 3C20                <1>       cmp    al, 20h
24729 00007BBB 74FB                <1>       je     short get_dfname_fchar
24730 00007BBD 0F82A4000000        <1>         jb      loc_print_dir_call_all
24731 00007BC3 3C2D                <1>       cmp    al, '-'
24732 00007BC5 7542                <1>       jne    short loc_print_dir_call_flt
24733                              <1> get_next_attr_char:
24734 00007BC7 AC                  <1>       lodsb
24735 00007BC8 3C20                <1>       cmp    al, 20h
24736 00007BCA 74FB                <1>       je     short get_next_attr_char
24737 00007BCC 0F821BFFFFFF        <1>         jb      loc_cmd_failed
24738 00007BD2 24DF                <1>       and    al, 0DFh
24739 00007BD4 3C44                <1>       cmp    al, 'D' ; directories only ?
24740 00007BD6 7512                <1>       jne    short pass_only_directories
24741 00007BD8 AC                  <1>       lodsb
24742 00007BD9 3C20                <1>       cmp    al, 20h
24743 00007BDB 0F870CFFFFFF        <1>         ja      loc_cmd_failed
24744 00007BE1 800D[30580100]10    <1>       or     byte [AttributesMask], 10h ; ..directory..
24745 00007BE8 EB18                <1>       jmp    short get_dfname_fchar_attr
24746                              <1> pass_only_directories:
24747 00007BEA 3C46                <1>       cmp    al, 'F'      ; files only ?
24748 00007BEC 0F85B0000000        <1>         jne     check_attr_s
24749 00007BF2 AC                  <1>       lodsb
24750 00007BF3 3C20                <1>       cmp    al, 20h
24751 00007BF5 0F87F2FEFFFF        <1>         ja      loc_cmd_failed
24752 00007BFB 800D[31580100]10    <1>       or     byte [AttributesMask+1], 10h ; ..except directories..
24753                              <1> get_dfname_fchar_attr:
24754 00007C02 AC                  <1>       lodsb
24755 00007C03 3C20                <1>       cmp    al, 20h
24756 00007C05 74FB                <1>       je     short get_dfname_fchar_attr
24757 00007C07 725E                <1>       jb     short loc_print_dir_call_all
24758                              <1>
24759                              <1> loc_print_dir_call_flt:
24760 00007C09 4E                  <1>       dec    esi
24761 00007C0A BF[32580100]        <1>       mov    edi, FindFile_Drv
24762 00007C0F E8F2250000          <1>       call   parse_path_name
24763 00007C14 7308                <1>       jnc    short loc_print_dir_change_drv_1
24764 00007C16 3C01                <1>       cmp    al, 1
24765 00007C18 0F87FAFEFFFF        <1>         ja      loc_run_cmd_failed
24766                              <1>
24767                              <1> loc_print_dir_change_drv_1:
24768 00007C1E 8A15[32580100]      <1>       mov    dl, [FindFile_Drv]
24769                              <1> loc_print_dir_change_drv_2:
24770 00007C24 3A15[EE560100]      <1>       cmp    dl, [RUN_CDRV]
24771 00007C2A 740B                <1>       je     short loc_print_dir_change_directory
24772 00007C2C E82CF0FFFF          <1>       call   change_current_drive
24773 00007C31 0F82E1FEFFFF        <1>         jc      loc_run_cmd_failed
24774                              <1> loc_print_dir_change_directory:
24775 00007C37 803D[33580100]20    <1>       cmp    byte [FindFile_Directory], 20h ; 0 or 20h ?
24776 00007C3E 761D                <1>       jna    short pass_print_dir_change_directory
24777                              <1>
24778 00007C40 FE05[98020100]      <1>       inc    byte [Restore_CDIR]
24779 00007C46 BE[33580100]        <1>       mov    esi, FindFile_Directory
24780 00007C4B 30E4                <1>       xor    ah, ah ; CD_COMMAND sign -> 0
```

```
24781 00007C4D E8A01F0000          <1>       call   change_current_directory
24782 00007C52 0F82C0FEFFFF        <1>         jc      loc_run_cmd_failed
24783                              <1>
24784                              <1> loc_print_dir_change_prompt_dir_string:
24785 00007C58 E8B51E0000          <1>       call   change_prompt_dir_string
24786                              <1>
24787                              <1> pass_print_dir_change_directory:
24788 00007C5D BE[74580100]        <1>       mov    esi, FindFile_Name
24789 00007C62 803E20              <1>       cmp    byte [esi], 20h ; ; 0 or 20h ?
24790 00007C65 7706                <1>       ja     short loc_print_dir_call
24791                              <1>
24792                              <1> loc_print_dir_call_all:
24793 00007C67 C7062A2E2A00        <1>       mov    dword [esi], '*.*'
24794                              <1> loc_print_dir_call:
24795 00007C6D E87E000000          <1>       call   print_directory
24796                              <1>
24797 00007C72 8A15[EE560100]      <1>       mov    dl, [RUN_CDRV]  ; it is set at the beginning
24798 00007C78 3A15[8E4E0100]      <1>       cmp    dl, [Current_Drv]
24799 00007C7E 7406                <1>       je     short loc_print_dir_call_restore_cdir_retn
24800 00007C80 E8D8EFFFFF          <1>       call   change_current_drive
24801 00007C85 C3                  <1>       retn
24802                              <1>
24803                              <1> loc_print_dir_call_restore_cdir_retn:
24804 00007C86 803D[98020100]00    <1>       cmp    byte [Restore_CDIR], 0
24805 00007C8D 7610                <1>       jna    short pass_print_dir_call_restore_cdir_retn
24806                              <1>
24807 00007C8F BE00010900          <1>       mov    esi, Logical_DOSDisks
24808 00007C94 31C0                <1>       xor    eax, eax
24809 00007C96 88D4                <1>       mov    ah, dl
24810 00007C98 01C6                <1>       add    esi, eax
24811                              <1>
24812 00007C9A E875F0FFFF          <1>       call   restore_current_directory
24813                              <1>
24814                              <1> pass_print_dir_call_restore_cdir_retn:
24815 00007C9F C3                  <1>       retn
24816                              <1>
24817                              <1> check_attr_s_cap:
24818 00007CA0 24DF                <1>       and    al, 0DFh
24819                              <1> check_attr_s:
24820 00007CA2 3C53                <1>       cmp    al, 'S'
24821 00007CA4 7514                <1>       jne    short pass_attr_s
24822 00007CA6 800D[30580100]04    <1>       or     byte [AttributesMask], 4 ; system
24823 00007CAD AC                  <1>       lodsb
24824 00007CAE 3C20                <1>       cmp    al, 20h
24825 00007CB0 0F844CFFFFFF        <1>         je      get_dfname_fchar_attr
24826 00007CB6 72AF                <1>       jb     short loc_print_dir_call_all
24827 00007CB8 24DF                <1>       and    al, 0DFh
24828                              <1> pass_attr_s:
24829 00007CBA 3C48                <1>       cmp    al, 'H'
24830 00007CBC 7514                <1>       jne    short pass_attr_h
24831 00007CBE 800D[30580100]02    <1>       or     byte [AttributesMask], 2 ; hidden
24832                              <1> pass_attr_shr:
24833 00007CC5 AC                  <1>       lodsb
24834 00007CC6 3C20                <1>       cmp    al, 20h
24835 00007CC8 0F8434FFFFFF        <1>         je      get_dfname_fchar_attr
24836 00007CCE 7297                <1>       jb     short loc_print_dir_call_all
24837 00007CD0 EBCE                <1>       jmp    short check_attr_s_cap
24838                              <1>
24839                              <1> pass_attr_h:
24840 00007CD2 3C52                <1>       cmp    al, 'R'
24841 00007CD4 7509                <1>       jne    short pass_attr_r
24842 00007CD6 800D[30580100]01    <1>       or     byte [AttributesMask], 1 ; read only
24843 00007CDD EBE6                <1>       jmp    short pass_attr_shr
24844                              <1>
24845                              <1> pass_attr_r:
24846 00007CDF 3C41                <1>       cmp    al, 'A'
24847 00007CE1 0F8506FEFFFF        <1>         jne     loc_cmd_failed
24848 00007CE7 800D[30580100]20    <1>       or     byte [AttributesMask], 20h ; archive
24849 00007CEE EBD5                <1>       jmp    short pass_attr_shr
24850                              <1>
24851                              <1> print_directory:
24852                              <1>       ; 13/05/2016
24853                              <1>       ; 11/02/2016
24854                              <1>       ; 10/02/2016
24855                              <1>       ; 08/02/2016 (TRDOS 386 =  TRDOS v2.0)
24856                              <1>       ; 30/10/2010 ('proc_print_directory')
24857                              <1>       ; 19/09/2009
24858                              <1>       ; 2005
24859                              <1>       ; INPUT ->
24860                              <1>       ;      ESI = Asciiz File/Dir Name Address
24861                              <1>
24862 00007CF0 56                  <1>       push   esi
24863                              <1>
24864 00007CF1 29C0                <1>       sub    eax, eax
24865                              <1>
24866 00007CF3 66A3[BC580100]      <1>       mov    word [Dir_Count], ax ; 0
24867 00007CF9 66A3[BA580100]      <1>       mov    word [File_Count], ax ; 0
24868 00007CFF A3[BE580100]        <1>       mov    dword [Total_FSize], eax ; 0
24869                              <1>
24870 00007D04 E86AE6FFFF          <1>       call    clear_screen
24871                              <1>
24872 00007D09 31C9                <1>       xor    ecx, ecx
24873 00007D0B 8A2D[8E4E0100]      <1>       mov     ch, [Current_Drv] ; DirBuff_Drv - 'A'
24874 00007D11 A0[8F4E0100]        <1>       mov     al, [Current_Dir_Drv]
24875 00007D16 A2[55040100]        <1>       mov     [Dir_Drive_Name], al
24876 00007D1B BE00010900          <1>       mov    esi, Logical_DOSDisks
24877 00007D20 01CE                <1>       add    esi, ecx
24878                              <1>
24879 00007D22 E86EF9FFFF          <1>       call   move_volume_name_and_serial_no
24880 00007D27 730C                <1>       jnc    short print_dir_strlen_check
24881                              <1>
24882 00007D29 5E                  <1>       pop    esi
```

```
24883 00007D2A 8A3D[F64D0100]      <1>       mov    bh, [ptty] ; [ACTIVE_PAGE]
24884                              <1>       ;call  beeper
24885                              <1>       ;retn
24886 00007D30 E95DA0FFFF          <1>       jmp    beeper  ; beep ! and return
24887                              <1>
24888                              <1> print_dir_strlen_check:
24889 00007D35 BE[914E0100]        <1>       mov    esi, Current_Dir_Root
24890 00007D3A BF[F2040100]        <1>       mov    edi, Dir_Str_Root
24891                              <1>
24892                              <1>       ;xor   ecx, ecx
24893 00007D3F 8A0D[ED4E0100]      <1>         mov     cl, [Current_Dir_StrLen]
24894 00007D45 FEC1                <1>       inc    cl
24895 00007D47 80F940              <1>       cmp    cl, 64
24896 00007D4A 760D                <1>       jna    short pass_print_dir_strlen_shorting
24897 00007D4C 46                  <1>       inc    esi
24898 00007D4D 01CE                <1>       add    esi, ecx
24899 00007D4F 83EE40              <1>       sub    esi, 64
24900 00007D52 47                  <1>       inc    edi
24901 00007D53 B82E2E2E20          <1>       mov    eax, '... '
24902 00007D58 AB                  <1>       stosd
24903                              <1>
24904                              <1> pass_print_dir_strlen_shorting:
24905 00007D59 F3A4                <1>       rep    movsb
24906                              <1>
24907 00007D5B BE[48040100]        <1>       mov    esi, Dir_Drive_Str
24908 00007D60 E8F8E5FFFF          <1>       call   print_msg
24909                              <1>
24910 00007D65 BE[A7040100]        <1>       mov    esi, Vol_Serial_Header
24911 00007D6A E8EEE5FFFF          <1>       call   print_msg
24912                              <1>
24913 00007D6F BE[E7040100]        <1>       mov    esi, Dir_Str_Header
24914 00007D74 E8E4E5FFFF          <1>       call   print_msg
24915                              <1>
24916 00007D79 BE[010F0100]        <1>       mov    esi, next2line
24917 00007D7E E8DAE5FFFF          <1>       call   print_msg
24918                              <1>
24919                              <1> loc_print_dir_first_file:
24920 00007D83 C605[D1580100]10    <1>       mov    byte [PrintDir_RowCounter], 16
24921 00007D8A 66A1[30580100]      <1>       mov    ax, [AttributesMask]
24922 00007D90 5E                  <1>       pop    esi
24923                              <1>
24924 00007D91 E859020000          <1>       call   find_first_file
24925 00007D96 0F826F010000        <1>         jc      loc_dir_ok
24926                              <1>
24927                              <1> loc_dfname_use_this:
24928                              <1>       ; bl = File Attributes (bh = Long Name Entry Length)
24929 00007D9C F6C310              <1>       test   bl, 10h  ; Is it a directory?
24930 00007D9F 741B                <1>       jz     short loc_not_dir
24931                              <1>
24932 00007DA1 66FF05[BC580100]    <1>       inc    word [Dir_Count]
24933 00007DA8 89F2                <1>       mov    edx, esi     ; FindFile_DirEntry address
24934 00007DAA BE[36060100]        <1>       mov    esi, Type_Dir; '<DIR>     '
24935 00007DAF BF[4D060100]        <1>       mov    edi, Dir_Or_FileSize
24936                              <1>       ; move 10 bytes
24937 00007DB4 A5                  <1>       movsd
24938 00007DB5 A5                  <1>       movsd
24939 00007DB6 66A5                <1>       movsw
24940 00007DB8 89D6                <1>       mov    esi, edx
24941 00007DBA EB36                <1>       jmp     short loc_dir_attribute
24942                              <1>
24943                              <1> loc_not_dir:
24944 00007DBC 66FF05[BA580100]    <1>       inc    word [File_Count]
24945 00007DC3 0105[BE580100]      <1>       add    [Total_FSize], eax
24946                              <1>
24947 00007DC9 B90A000000          <1>       mov    ecx, 10  ; 32 bit divisor
24948 00007DCE 89CF                <1>       mov    edi, ecx
24949 00007DD0 81C7[4D060100]      <1>       add    edi, Dir_Or_FileSize
24950                              <1> loc_dir_rdivide:
24951 00007DD6 29D2                <1>       sub    edx, edx
24952 00007DD8 F7F1                <1>       div    ecx     ; remainder in dl (< 10)
24953 00007DDA 80C230              <1>       add     dl, '0'      ; to make visible (ascii)
24954 00007DDD 4F                  <1>       dec    edi
24955 00007DDE 8817                <1>       mov     [edi], dl
24956 00007DE0 21C0                <1>       and    eax, eax
24957 00007DE2 75F2                <1>       jnz    short loc_dir_rdivide
24958                              <1>
24959                              <1> loc_dir_fill_space:
24960 00007DE4 81FF[4D060100]      <1>       cmp     edi, Dir_Or_FileSize
24961 00007DEA 7606                <1>       jna     short loc_dir_attribute
24962 00007DEC 4F                  <1>       dec     edi
24963 00007DED C60720              <1>       mov     byte [edi], 20h
24964 00007DF0 EBF2                <1>       jmp     short loc_dir_fill_space
24965                              <1>
24966                              <1> loc_dir_attribute:
24967 00007DF2 C705[58060100]2020- <1>       mov    dword [File_Attribute], 20202020h
24968 00007DFA 2020                <1>
24969                              <1>
24970 00007DFC 80FB20              <1>       cmp    bl, 20h  ; Is it an archive file?
24971 00007DFF 7207                <1>       jb     short loc_dir_pass_arch
24972 00007E01 C605[5B060100]41    <1>       mov    byte [File_Attribute+3], 'A'
24973                              <1>
24974                              <1> loc_dir_pass_arch:
24975 00007E08 80E307              <1>       and    bl, 7
24976 00007E0B 7428                <1>       jz     short loc_dir_file_name
24977 00007E0D 88DF                <1>       mov    bh, bl
24978 00007E0F 80E303              <1>       and    bl, 3
24979 00007E12 38DF                <1>       cmp    bh, bl
24980 00007E14 7607                <1>       jna    short loc_dir_pass_s
24981 00007E16 C605[58060100]53    <1>       mov    byte [File_Attribute], 'S'
24982                              <1>
24983                              <1> loc_dir_pass_s:
24984 00007E1D 80E302              <1>       and     bl,2
```

```
24985 00007E20 7407                <1>       jz      short loc_dir_pass_h
24986 00007E22 C605[59060100]48    <1>       mov     byte [File_Attribute+1], 'H'
24987                              <1> loc_dir_pass_h:
24988 00007E29 80E701              <1>       and     bh,1
24989 00007E2C 7407                <1>       jz      short loc_dir_file_name
24990 00007E2E C605[5A060100]52    <1>       mov     byte [File_Attribute+2], 'R'
24991                              <1> loc_dir_file_name:
24992                              <1>       ;mov     bx, [esi+18h] ; Date
24993                              <1>       ;mov     dx, [esi+16h] ; Time
24994 00007E35 8B5E16              <1>       mov    ebx, [esi+16h]
24995 00007E38 89F1                <1>       mov    ecx, esi ; FindFile_DirEntry address
24996 00007E3A BF[40060100]        <1>       mov     edi, File_Name
24997                              <1>       ; move 8 bytes
24998 00007E3F A5                  <1>       movsd
24999 00007E40 A5                  <1>       movsd
25000 00007E41 C60720              <1>       mov    byte [edi], 20h
25001 00007E44 47                  <1>       inc    edi
25002                              <1>       ; move 3 bytes
25003 00007E45 66A5                <1>       movsw
25004 00007E47 A4                  <1>       movsb
25005 00007E48 89CE                <1>       mov    esi, ecx
25006                              <1>
25007                              <1> Dir_Time_start:
25008                              <1>       ;mov   ax, dx       ; Time
25009 00007E4A 6689D8              <1>       mov    ax, bx
25010 00007E4D 66C1E805            <1>       shr    ax, 5        ; shift right 5 times
25011 00007E51 6683E03F            <1>       and    ax, 0000111111b     ; Minute Mask
25012 00007E55 D40A                <1>       aam                 ; Q([AL]/10)->AH
25013                              <1>                           ; R([AL]/10)->AL
25014                              <1>                           ; [AL]+[AH]= Minute as BCD
25015 00007E57 660D3030            <1>       or     ax, '00'     ; Convert to ASCII
25016 00007E5B 86E0                <1>       xchg   ah, al
25017 00007E5D 66A3[6B060100]      <1>       mov    [File_Minute], ax
25018                              <1>
25019                              <1>       ;mov   al, dh
25020 00007E63 88F8                <1>       mov    al, bh
25021 00007E65 C0E803              <1>       shr    al, 3        ; shift right 3 times
25022 00007E68 D40A                <1>       aam                 ; [AL]+[AH]= Hours as BCD
25023 00007E6A 660D3030            <1>       or     ax, '00'
25024 00007E6E 86E0                <1>       xchg   ah, al
25025 00007E70 66A3[68060100]      <1>       mov     [File_Hour], ax
25026                              <1>
25027 00007E76 C1EB10              <1>       shr    ebx, 16             ; BX = Date
25028                              <1>
25029                              <1> Dir_Date_start:
25030 00007E79 6689D8              <1>       mov    ax, bx       ; Date
25031 00007E7C 6683E01F            <1>       and    ax, 00011111b; Day Mask
25032 00007E80 D40A                <1>       aam                 ; Q([AL]/10)->AH
25033                              <1>                           ; R([AL]/10)->AL
25034                              <1>                           ; [AL]+[AH]= Day as BCD
25035 00007E82 660D3030            <1>       or     ax, '00'     ; Convert to ASCII
25036 00007E86 86C4                <1>       xchg   al, ah
25037                              <1>
25038 00007E88 66A3[5D060100]      <1>       mov    [File_Day], ax
25039                              <1>
25040 00007E8E 6689D8              <1>       mov    ax, bx
25041 00007E91 66C1E805            <1>       shr    ax, 5        ; shift right 5 times
25042 00007E95 6683E00F            <1>       and    ax, 00001111b; Month Mask
25043 00007E99 D40A                <1>       aam
25044 00007E9B 660D3030            <1>       or     ax, '00'
25045 00007E9F 86E0                <1>       xchg   ah, al
25046 00007EA1 66A3[60060100]      <1>       mov    [File_Month], ax
25047                              <1>
25048 00007EA7 6689D8              <1>       mov    ax, bx
25049 00007EAA 66C1E809            <1>       shr     ax, 9
25050 00007EAE 6683E07F            <1>       and    ax, 01111111b; Result = Year - 1980
25051 00007EB2 6605BC07            <1>       add    ax, 1980
25052                              <1>
25053 00007EB6 B10A                <1>       mov    cl, 10
25054 00007EB8 F6F1                <1>       div    cl           ; Q -> AL, R -> AH
25055 00007EBA 80CC30              <1>       or     ah, '0'
25056 00007EBD 8825[66060100]      <1>       mov    [File_Year+3], ah
25057 00007EC3 D40A                <1>       aam
25058 00007EC5 86E0                <1>       xchg   ah, al
25059 00007EC7 80CC30              <1>       or     ah, '0'        ; Convert to ASCII
25060 00007ECA 8825[65060100]      <1>       mov    [File_Year+2], ah
25061 00007ED0 D40A                <1>       aam
25062 00007ED2 86C4                <1>       xchg   al, ah
25063 00007ED4 660D3030            <1>       or     ax, '00'
25064 00007ED8 66A3[63060100]      <1>       mov    [File_Year], ax
25065                              <1>
25066                              <1> loc_show_line:
25067 00007EDE 56                  <1>       push   esi
25068 00007EDF BE[40060100]        <1>       mov     esi, File_Name
25069 00007EE4 E874E4FFFF          <1>       call   print_msg
25070 00007EE9 BE[030F0100]        <1>       mov    esi, nextline
25071 00007EEE E86AE4FFFF          <1>       call   print_msg
25072 00007EF3 5E                  <1>       pop    esi
25073                              <1>
25074 00007EF4 FE0D[D1580100]      <1>       dec    byte [PrintDir_RowCounter]
25075 00007EFA 0F84D4000000        <1>         jz      pause_dir_scroll
25076                              <1>
25077                              <1> loc_next_entry:
25078 00007F00 E899010000          <1>       call   find_next_file
25079 00007F05 0F8391FEFFFF        <1>         jnc     loc_dfname_use_this
25080                              <1>
25081                              <1> loc_dir_ok:
25082 00007F0B B90A000000          <1>       mov     ecx, 10
25083 00007F10 66A1[BC580100]      <1>       mov    ax, [Dir_Count]
25084 00007F16 BF[81060100]        <1>       mov    edi, Decimal_Dir_Count
25085 00007F1B 6639C8              <1>       cmp    ax, cx ; 10
25086 00007F1E 7216                <1>       jb     short pass_ddc
```

```
25087 00007F20 47                  <1>       inc    edi
25088 00007F21 6683F864            <1>       cmp    ax, 100
25089 00007F25 720F                <1>       jb     short pass_ddc
25090 00007F27 47                  <1>       inc    edi
25091 00007F28 663DE803            <1>       cmp    ax, 1000
25092 00007F2C 7208                <1>       jb     short pass_ddc
25093 00007F2E 47                  <1>       inc    edi
25094 00007F2F 663D1027            <1>       cmp    ax, 10000
25095 00007F33 7201                <1>       jb     short pass_ddc
25096 00007F35 47                  <1>       inc    edi
25097                              <1> pass_ddc:
25098 00007F36 886F01              <1>       mov     [edi+1], ch ; 0
25099                              <1> loc_ddc_rediv:
25100 00007F39 31D2                <1>       xor     edx, edx
25101 00007F3B 66F7F1              <1>       div     cx   ; 10
25102 00007F3E 80C230              <1>       add     dl, '0'
25103 00007F41 8817                <1>       mov     [edi], dl
25104 00007F43 4F                  <1>       dec     edi
25105 00007F44 6609C0              <1>       or     ax, ax
25106 00007F47 75F0                <1>       jnz    short loc_ddc_rediv
25107                              <1>
25108 00007F49 66A1[BA580100]      <1>       mov     ax, [File_Count]
25109 00007F4F BF[70060100]        <1>       mov     edi, Decimal_File_Count
25110 00007F54 6639C8              <1>       cmp     ax, cx ; 10
25111 00007F57 7216                <1>       jb      short pass_dfc
25112 00007F59 47                  <1>       inc     edi
25113 00007F5A 6683F864            <1>       cmp     ax, 100
25114 00007F5E 720F                <1>       jb      short pass_dfc
25115 00007F60 47                  <1>       inc     edi
25116 00007F61 663DE803            <1>       cmp     ax, 1000
25117 00007F65 7208                <1>       jb      short pass_dfc
25118 00007F67 47                  <1>       inc     edi
25119 00007F68 663D1027            <1>       cmp     ax, 10000
25120 00007F6C 7201                <1>       jb      short pass_dfc
25121 00007F6E 47                  <1>       inc     edi
25122                              <1> pass_dfc:
25123                              <1>       ;mov    cx, 10
25124 00007F6F 886F01              <1>       mov     [edi+1], ch ; 00
25125                              <1> loc_dfc_rediv:
25126                              <1>       ;xor   dx, dx
25127 00007F72 30D2                <1>       xor    dl, dl
25128 00007F74 66F7F1              <1>       div    cx
25129 00007F77 80C230              <1>       add    dl, '0'
25130 00007F7A 8817                <1>       mov    [edi], dl
25131 00007F7C 4F                  <1>       dec    edi
25132 00007F7D 6609C0              <1>       or     ax, ax
25133 00007F80 75F0                <1>       jnz    short loc_dfc_rediv
25134                              <1>
25135 00007F82 BF[D0580100]        <1>       mov     edi, TFS_Dec_End
25136                              <1>         ;mov    byte [edi], 0
25137 00007F87 A1[BE580100]        <1>       mov     eax, [Total_FSize]
25138                              <1>       ;mov    ecx, 10
25139                              <1> rediv_tfs_hex:
25140                              <1>       ;sub   edx, edx
25141 00007F8C 28D2                <1>       sub    dl, dl
25142 00007F8E F7F1                <1>       div    ecx
25143 00007F90 80C230              <1>       add    dl, '0'
25144 00007F93 4F                  <1>       dec     edi
25145 00007F94 8817                <1>       mov     [edi], dl
25146 00007F96 21C0                <1>       and    eax, eax
25147 00007F98 75F2                <1>       jnz    short rediv_tfs_hex
25148                              <1>
25149 00007F9A 893D[C2580100]      <1>       mov    [TFS_Dec_Begin], edi
25150 00007FA0 BE[6E060100]        <1>       mov    esi, Decimal_File_Count_Header
25151 00007FA5 E8B3E3FFFF          <1>       call   print_msg
25152 00007FAA BE[76060100]        <1>       mov    esi, str_files
25153 00007FAF E8A9E3FFFF          <1>       call   print_msg
25154 00007FB4 BE[87060100]        <1>       mov    esi, str_dirs
25155 00007FB9 E89FE3FFFF          <1>       call   print_msg
25156 00007FBE 8B35[C2580100]      <1>       mov    esi, [TFS_Dec_Begin]
25157 00007FC4 E894E3FFFF          <1>       call   print_msg
25158 00007FC9 BE[98060100]        <1>       mov    esi, str_bytes
25159 00007FCE E88AE3FFFF          <1>       call   print_msg
25160                              <1>
25161 00007FD3 C3                  <1>       retn
25162                              <1>
25163                              <1> pause_dir_scroll:
25164 00007FD4 28E4                <1>       sub    ah, ah
25165 00007FD6 E83B8CFFFF          <1>       call   int16h
25166 00007FDB 3C1B                <1>       cmp    al, 1Bh
25167 00007FDD 0F8428FFFFFF        <1>         je      loc_dir_ok
25168 00007FE3 C605[D1580100]10    <1>       mov    byte [PrintDir_RowCounter], 16 ; Reset counter
25169 00007FEA E911FFFFFF          <1>         jmp     loc_next_entry
25170                              <1>
25171                              <1> find_first_file:
25172                              <1>       ; 11/02/2016
25173                              <1>       ; 10/02/2016
25174                              <1>       ; 08/02/2016 (TRDOS 386 =  TRDOS v2.0)
25175                              <1>       ; 09/10/2011
25176                              <1>       ; 17/09/2009
25177                              <1>       ; 2005
25178                              <1>       ; INPUT ->
25179                              <1>       ;      ESI = ASCIIZ File/Dir Name Address (in Current Directory)
25180                              <1>       ;      AL = Attributes AND mask (The AND result must be equal to AL)
25181                              <1>       ;            bit 0 = Read Only
25182                              <1>       ;            bir 1 = Hidden
25183                              <1>       ;            bit 2 = System
25184                              <1>       ;            bit 3 = Volume Label
25185                              <1>       ;            bit 4 = Directory
25186                              <1>       ;            bit 5 = Archive
25187                              <1>       ;            bit 6 = Reserved, must be 0
25188                              <1>       ;            bit 7 = Reserved, must be 0
```

```
25189                                  <1>       ;       AH = Attributes Negative AND mask (The AND result must be ZERO)
25190                                  <1>       ;
25191                                  <1>       ; OUTPUT ->
25192                                  <1>       ;     CF = 1 -> Error, Error Code in EAX (AL)
25193                                  <1>       ;     CF = 0 ->
25194                                  <1>       ;          ESI = Directory Entry (FindFile_DirEntry) Location
25195                                  <1>       ;          EDI = Directory Buffer Directory Entry Location
25196                                  <1>       ;          EAX = File Size
25197                                  <1>       ;           BL = Attributes of The File/Directory
25198                                  <1>       ;           BH = Long Name Yes/No Status (>0 is YES)
25199                                  <1>       ;             DX > 0 : Ambiguous filename chars are used
25200                                  <1>       ;
25201                                  <1>       ; (EAX, EBX, ECX, EDX, ESI, EDI will be changed)
25202                                  <1>
25203 00007FEF 66A3[82580100]          <1>       mov    [FindFile_AttributesMask], ax
25204 00007FF5 BF[84580100]            <1>       mov    edi, FindFile_DirEntry ; TR-DOS Fullfilename formatted buffer
25205 00007FFA 31C0                    <1>       xor    eax, eax
25206 00007FFC B90B000000              <1>       mov    ecx, 11
25207 00008001 F3AB                    <1>       rep    stosd  ; 44 bytes
25208                                  <1>       ;stosw       ; +2 bytes
25209                                  <1>
25210 00008003 BF[74580100]            <1>       mov    edi, FindFile_Name ; FFF structure, offset 66
25211 00008008 39FE                    <1>       cmp    esi, edi
25212 0000800A 7408                    <1>       je     short loc_fff_mfn_ok
25213 0000800C 89FA                    <1>       mov    edx, edi
25214                                  <1>        ; move 13 bytes
25215 0000800E A5                      <1>       movsd
25216 0000800F A5                      <1>       movsd
25217 00008010 A5                      <1>       movsd
25218 00008011 AA                      <1>       stosb
25219 00008012 89D6                    <1>       mov    esi, edx
25220                                  <1> loc_fff_mfn_ok:
25221 00008014 BF[23580100]            <1>       mov    edi, Dir_Entry_Name ; Dir Entry Format File Name
25222 00008019 E81D210000              <1>       call   convert_file_name
25223 0000801E 89FE                    <1>       mov    esi, edi ; offset Dir_Entry_Name
25224                                  <1>
25225 00008020 66A1[82580100]          <1>       mov    ax, [FindFile_AttributesMask]
25226                                  <1>       ;xor   ecx, ecx
25227 00008026 30C9                    <1>       xor    cl, cl
25228 00008028 E8191E0000              <1>       call   locate_current_dir_file
25229 0000802D 726E                    <1>       jc     short loc_fff_retn
25230                                  <1>       ; EDI = Directory Entry
25231                                  <1>       ; EBX = Directory Buffer Entry Index/Number
25232                                  <1>
25233                                  <1> loc_fff_fnf_ln_check:
25234 0000802F 30ED                    <1>       xor    ch, ch
25235 00008031 80F60F                  <1>       xor    dh, 0Fh
25236 00008034 7408                    <1>       jz     short loc_fff_longname_yes
25237 00008036 882D[81580100]          <1>       mov    [FindFile_LongNameYes], ch ; 0
25238 0000803C EB0C                    <1>       jmp    short loc_fff_longname_no
25239                                  <1>
25240                                  <1> loc_fff_longname_yes:
25241                                  <1>       ;inc   byte [FindFile_LongNameYes]
25242 0000803E 8A0D[8E570100]          <1>       mov    cl, [LFN_EntryLength]
25243 00008044 880D[81580100]          <1>       mov    [FindFile_LongNameEntryLength], cl ; FindFile_LongNameYes
25244                                  <1>
25245                                  <1> loc_fff_longname_no:
25246                                  <1>       ;mov   bx, [DirBuff_CurrentEntry]
25247 0000804A 66891D[AC580100]        <1>       mov    [FindFile_DirEntryNumber], bx
25248 00008051 6689C2                  <1>       mov    dx, ax ; Ambigouos Filename chars used sign > 0
25249                                  <1>
25250 00008054 A0[8E4E0100]            <1>       mov    al, [Current_Drv]
25251 00008059 A2[32580100]            <1>       mov    [FindFile_Drv], al
25252                                  <1>
25253 0000805E A1[884E0100]            <1>       mov    eax, [Current_Dir_FCluster]
25254 00008063 A3[A4580100]            <1>       mov    [FindFile_DirFirstCluster], eax
25255                                  <1>
25256 00008068 A1[BD560100]            <1>       mov    eax, [DirBuff_Cluster]
25257 0000806D A3[A8580100]            <1>       mov    [FindFile_DirCluster], eax
25258                                  <1>
25259 00008072 66FF05[AE580100]        <1>       inc    word [FindFile_MatchCounter]
25260                                  <1>
25261 00008079 89FB                    <1>       mov    ebx, edi
25262 0000807B 89FE                    <1>       mov    esi, edi
25263 0000807D BF[84580100]            <1>       mov    edi, FindFile_DirEntry
25264 00008082 89F8                    <1>       mov    eax, edi
25265 00008084 B108                    <1>       mov    cl, 8
25266 00008086 F3A5                    <1>       rep    movsd
25267 00008088 89C6                    <1>       mov    esi, eax
25268 0000808A 89DF                    <1>       mov    edi, ebx
25269                                  <1>
25270 0000808C A1[A0580100]            <1>       mov    eax, [FindFile_DirEntry+28] ; File Size
25271                                  <1>
25272 00008091 8A1D[8F580100]          <1>       mov    bl, [FindFile_DirEntry+11] ; File Attributes
25273 00008097 8A3D[81580100]          <1>       mov    bh, [FindFile_LongNameYes]
25274                                  <1>
25275                                  <1>       ;mov   cx, [DirBuff_EntryCounter]
25276                                  <1>       ;mov   [FindFile_DirEntryNumber], cx
25277                                  <1>       ;mov   cx, [FindFile_DirEntryNumber]
25278                                  <1>       ; ecx = 0
25279                                  <1>
25280                                  <1> loc_fff_retn:
25281 0000809D C3                      <1>       retn
25282                                  <1>
25283                                  <1> find_next_file:
25284                                  <1>       ; 15/10/2016
25285                                  <1>       ; 10/02/2016
25286                                  <1>       ; 08/02/2016 (TRDOS 386 =  TRDOS v2.0)
25287                                  <1>       ; 06/02/2011
25288                                  <1>       ; 17/09/2009
25289                                  <1>       ; 2005
25290                                  <1>       ; INPUT ->
```

```
25291                                  <1>       ;      NONE, Find First File Parameters
25292                                  <1>       ; OUTPUT ->
25293                                  <1>       ;      CF = 1 -> Error, Error Code in EAX (AL)
25294                                  <1>       ;      CF = 0 ->
25295                                  <1>       ;          ESI = Directory Entry (FindFile_DirEntry) Location
25296                                  <1>       ;          EDI = Directory Buffer Directory Entry Location
25297                                  <1>       ;          EAX = File Size
25298                                  <1>       ;            BL = Attributes of The File/Directory
25299                                  <1>       ;            BH = Long Name Yes/No Status (>0 is YES)
25300                                  <1>       ;              DX > 0 : Ambiguous filename chars are used
25301                                  <1>       ;
25302                                  <1>       ; (EAX, EBX, ECX, EDX, ESI, EDI will be changed)
25303                                  <1>
25304 0000809E 66833D[AE580100]00      <1>       cmp    word [FindFile_MatchCounter], 0
25305 000080A6 7707                    <1>       ja     short loc_start_search_next_file
25306                                  <1>
25307                                  <1> loc_fnf_stc_retn:
25308 000080A8 F9                      <1>       stc
25309                                  <1> loc_fnf_ax12h_retn:
25310 000080A9 B80C000000              <1>       mov    eax, 12 ; No More files
25311                                  <1> ;loc_fnf_retn:
25312 000080AE C3                      <1>       retn
25313                                  <1>
25314                                  <1> loc_start_search_next_file:
25315 000080AF 668B1D[AC580100]        <1>       mov    bx, [FindFile_DirEntryNumber]
25316 000080B6 6643                    <1>       inc    bx
25317 000080B8 663B1D[BB560100]        <1>       cmp    bx, [DirBuff_LastEntry]
25318 000080BF 7719                    <1>       ja     short loc_cont_search_next_file
25319                                  <1>
25320                                  <1> loc_fnf_search:
25321 000080C1 BE[23580100]            <1>       mov    esi, Dir_Entry_Name
25322 000080C6 66A1[82580100]          <1>       mov    ax, [FindFile_AttributesMask]
25323 000080CC 6631C9                  <1>       xor    cx, cx
25324 000080CF E8741E0000              <1>       call   find_directory_entry
25325 000080D4 0F8355FFFFFF            <1>         jnc     loc_fff_fnf_ln_check
25326                                  <1>
25327                                  <1> loc_cont_search_next_file:
25328 000080DA 31DB                    <1>       xor    ebx, ebx
25329 000080DC 8A3D[8E4E0100]          <1>       mov    bh, [Current_Drv]
25330 000080E2 BE00010900              <1>       mov    esi, Logical_DOSDisks
25331 000080E7 01DE                    <1>       add    esi, ebx
25332                                  <1>
25333 000080E9 803D[8C4E0100]00        <1>       cmp    byte [Current_Dir_Level], 0
25334 000080F0 7608                    <1>       jna    short loc_fnf_check_FAT_type
25335 000080F2 807E0301                <1>       cmp    byte [esi+LD_FATType], 1
25336 000080F6 72B1                    <1>       jb     short loc_fnf_ax12h_retn
25337 000080F8 EB06                    <1>       jmp    short loc_fnf_check_next_cluster
25338                                  <1>
25339                                  <1> loc_fnf_check_FAT_type:
25340 000080FA 807E0303                <1>       cmp    byte [esi+LD_FATType], 3
25341 000080FE 72A9                    <1>       jb     short loc_fnf_ax12h_retn
25342                                  <1>
25343                                  <1> loc_fnf_check_next_cluster:
25344 00008100 A1[BD560100]            <1>       mov    eax, [DirBuff_Cluster]
25345 00008105 E813380000              <1>       call   get_next_cluster
25346 0000810A 7306                    <1>       jnc    short loc_fnf_load_next_dir_cluster
25347 0000810C 09C0                    <1>       or     eax, eax
25348 0000810E 7498                    <1>       jz     short loc_fnf_stc_retn
25349                                  <1>       ;mov   eax, 17 ;Drive not ready or read error
25350 00008110 F5                      <1>       cmc    ;stc
25351                                  <1> loc_fnf_retn:
25352 00008111 C3                      <1>       retn
25353                                  <1>
25354                                  <1> loc_fnf_load_next_dir_cluster:
25355 00008112 E8EC390000              <1>       call   load_FAT_sub_directory
25356 00008117 72F8                    <1>       jc     short loc_fnf_retn
25357 00008119 6631DB                  <1>       xor    bx, bx
25358 0000811C 66891D[AC580100]        <1>       mov    [FindFile_DirEntryNumber], bx
25359 00008123 EB9C                    <1>       jmp    short loc_fnf_search
25360                                  <1>
25361                                  <1> get_and_print_longname:
25362                                  <1>       ; 16/10/2016
25363                                  <1>       ; 13/02/2016 (TRDOS 386 =  TRDOS v2.0)
25364                                  <1>       ; 24/01/2010
25365                                  <1>       ; 17/10/2009 (CMD_INTR.ASM, 'cmp_cmd_longname')
25366                                  <1> get_longname_fchar:
25367 00008125 803E20                  <1>       cmp    byte [esi], 20h
25368 00008128 7701                    <1>       ja     short loc_find_longname
25369                                  <1>       ;jb    short loc_longname_retn
25370                                  <1>       ;inc   esi
25371                                  <1>       ;je    short get_longname_fchar
25372                                  <1> ;loc_longname_retn:
25373 0000812A C3                      <1>       retn
25374                                  <1> loc_find_longname:
25375 0000812B E87F210000              <1>       call   find_longname
25376 00008130 7328                    <1>       jnc    short loc_print_longname
25377                                  <1>
25378 00008132 08C0                    <1>       or     al, al
25379 00008134 741A                    <1>       jz     short loc_longname_not_found
25380                                  <1>
25381                                  <1>       ; 16/10/2016 (15h -> 15, 17)
25382 00008136 3C0F                    <1>       cmp    al, 15
25383 00008138 0F84C5F7FFFF            <1>       je     cd_drive_not_ready ; drive not ready
25384                                  <1>                                 ; or
25385 0000813E 3C11                    <1>       cmp    al, 17              ; read error
25386 00008140 0F84BDF7FFFF            <1>       je     cd_drive_not_ready
25387                                  <1>
25388                                  <1> loc_ln_file_dir_not_found:
25389 00008146 BE[C3050100]            <1>       mov    esi, Msg_File_Directory_Not_Found
25390                                  <1>       ;call  print_msg
25391                                  <1>         ;retn
25392 0000814B E90DE2FFFF              <1>       jmp    print_msg
```

```
25393                                   <1>
25394                                   <1> loc_longname_not_found:
25395 00008150 BE[E2050100]               <1>         mov     esi, Msg_LongName_Not_Found
25396                                   <1>       ;call  print_msg
25397                                   <1>         ;retn
25398 00008155 E903E2FFFF                 <1>       jmp    print_msg
25399                                   <1>
25400                                   <1> loc_print_longname:
25401                                   <1>       ;mov   esi, LongFileName
25402 0000815A BF[8E4F0100]               <1>       mov    edi, TextBuffer
25403 0000815F 57                         <1>       push   edi
25404 00008160 3C00                       <1>       cmp    al, 0
25405 00008162 7708                       <1>       ja     short loc_print_longname_1
25406                                   <1> loc_print_FS_longname: ; Singlix FS (64 byte ASCIIZ file name)
25407 00008164 AC                         <1>       lodsb
25408 00008165 AA                         <1>       stosb
25409 00008166 08C0                       <1>       or     al, al
25410 00008168 75FA                       <1>       jnz    short loc_print_FS_longname
25411 0000816A EB07                       <1>       jmp    short loc_print_longname_2
25412                                   <1>       ;
25413                                   <1> loc_print_longname_1: ; MS Windows long name (UNICODE chars)
25414 0000816C 66AD                       <1>       lodsw
25415 0000816E AA                         <1>       stosb
25416 0000816F 08C0                       <1>       or     al, al
25417 00008171 75F9                       <1>       jnz    short loc_print_longname_1
25418                                   <1>       ;
25419                                   <1> loc_print_longname_2:
25420 00008173 5E                         <1>       pop    esi
25421 00008174 E8E4E1FFFF                 <1>       call   print_msg
25422 00008179 BE[030F0100]               <1>       mov    esi, nextline
25423                                   <1>       ;call  print_msg
25424                                   <1>       ;retn
25425 0000817E E9DAE1FFFF                 <1>       jmp    print_msg
25426                                   <1>
25427                                   <1> show_file:
25428                                   <1>       ; 18/02/2016
25429                                   <1>       ; 17/02/2016
25430                                   <1>       ; 15/02/2016 (TRDOS 386 =  TRDOS v2.0)
25431                                   <1>       ; 13/09/2011 (CMD_INTR.ASM, 'cmp_cmd_show')
25432                                   <1>       ; 08/11/2009
25433                                   <1>
25434                                   <1> loc_show_parse_path_name:
25435 00008183 BF[32580100]               <1>       mov    edi, FindFile_Drv
25436 00008188 E879200000                 <1>       call   parse_path_name
25437 0000818D 0F825AF9FFFF               <1>       jc     loc_cmd_failed
25438                                   <1>
25439                                   <1> loc_show_check_filename_exists:
25440 00008193 BE[74580100]               <1>       mov    esi, FindFile_Name
25441 00008198 803E20                     <1>       cmp    byte [esi], 20h
25442 0000819B 0F864CF9FFFF               <1>       jna    loc_cmd_failed
25443                                   <1>
25444                                   <1>       ; 15/02/2016 (invalid file name check)
25445 000081A1 E809020000                 <1>       call   check_filename
25446 000081A6 730A                       <1>       jnc    short loc_show_change_drv
25447                                   <1>
25448 000081A8 BE[AE060100]               <1>       mov    esi, Msg_invalid_name_chars
25449 000081AD E9ABE1FFFF                 <1>       jmp    print_msg
25450                                   <1>
25451                                   <1> loc_show_change_drv:
25452 000081B2 8A35[8E4E0100]             <1>       mov    dh, [Current_Drv]
25453 000081B8 8835[EE560100]             <1>       mov    [RUN_CDRV], dh
25454 000081BE 8A15[32580100]             <1>       mov    dl, [FindFile_Drv]
25455 000081C4 38F2                       <1>       cmp    dl, dh
25456 000081C6 740B                       <1>       je     short loc_show_change_directory
25457 000081C8 E890EAFFFF                 <1>       call   change_current_drive
25458                                   <1>       ;jc    loc_file_rw_cmd_failed
25459 000081CD 0F8245F9FFFF               <1>       jc     loc_run_cmd_failed
25460                                   <1>
25461                                   <1> loc_show_change_directory:
25462 000081D3 803D[33580100]20           <1>       cmp    byte [FindFile_Directory], 20h
25463 000081DA 7618                       <1>       jna    short loc_findload_showfile
25464                                   <1>
25465 000081DC FE05[98020100]             <1>       inc    byte [Restore_CDIR]
25466 000081E2 BE[33580100]               <1>       mov    esi, FindFile_Directory
25467 000081E7 30E4                       <1>       xor    ah, ah ; CD_COMMAND sign -> 0
25468 000081E9 E8041A0000                 <1>       call   change_current_directory
25469                                   <1>       ;jc    loc_file_rw_cmd_failed
25470 000081EE 0F8224F9FFFF               <1>       jc     loc_run_cmd_failed
25471                                   <1>
25472                                   <1> ;loc_show_change_prompt_dir_string:
25473                                   <1>       ;call  change_prompt_dir_string
25474                                   <1>
25475                                   <1> loc_findload_showfile:
25476                                   <1>       ; 15/02/2016
25477 000081F4 BE[74580100]               <1>       mov    esi, FindFile_Name
25478 000081F9 BF[23580100]               <1>       mov    edi, Dir_Entry_Name ; Dir Entry Format File Name
25479 000081FE E8381F0000                 <1>       call   convert_file_name
25480 00008203 89FE                       <1>       mov    esi, edi ; offset Dir_Entry_Name
25481                                   <1>
25482 00008205 28C0                       <1>       sub    al, al ; Attrib AND mask = 0
25483                                   <1>       ; Directory attribute : 10h
25484                                   <1>       ; Volume name attribute: 8h
25485 00008207 B418                       <1>       mov    ah, 00011000b ; 18h (Attrib NAND, AND --> zero mask)
25486                                   <1>       ;
25487 00008209 6631C9                     <1>       xor    cx, cx
25488 0000820C E8351C0000                 <1>       call   locate_current_dir_file
25489                                   <1>       ;jc    loc_file_rw_cmd_failed
25490 00008211 0F8201F9FFFF               <1>       jc     loc_run_cmd_failed
25491                                   <1>
25492                                   <1> loc_show_load_file:
25493                                   <1>       ; EDI = Directory Entry
25494 00008217 668B4714                   <1>       mov    ax, [edi+DirEntry_FstClusHI] ; First Cluster High Word
```

```
25495 0000821B C1E010              <1>       shl    eax, 16
25496 0000821E 668B471A            <1>       mov    ax, [edi+DirEntry_FstClusLO] ; First Cluster Low Word
25497 00008222 A3[DC580100]        <1>       mov    [Show_Cluster], eax
25498 00008227 8B471C              <1>       mov    eax, [edi+DirEntry_FileSize] ; File Size
25499 0000822A 21C0                <1>       and    eax, eax ; Empty file !
25500 0000822C 0F8491000000        <1>         jz     end_of_show_file
25501 00008232 A3[E0580100]        <1>       mov    [Show_FileSize], eax
25502 00008237 31C0                <1>       xor    eax, eax
25503 00008239 A3[E4580100]        <1>       mov    [Show_FilePointer], eax ; 0
25504 0000823E 66A3[E8580100]      <1>       mov    [Show_ClusterPointer], ax ; 0
25505 00008244 29DB                <1>       sub    ebx, ebx
25506 00008246 8A3D[8E4E0100]      <1>       mov    bh, [Current_Drv]
25507 0000824C BE00010900          <1>       mov    esi, Logical_DOSDisks
25508 00008251 01DE                <1>       add    esi, ebx
25509 00008253 8935[D8580100]      <1>       mov    [Show_LDDDT], esi ; Logical DOS Drv Description Table addr
25510                              <1>
25511 00008259 807E0300            <1>       cmp    byte [esi+LD_FATType], 0
25512 0000825D 7713                <1>       ja     short loc_show_calculate_cluster_size
25513                              <1>       ; Singlix FS
25514                              <1>       ; First Cluster Number is FDT number (in compatibility buffer)
25515 0000825F 8B15[DC580100]      <1>       mov    edx, [Show_Cluster] ; Compatibility dir. buffer value (FDT)
25516 00008265 8915[D4580100]      <1>       mov    [Show_FDT], edx
25517 0000826B 31C0                <1>       xor    eax, eax
25518 0000826D A3[DC580100]        <1>       mov    [Show_Cluster], eax ; Sector index  = 0
25519                              <1>                                  ; (next time it will be 1)
25520                              <1> loc_show_calculate_cluster_size:
25521 00008272 668B5E11            <1>       mov    bx, [esi+LD_BPB+BPB_BytsPerSec] ; FAT 12-16-32 (512)
25522                              <1>       ; BX = 512 = [esi+LD_FS_BytesPerSec] ; Singlix FS
25523 00008276 8A4613              <1>       mov    al, [esi+LD_BPB+BPB_SecPerClust] ; FAT 12-16-32 (<= 128)
25524                              <1>       ; AL = 1 = [esi+LD_FS_Reserved2] ; SectPerClust for Singlix FS
25525 00008279 F7E3                <1>       mul    ebx
25526                              <1>
25527                              <1>       ;cmp   eax, 65536 ; non-compatible (very big) cluster size
25528                              <1>       ;ja    short end_of_show_file
25529 0000827B 66A3[EA580100]      <1>       mov    [Show_ClusterSize], ax
25530                              <1>
25531                              <1> loc_start_show_file:
25532 00008281 BE[030F0100]        <1>       mov    esi, nextline
25533 00008286 E8D2E0FFFF          <1>       call   print_msg
25534                              <1>
25535 0000828B A1[DC580100]        <1>       mov    eax, [Show_Cluster]
25536 00008290 C605[EC580100]17    <1>       mov    byte [Show_RowCount], 23
25537                              <1>
25538                              <1>       ; 17/02/2016
25539 00008297 8B35[D8580100]      <1>       mov    esi, [Show_LDDDT]
25540                              <1>
25541                              <1> loc_show_next_cluster:
25542                              <1>       ; 15/02/2016
25543 0000829D BB00000700          <1>       mov    ebx, Cluster_Buffer ; 70000h (for current TRDOS 386 version)
25544                              <1>       ; ESI = Logical DOS drv description table address
25545 000082A2 E89A380000          <1>       call   read_cluster
25546                              <1>       ;jc    loc_file_rw_cmd_failed
25547 000082A7 0F826BF8FFFF        <1>       jc     loc_run_cmd_failed
25548                              <1>
25549 000082AD 31DB                <1>       xor    ebx, ebx
25550                              <1> loc_show_next_byte:
25551 000082AF 803D[EC580100]00    <1>       cmp    byte [Show_RowCount], 0
25552 000082B6 7521                <1>       jne    short pass_show_wait_for_key
25553 000082B8 30E4                <1>       xor    ah, ah
25554 000082BA E85789FFFF          <1>       call   int16h
25555 000082BF 3C1B                <1>       cmp    al, 1Bh
25556 000082C1 750F                <1>       jne    short pass_exit_show
25557                              <1> end_of_show_file:
25558                              <1> pass_show_file:
25559 000082C3 BE[030F0100]        <1>       mov    esi, nextline
25560 000082C8 E890E0FFFF          <1>       call   print_msg
25561 000082CD E94D010000          <1>       jmp    loc_file_rw_restore_retn
25562                              <1>
25563                              <1> pass_exit_show:
25564 000082D2 C605[EC580100]14    <1>       mov    byte [Show_RowCount], 20
25565                              <1> pass_show_wait_for_key:
25566 000082D9 81C300000700        <1>       add    ebx, Cluster_Buffer
25567 000082DF 8A03                <1>       mov    al, [ebx]
25568 000082E1 3C0D                <1>       cmp    al, 0Dh
25569 000082E3 0F8590000000        <1>         jne    loc_show_check_tab_space
25570 000082E9 FE0D[EC580100]      <1>       dec    byte [Show_RowCount]
25571                              <1> pass_show_dec_rowcount:
25572 000082EF B307                <1>       mov    bl, 7 ; (light gray character color, black background)
25573 000082F1 8A3D[F64D0100]      <1>       mov    bh, [ACTIVE_PAGE] ; [ptty]
25574 000082F7 E8B699FFFF          <1>       call   _write_tty
25575                              <1> loc_show_check_eof:
25576 000082FC FF05[E4580100]      <1>       inc    dword [Show_FilePointer]
25577 00008302 A1[E4580100]        <1>       mov    eax, [Show_FilePointer]
25578 00008307 3B05[E0580100]      <1>       cmp    eax, [Show_FileSize]
25579 0000830D 73B4                <1>       jnb    short end_of_show_file
25580 0000830F 66FF05[E8580100]    <1>       inc    word [Show_ClusterPointer]
25581 00008316 0FB71D[E8580100]    <1>       movzx  ebx, word [Show_ClusterPointer]
25582                              <1>
25583                              <1>       ; 17/02/2016
25584                              <1>       ; (sector boundary -9 bits- check, 512 = 0)
25585 0000831D 66F7C3FF01          <1>         test    bx, 1FFh ;  1 to 511
25586 00008322 758B                <1>       jnz    short loc_show_next_byte
25587                              <1>
25588                              <1>       ; 16/02/2016
25589 00008324 8B35[D8580100]      <1>       mov    esi, [Show_LDDDT]
25590                              <1>       ;
25591 0000832A 807E0300            <1>       cmp    byte [esi+LD_FATType], 0
25592 0000832E 7719                <1>       ja     short loc_show_check_fat_cluster_size
25593                              <1>
25594                              <1>       ; Singlix FS
25595                              <1>       ; 1 sector, more... (cluster size = 1 sector)
25596 00008330 A1[DC580100]        <1>       mov    eax, [Show_Cluster]
```

```
25597 00008335 40                  <1>       inc    eax
25598 00008336 A3[DC580100]        <1>       mov    [Show_Cluster], eax
25599                              <1>
25600 0000833B 6621DB              <1>       and    bx, bx ; 65536 -> 0
25601 0000833E 0F856BFFFFFF        <1>         jnz  loc_show_next_byte
25602 00008344 E954FFFFFF          <1>       jmp     loc_show_next_cluster
25603                              <1>
25604                              <1> loc_show_check_fat_cluster_size:
25605                              <1>       ; 17/02/2016
25606 00008349 663B1D[EA580100]    <1>       cmp    bx, [Show_ClusterSize] ; cluster size in bytes
25607 00008350 0F8259FFFFFF        <1>         jb   loc_show_next_byte
25608 00008356 66C705[E8580100]00- <1>       mov    word [Show_ClusterPointer], 0
25609 0000835E 00                  <1>
25610                              <1>
25611 0000835F A1[DC580100]        <1>       mov    eax, [Show_Cluster]
25612                              <1>       ;mov   esi, [Show_LDDDT]
25613                              <1> loc_show_get_next_cluster:
25614 00008364 E8B4350000          <1>       call   get_next_cluster
25615                              <1>       ;jc    loc_file_rw_cmd_failed
25616 00008369 0F82A9F7FFFF        <1>       jc     loc_run_cmd_failed
25617                              <1> loc_show_update_ccluster:
25618 0000836F A3[DC580100]        <1>       mov    [Show_Cluster], eax
25619 00008374 E924FFFFFF          <1>         jmp     loc_show_next_cluster
25620                              <1>
25621                              <1> loc_show_check_tab_space:
25622 00008379 3C09                <1>       cmp    al, 09h
25623 0000837B 0F856EFFFFFF        <1>         jne     pass_show_dec_rowcount
25624                              <1> loc_show_put_tab_space:
25625 00008381 8A3D[F64D0100]      <1>       mov    bh, [ACTIVE_PAGE] ; [ptty]
25626 00008387 E8B595FFFF          <1>       call   get_cpos
25627                              <1>       ; dl = cursor column
25628 0000838C 80E207              <1>       and    dl, 7 ; 18/02/2016
25629                              <1>       ;shr   bh, 1 ; [ACTIVE_PAGE]
25630 0000838F 8A3D[F64D0100]      <1>       mov    bh, [ACTIVE_PAGE]
25631 00008395 B307                <1>       mov    bl, 7 ; color attribute
25632                              <1> loc_show_put_space_chars:
25633 00008397 B020                <1>       mov    al, 20h ; space
25634                              <1>       ;mov   bh, [ACTIVE_PAGE] ; [ptty]
25635                              <1>       ;mov   bl, 7 ; color attribute
25636 00008399 6652                <1>       push   dx
25637 0000839B E81299FFFF          <1>       call   _write_tty
25638 000083A0 665A                <1>       pop    dx
25639                              <1>       ; 18/02/2016
25640 000083A2 80FA07              <1>       cmp    dl, 7
25641 000083A5 0F8351FFFFFF        <1>       jnb    loc_show_check_eof
25642 000083AB FEC2                <1>       inc    dl
25643 000083AD EBE8                <1>       jmp    short loc_show_put_space_chars
25644                              <1>
25645                              <1> check_filename:
25646                              <1>       ; 10/10/2016
25647                              <1>       ; 15/02/2016 (TRDOS 386 =  TRDOS v2.0)
25648                              <1>       ; 07/08/2010 (FILE.ASM, 'proc_check_filename')
25649                              <1>       ; 10/07/2010
25650                              <1>       ; Derived from 'proc_check_filename'
25651                              <1>       ; in the old TRDOS.ASM (09/02/2005).
25652                              <1>       ;
25653                              <1>       ; INPUT ->
25654                              <1>       ;     ESI = Dot File Name Location
25655                              <1>       ; OUTPUT ->
25656                              <1>       ;     cf = 1 -> error code in AL
25657                              <1>       ;          AL = ERR_INV_FILE_NAME (=26)
25658                              <1>       ;             Invalid file name chars
25659                              <1>       ;     cf = 0 -> valid file name
25660                              <1>       ;
25661                              <1>       ;(EAX, ECX, EDI will be changed)
25662                              <1>
25663                              <1> check_invalid_filename_chars:
25664                              <1>       ; 15/02/2016 (TRDOS 386 =  TRDOS v2.0)
25665                              <1>       ; 10/07/2010 (FILE.ASM, 'proc_check_invalid_filename_chars')
25666                              <1>       ; 10/02/2010
25667                              <1>       ; Derived from 'proc_check_invalid_filename_chars'
25668                              <1>       ; in the old TRDOS.ASM (09/02/2005).
25669                              <1>       ;
25670                              <1>       ; INPUT ->
25671                              <1>       ;     ESI = ASCIIZ FileName
25672                              <1>       ; OUTPUT ->
25673                              <1>       ;     cf = 1 -> invalid
25674                              <1>       ;     cf = 0 -> valid
25675                              <1>       ;
25676                              <1>       ;(EAX, ECX, EDI will be changed)
25677                              <1>
25678 000083AF 56                  <1>       push   esi
25679                              <1>
25680 000083B0 BF[97030100]        <1>         mov     edi, invalid_fname_chars
25681 000083B5 AC                  <1>       lodsb
25682                              <1> check_filename_next_char:
25683 000083B6 B914000000          <1>       mov    ecx, sizeInvFnChars
25684 000083BB BF[97030100]        <1>       mov    edi, invalid_fname_chars
25685                              <1> loc_scan_invalid_filename_char:
25686 000083C0 AE                  <1>       scasb
25687 000083C1 741F                <1>       je     short loc_invalid_filename_stc
25688 000083C3 E2FB                <1>       loop   loc_scan_invalid_filename_char
25689 000083C5 AC                  <1>       lodsb
25690 000083C6 3C1F                <1>       cmp    al, 1Fh  ; 20h and above
25691 000083C8 77EC                <1>       ja     short check_filename_next_char
25692                              <1>
25693                              <1> check_filename_dot:
25694 000083CA 8B3424              <1>       mov    esi, [esp]
25695                              <1>
25696 000083CD B421                <1>       mov    ah, 21h
25697 000083CF B908000000          <1>       mov    ecx, 8
25698                              <1> loc_check_filename_next_char:
```

```
25699 000083D4 AC                <1>       lodsb
25700 000083D5 3C2E              <1>       cmp    al, 2Eh
25701 000083D7 7511              <1>       jne    short pass_check_fn_dot_check
25702                            <1> loc_check_filename_ext_0:
25703 000083D9 AC                <1>       lodsb
25704 000083DA 38E0              <1>       cmp    al, ah ; 21h
25705 000083DC 7205              <1>       jb     short loc_invalid_filename
25706 000083DE 3C2E              <1>       cmp    al, 2Eh
25707 000083E0 7519              <1>       jne    short loc_check_filename_ext_1
25708                            <1>
25709                            <1> loc_invalid_filename_stc:
25710                            <1> loc_check_fn_stc_rtn:
25711 000083E2 F9                <1>       stc
25712                            <1> loc_invalid_filename:
25713                            <1>       ; 10/10/2016 (0Bh -> 26)
25714 000083E3 B81A000000        <1>       mov    eax, ERR_INV_FILE_NAME ; (=26)
25715                            <1>       ; Invalid file name chars
25716                            <1> loc_check_fn_rtn:
25717 000083E8 5E                <1>       pop    esi
25718 000083E9 C3                <1>       retn
25719                            <1>
25720                            <1> pass_check_fn_dot_check:
25721 000083EA 38E0              <1>       cmp    al, ah ; 21h
25722 000083EC 7224              <1>       jb     short loc_check_fn_clc_rtn
25723 000083EE E2E4              <1>       loop   loc_check_filename_next_char
25724 000083F0 AC                <1>       lodsb
25725 000083F1 38E0              <1>       cmp    al, ah ; 21h
25726 000083F3 721D              <1>       jb     short loc_check_fn_clc_rtn
25727 000083F5 3C2E              <1>       cmp    al, 2Eh
25728 000083F7 75E9              <1>       jne    short loc_check_fn_stc_rtn
25729 000083F9 EBDE              <1>       jmp    short loc_check_filename_ext_0
25730                            <1>
25731                            <1> loc_check_filename_ext_1:
25732 000083FB AC                <1>       lodsb
25733 000083FC 38E0              <1>       cmp    al, ah ; 21h
25734 000083FE 7212              <1>       jb     short loc_check_fn_clc_rtn
25735 00008400 3C2E              <1>       cmp    al, 2Eh
25736 00008402 74DE              <1>       je     short loc_check_fn_stc_rtn
25737 00008404 AC                <1>       lodsb
25738 00008405 38E0              <1>       cmp    al, ah ; 21h
25739 00008407 7209              <1>       jb     short loc_check_fn_clc_rtn
25740 00008409 3C2E              <1>       cmp    al, 2Eh
25741 0000840B 74D5              <1>       je     short loc_check_fn_stc_rtn
25742 0000840D AC                <1>       lodsb
25743 0000840E 38E0              <1>       cmp    al, ah ; 21h
25744 00008410 73D0              <1>       jnb    short loc_check_fn_stc_rtn
25745                            <1>
25746                            <1> loc_check_fn_clc_rtn:
25747 00008412 5E                <1>       pop    esi
25748 00008413 F8                <1>       clc
25749 00008414 C3                <1>       retn
25750                            <1>
25751                            <1> loc_print_deleted_message:
25752 00008415 BE[83070100]      <1>       mov    esi, Msg_Deleted
25753 0000841A E83EDFFFFF        <1>       call   print_msg
25754                            <1>
25755                            <1>       ;clc
25756                            <1>
25757                            <1> loc_file_rw_restore_retn:
25758                            <1>       ; 15/02/2016 (TRDOS 386 =  TRDOS v2.0)
25759                            <1>       ; 28/02/2010 (CMD_INTR.ASM)
25760                            <1> loc_file_rw_cmd_failed:
25761 0000841F 9C                <1>       pushf
25762 00008420 E84DF7FFFF        <1>       call   restore_cdir_after_cmd_fail
25763 00008425 9D                <1>       popf
25764 00008426 720D              <1>       jc     short loc_file_rw_check_write_fault
25765 00008428 C3                <1>       retn
25766                            <1>
25767                            <1> loc_permission_denied:
25768                            <1>       ; 27/02/2016
25769 00008429 BE[90070100]      <1>       mov    esi, Msg_Permission_Denied
25770 0000842E E82ADFFFFF        <1>       call   print_msg
25771 00008433 EBEA              <1>       jmp    short loc_file_rw_restore_retn
25772                            <1>
25773                            <1> loc_file_rw_check_write_fault:
25774                            <1>       ;cmp   al, 1Dh ; Write Fault
25775 00008435 3C12              <1>         cmp  al, 18 ; 05/11/2016
25776 00008437 0F85E0F6FFFF      <1>       jne     loc_run_cmd_failed_cmp_al
25777 0000843D BE[78050100]      <1>       mov    esi, Msg_Not_Ready_Write_Err
25778                            <1>       ;call  print_msg
25779                            <1>       ;retn
25780 00008442 E916DFFFFF        <1>       jmp    print_msg
25781                            <1>
25782                            <1> make_directory:
25783                            <1>       ; 21/02/2016 (TRDOS 386 =  TRDOS v2.0)
25784                            <1>       ; 12/03/2011 (CMD_INTR.ASM, 'cmp_cmd_mkdir')
25785                            <1>       ; 14/08/2010
25786                            <1>       ; 10/07/2010
25787                            <1>       ; 29/11/2009
25788                            <1>       ;
25789                            <1> get_mkdir_fchar:
25790                            <1>       ; esi = directory name
25791 00008447 803E20            <1>       cmp    byte [esi], 20h
25792 0000844A 7701              <1>         ja   short loc_mkdir_parse_path_name
25793                            <1>
25794                            <1> loc_mkdir_nodirname_retn:
25795 0000844C C3                <1>       retn
25796                            <1>
25797                            <1> loc_mkdir_parse_path_name:
25798 0000844D BF[32580100]      <1>       mov    edi, FindFile_Drv
25799 00008452 E8AF1D0000        <1>         call    parse_path_name
25800 00008457 0F8290F6FFFF      <1>       jc     loc_cmd_failed
```

```
25801                                  <1>
25802                                  <1> loc_mkdir_check_dirname_exists:
25803 0000845D BE[74580100]            <1>       mov   esi, FindFile_Name
25804 00008462 803E20                  <1>       cmp   byte [esi], 20h
25805 00008465 0F8682F6FFFF            <1>       jna   loc_cmd_failed
25806 0000846B 8935[F0580100]          <1>       mov   [DelFile_FNPointer], esi
25807 00008471 E839FFFFFF              <1>       call  check_filename
25808 00008476 7259                    <1>       jc    short loc_mkdir_invalid_dir_name_chars
25809                                  <1>
25810                                  <1> loc_mkdir_drv:
25811 00008478 8A35[8E4E0100]          <1>       mov   dh, [Current_Drv]
25812 0000847E 8835[EE560100]          <1>       mov   [RUN_CDRV], dh
25813                                  <1>
25814 00008484 8A15[32580100]          <1>       mov   dl, [FindFile_Drv]
25815 0000848A 38F2                    <1>       cmp   dl, dh
25816 0000848C 7407                    <1>       je    short loc_mkdir_change_directory
25817                                  <1>
25818 0000848E E8CAE7FFFF              <1>       call  change_current_drive
25819 00008493 728A                    <1>       jc    loc_file_rw_cmd_failed
25820                                  <1>
25821                                  <1> loc_mkdir_change_directory:
25822 00008495 803D[33580100]20        <1>       cmp   byte [FindFile_Directory], 20h
25823 0000849C 7614                    <1>       jna   short loc_mkdir_find_directory
25824                                  <1>
25825 0000849E FE05[98020100]          <1>       inc   byte [Restore_CDIR]
25826 000084A4 BE[33580100]            <1>       mov   esi, FindFile_Directory
25827 000084A9 30E4                    <1>       xor   ah, ah ; CD_COMMAND sign -> 0
25828 000084AB E842170000              <1>       call  change_current_directory
25829 000084B0 722E                    <1>       jc    short loc_mkdir_check_error_code
25830                                  <1>
25831                                  <1> ;loc_mkdir_change_prompt_dir_string:
25832                                  <1>       ;call  change_prompt_dir_string
25833                                  <1>
25834                                  <1> loc_mkdir_find_directory:
25835                                  <1>       ;mov   esi, FindFile_Name
25836 000084B2 8B35[F0580100]          <1>       mov   esi, [DelFile_FNPointer]
25837                                  <1>       ;xor   eax, eax
25838 000084B8 6631C0                  <1>       xor   ax, ax ; any name (dir, file, volume)
25839 000084BB E82FFBFFFF              <1>       call  find_first_file
25840 000084C0 721E                    <1>       jc    short loc_mkdir_check_error_code
25841                                  <1>
25842                                  <1> loc_mkdir_directory_found:
25843 000084C2 BE[DB060100]            <1>       mov   esi, Msg_Name_Exists
25844 000084C7 E891DEFFFF              <1>       call  print_msg
25845                                  <1>
25846 000084CC E94EFFFFFF              <1>         jmp     loc_file_rw_restore_retn
25847                                  <1>
25848                                  <1> loc_mkdir_invalid_dir_name_chars:
25849 000084D1 BE[AE060100]            <1>       mov   esi, Msg_invalid_name_chars
25850 000084D6 E882DEFFFF              <1>       call  print_msg
25851                                  <1>
25852 000084DB E93FFFFFFF              <1>         jmp     loc_file_rw_restore_retn
25853                                  <1>
25854                                  <1> loc_mkdir_check_error_code:
25855 000084E0 3C02                    <1>       cmp   al, 2
25856                                  <1>       ;je    short loc_mkdir_directory_not_found
25857 000084E2 7406                    <1>       je    short loc_mkdir_ask_for_yes_no
25858 000084E4 F9                      <1>       stc
25859 000084E5 E935FFFFFF              <1>         jmp     loc_file_rw_cmd_failed
25860                                  <1>
25861                                  <1> loc_mkdir_directory_not_found:
25862                                  <1> loc_mkdir_ask_for_yes_no:
25863 000084EA BE[FC060100]            <1>       mov   esi, Msg_DoYouWantMkdir
25864 000084EF E869DEFFFF              <1>       call  print_msg
25865 000084F4 8B35[F0580100]          <1>       mov   esi, [DelFile_FNPointer]
25866 000084FA E85EDEFFFF              <1>       call  print_msg
25867 000084FF BE[1B070100]            <1>       mov   esi, Msg_YesNo
25868 00008504 E854DEFFFF              <1>       call  print_msg
25869                                  <1>
25870 00008509 C605[25070100]20        <1>       mov   byte [Y_N_nextline], 20h
25871                                  <1>
25872                                  <1> loc_mkdir_ask_again:
25873 00008510 30E4                    <1>       xor   ah, ah
25874 00008512 E8FF86FFFF              <1>       call  int16h
25875 00008517 3C1B                    <1>       cmp   al, 1Bh
25876                                  <1>       ;je    short loc_do_not_make_directory
25877 00008519 7447                    <1>       je    short loc_mkdir_y_n_escape
25878 0000851B 24DF                    <1>       and   al, 0DFh ; y -> Y, n -> N
25879 0000851D 3C59                    <1>       cmp   al, 'Y' ; 'yes'
25880 0000851F 7404                    <1>       je    short loc_mkdir_yes_make_directory
25881 00008521 3C4E                    <1>       cmp   al, 'N' ; 'no'
25882 00008523 75EB                    <1>       jne   short loc_mkdir_ask_again
25883                                  <1>
25884                                  <1> loc_do_not_make_directory:
25885                                  <1> loc_mkdir_yes_make_directory:
25886 00008525 A2[25070100]            <1>       mov   [Y_N_nextline], al
25887 0000852A 6650                    <1>       push  ax
25888 0000852C BE[25070100]            <1>       mov   esi, Y_N_nextline
25889 00008531 E827DEFFFF              <1>       call  print_msg
25890 00008536 6658                    <1>       pop   ax
25891                                  <1>       ;cmp   al, 'Y' ; 'yes'
25892                                  <1>       ;cmc
25893                                  <1>         ;jnc loc_file_rw_restore_retn
25894 00008538 3C4E                    <1>       cmp   al, 'N' ; 'no'
25895 0000853A 0F84DFFEFFFF            <1>         je    loc_file_rw_restore_retn
25896                                  <1>
25897                                  <1> loc_mkdir_call_make_sub_directory:
25898 00008540 8B35[F0580100]          <1>       mov   esi, [DelFile_FNPointer]
25899 00008546 B110                    <1>       mov   cl, 10h ; Directory attributes
25900 00008548 E8B81D0000              <1>       call  make_sub_directory
25901                                  <1> loc_rename_file_ok: ; 06/03/2016
25902 0000854D 0F82CCFEFFFF            <1>         jc    loc_file_rw_cmd_failed
```

```
25903                                  <1> move_source_file_to_destination_OK:
25904 00008553 BE[29070100]            <1>       mov    esi, Msg_OK
25905 00008558 E800DEFFFF              <1>       call   print_msg
25906 0000855D E9BDFEFFFF              <1>       jmp    loc_file_rw_restore_retn
25907                                  <1>
25908                                  <1> loc_mkdir_y_n_escape:
25909 00008562 B04E                    <1>       mov    al, 'N' ; 'no'
25910 00008564 EBBF                    <1>       jmp    short loc_do_not_make_directory
25911                                  <1>
25912                                  <1> delete_directory:
25913                                  <1>       ; 15/10/2016
25914                                  <1>       ; 06/03/2016
25915                                  <1>       ; 01/03/2016
25916                                  <1>       ; 29/02/2016
25917                                  <1>       ; 28/02/2016
25918                                  <1>       ; 27/02/2016
25919                                  <1>       ; 26/02/2016 (TRDOS 386 =  TRDOS v2.0)
25920                                  <1>       ; 16/10/2010 (CMD_INTR.ASM, 'cmp_cmd_rmdir')
25921                                  <1>       ; 05/06/2010
25922                                  <1>       ;
25923                                  <1> get_rmdir_fchar:
25924                                  <1>       ; esi = directory name
25925 00008566 803E20                  <1>       cmp    byte [esi], 20h
25926 00008569 7701                    <1>         ja   short loc_rmdir_parse_path_name
25927                                  <1>
25928                                  <1> loc_rmdir_nodirname_retn:
25929 0000856B C3                      <1>       retn
25930                                  <1>
25931                                  <1> loc_rmdir_parse_path_name:
25932 0000856C BF[32580100]            <1>       mov    edi, FindFile_Drv
25933 00008571 E8901C0000              <1>       call   parse_path_name
25934 00008576 0F8271F5FFFF            <1>       jc     loc_cmd_failed
25935                                  <1>
25936                                  <1> loc_rmdir_check_dirname_exists:
25937 0000857C BE[74580100]            <1>       mov    esi, FindFile_Name
25938 00008581 803E20                  <1>       cmp    byte [esi], 20h
25939 00008584 0F8663F5FFFF            <1>       jna    loc_cmd_failed
25940 0000858A 8935[F0580100]          <1>       mov    [DelFile_FNPointer], esi
25941                                  <1>
25942                                  <1> loc_rmdir_drv:
25943 00008590 8A35[8E4E0100]          <1>       mov    dh, [Current_Drv]
25944 00008596 8835[EE560100]          <1>       mov    [RUN_CDRV], dh
25945                                  <1>
25946 0000859C 8A15[32580100]          <1>       mov    dl, [FindFile_Drv]
25947 000085A2 38F2                    <1>       cmp    dl, dh
25948 000085A4 740B                    <1>       je     short loc_rmdir_change_directory
25949                                  <1>
25950 000085A6 E8B2E6FFFF              <1>       call   change_current_drive
25951 000085AB 0F826EFEFFFF            <1>       jc     loc_file_rw_cmd_failed
25952                                  <1>
25953                                  <1> loc_rmdir_change_directory:
25954 000085B1 803D[33580100]20        <1>       cmp    byte [FindFile_Directory], 20h
25955 000085B8 7614                    <1>       jna    short loc_rmdir_find_directory
25956                                  <1>
25957 000085BA FE05[98020100]          <1>       inc    byte [Restore_CDIR]
25958 000085C0 BE[33580100]            <1>       mov    esi, FindFile_Directory
25959 000085C5 30E4                    <1>       xor    ah, ah ; CD_COMMAND sign -> 0
25960 000085C7 E826160000              <1>       call   change_current_directory
25961 000085CC 7211                    <1>       jc     short loc_rmdir_check_error_code
25962                                  <1>
25963                                  <1> ;loc_rmdir_change_prompt_dir_string:
25964                                  <1>       ;call  change_prompt_dir_string
25965                                  <1>
25966                                  <1> loc_rmdir_find_directory:
25967                                  <1>       ;mov   esi, FindFile_Name
25968 000085CE 8B35[F0580100]          <1>       mov    esi, [DelFile_FNPointer]
25969 000085D4 66B81008                <1>       mov    ax, 0810h ; Only directories
25970 000085D8 E812FAFFFF              <1>       call   find_first_file
25971 000085DD 730A                    <1>       jnc    short loc_rmdir_ambgfn_check
25972                                  <1>
25973                                  <1> loc_rmdir_check_error_code:
25974 000085DF 3C02                    <1>       cmp    al, 2
25975 000085E1 740B                    <1>       je     short loc_rmdir_directory_not_found
25976 000085E3 F9                      <1>       stc
25977 000085E4 E936FEFFFF              <1>       jmp    loc_file_rw_cmd_failed
25978                                  <1>
25979                                  <1> loc_rmdir_ambgfn_check:
25980 000085E9 6621D2                  <1>       and    dx, dx ; Ambiguous filename chars used sign (DX>0)
25981 000085EC 740F                    <1>       jz     short loc_rmdir_directory_found
25982                                  <1>
25983                                  <1> loc_rmdir_directory_not_found:
25984 000085EE BE[9A050100]            <1>       mov    esi, Msg_Dir_Not_Found
25985 000085F3 E865DDFFFF              <1>       call   print_msg
25986                                  <1>
25987 000085F8 E922FEFFFF              <1>       jmp    loc_file_rw_restore_retn
25988                                  <1>
25989                                  <1> loc_rmdir_directory_found:
25990 000085FD 80E307                  <1>       and    bl, 07h ; Attributes
25991 00008600 0F8523FEFFFF            <1>       jnz    loc_permission_denied
25992                                  <1>
25993                                  <1> loc_rmdir_save_lnel: ; 28/02/2016
25994                                  <1>        ;mov  bh, [LongName_EntryLength]
25995 00008606 883D[FA580100]          <1>       mov    [DelFile_LNEL], bh ; Long name entry length (if > 0)
25996                                  <1>       ; edi = Directory Entry Offset (DirBuff)
25997                                  <1>       ; esi = Directory Entry (FFF Structure)
25998                                  <1>       ;mov   [DelFile_DirEntryAddr], edi ; not required
25999                                  <1>       ;mov   ax, [edi+20] ; First Cluster High Word
26000                                  <1>         ;shl eax, 16
26001                                  <1>       ;mov   ax, [edi+26] ; First Cluster Low Word
26002                                  <1>       ; ROOT Dir First Cluster = 0
26003                                  <1>         ;cmp eax, 2
26004                                  <1>       ;jb    loc_update_direntry_1
```

```
26005                                  <1>
26006                                  <1> pass_rmdir_fc_check:
26007 0000860C 57                      <1>       push   edi ; * (29/02/2016)
26008                                  <1>
26009 0000860D BE[2F070100]            <1>       mov    esi, Msg_DoYouWantRmDir
26010 00008612 E846DDFFFF              <1>       call   print_msg
26011 00008617 8B35[F0580100]          <1>       mov    esi, [DelFile_FNPointer]
26012 0000861D E83BDDFFFF              <1>       call   print_msg
26013 00008622 BE[1B070100]            <1>       mov    esi, Msg_YesNo
26014 00008627 E831DDFFFF              <1>       call   print_msg
26015                                  <1>
26016                                  <1> loc_rmdir_ask_again:
26017 0000862C 30E4                    <1>       xor    ah, ah
26018 0000862E E8E385FFFF              <1>       call   int16h
26019 00008633 3C1B                    <1>       cmp    al, 1Bh
26020                                  <1>       ;je    short loc_do_not_delete_directory
26021 00008635 0F8498000000            <1>         je      loc_rmdir_y_n_escape ; 06/03/2016
26022 0000863B 24DF                    <1>       and    al, 0DFh
26023 0000863D A2[25070100]            <1>       mov    [Y_N_nextline], al
26024 00008642 3C59                    <1>       cmp    al, 'Y'
26025 00008644 7404                    <1>       je     short loc_rmdir_yes_delete_directory
26026 00008646 3C4E                    <1>       cmp    al, 'N'
26027 00008648 75E2                    <1>       jne    short loc_rmdir_ask_again
26028                                  <1>
26029                                  <1> loc_do_not_delete_directory:
26030                                  <1> loc_rmdir_yes_delete_directory:
26031 0000864A A2[25070100]            <1>       mov    [Y_N_nextline], al
26032 0000864F 6650                    <1>       push   ax
26033 00008651 BE[25070100]            <1>       mov    esi, Y_N_nextline
26034 00008656 E802DDFFFF              <1>       call   print_msg
26035 0000865B 6658                    <1>       pop    ax
26036 0000865D 5F                      <1>       pop    edi ; * (29/02/2016)
26037                                  <1>       ;cmp   al, 'Y' ; 'yes'
26038                                  <1>       ;cmc
26039                                  <1>         ;jnc loc_file_rw_restore_retn
26040 0000865E 3C4E                    <1>       cmp    al, 'N' ; 'no'
26041 00008660 0F84B9FDFFFF            <1>         je   loc_file_rw_restore_retn
26042                                  <1>
26043                                  <1> loc_rmdir_delete_short_name_check_dir_empty:
26044                                  <1>       ; EDI = Directory buffer entry offset/address
26045 00008666 668B4714                <1>       mov    ax, [edi+20] ; First Cluster High Word
26046 0000866A C1E010                  <1>         shl  eax, 16
26047 0000866D 668B471A                <1>       mov    ax, [edi+26] ; First Cluster Low Word
26048                                  <1>
26049 00008671 A3[F4580100]            <1>       mov    [DelFile_FCluster], eax
26050                                  <1>
26051                                  <1>       ;mov   bx, [DirBuff_EntryCounter]
26052 00008676 668B1D[AC580100]        <1>       mov    bx, [FindFile_DirEntryNumber] ; 27/02/2016
26053 0000867D 66891D[F8580100]        <1>       mov    [DelFile_EntryCounter], bx
26054                                  <1>
26055 00008684 29DB                    <1>       sub    ebx, ebx
26056 00008686 8A3D[32580100]          <1>       mov    bh, [FindFile_Drv]
26057 0000868C BE00010900              <1>       mov    esi, Logical_DOSDisks
26058 00008691 01DE                    <1>       add    esi, ebx
26059                                  <1>
26060 00008693 66817F0CA101            <1>       cmp    word [edi+DirEntry_NTRes], 01A1h
26061 00008699 743F                    <1>       je     short loc_rmdir_delete_fs_directory
26062                                  <1>
26063                                  <1>       ;cmp   byte [esi+LD_FATType], 1
26064                                  <1>       ;jnb   short loc_rmdir_get__last_cluster_0
26065                                  <1>       ;mov   eax, 0Bh ; Invalid Format
26066                                  <1>       ;jmp   loc_file_rw_cmd_failed
26067                                  <1>
26068                                  <1> ;loc_rmdir_get_last_cluster_0:
26069 0000869B 8B15[BD560100]          <1>       mov    edx, [DirBuff_Cluster]
26070 000086A1 8915[24590100]          <1>       mov    [RmDir_ParentDirCluster], edx
26071                                  <1>
26072 000086A7 893D[20590100]          <1>       mov    [RmDir_DirEntryOffset], edi
26073                                  <1>
26074                                  <1>       ; 01/03/2016
26075 000086AD C705[AE560100]0000-     <1>       mov    dword [FAT_ClusterCounter], 0 ; Reset
26076 000086B5 0000                    <1>
26077                                  <1>
26078                                  <1> loc_rmdir_get_last_cluster:
26079 000086B7 E85C3A0000              <1>       call   get_last_cluster
26080 000086BC 0F82B8000000            <1>         jc      loc_rmdir_cmd_failed
26081                                  <1>
26082 000086C2 3B05[F4580100]          <1>       cmp    eax, [DelFile_FCluster]
26083 000086C8 752F                    <1>       jne    short loc_rmdir_multi_dir_clusters
26084                                  <1>
26085 000086CA C605[1F590100]00        <1>       mov    byte [RmDir_MultiClusters], 0
26086 000086D1 EB2D                    <1>       jmp    short pass_rmdir_multi_dir_clusters
26087                                  <1>
26088                                  <1> loc_rmdir_y_n_escape:
26089 000086D3 B04E                    <1>       mov    al, 'N' ; 'no'
26090 000086D5 E970FFFFFF              <1>         jmp     loc_do_not_delete_directory
26091                                  <1>
26092                                  <1> loc_rmdir_delete_fs_directory:
26093 000086DA 807E04A1                <1>       cmp    byte [esi+LD_FSType], 0A1h
26094 000086DE 0F8545FDFFFF            <1>       jne    loc_permission_denied
26095                                  <1>
26096 000086E4 E826140000              <1>       call   delete_fs_directory
26097 000086E9 0F8326FDFFFF            <1>       jnc    loc_print_deleted_message
26098                                  <1>
26099 000086EF 09C0                    <1>       or     eax, eax
26100 000086F1 745D                    <1>       jz     loc_rmdir_directory_not_empty_2
26101 000086F3 F9                      <1>       stc
26102 000086F4 E926FDFFFF              <1>       jmp    loc_file_rw_cmd_failed
26103                                  <1>
26104                                  <1> loc_rmdir_multi_dir_clusters:
26105 000086F9 C605[1F590100]01        <1>       mov    byte [RmDir_MultiClusters], 1
26106                                  <1>
```

```
26107                                  <1> pass_rmdir_multi_dir_clusters:
26108 00008700 A3[28590100]            <1>       mov   [RmDir_DirLastCluster], eax
26109 00008705 890D[2C590100]          <1>       mov   [RmDir_PreviousCluster], ecx
26110                                  <1>
26111                                  <1> loc_rmdir_load_fat_sub_directory:
26112 0000870B E8F3330000              <1>       call  load_FAT_sub_directory
26113 00008710 7268                    <1>       jc    loc_rmdir_cmd_failed
26114                                  <1>
26115                                  <1> loc_rmdir_find_last_dir_entry:
26116 00008712 56                      <1>       push  esi
26117 00008713 BE[16580100]            <1>       mov   esi, Dir_File_Name
26118 00008718 C6062A                  <1>       mov   byte [esi], '*'
26119 0000871B C646082A                <1>       mov   byte [esi+8], '*'
26120 0000871F 31DB                    <1>       xor   ebx, ebx ; Entry offset  = 0
26121                                  <1> loc_rmdir_find_last_dir_entry_next:
26122 00008721 66B80008                <1>       mov   ax, 0800h ; Except volume/long names
26123 00008725 6631C9                  <1>       xor   cx, cx ; 0 = Find a valid file or dir name
26124 00008728 E81B180000              <1>       call  find_directory_entry
26125 0000872D 7271                    <1>       jc    short loc_rmdir_empty_dir_cluster
26126 0000872F 83FB01                  <1>       cmp   ebx, 1
26127 00008732 771B                    <1>       ja    short loc_rmdir_directory_not_empty_1
26128                                  <1> loc_rmdir_dot_entry_check:
26129 00008734 80FD2E                  <1>       cmp   ch, '.' ; The first char of the dir entry
26130 00008737 7516                    <1>       jne   short loc_rmdir_directory_not_empty_1
26131 00008739 08DB                    <1>       or    bl, bl
26132 0000873B 7506                    <1>       jnz   short loc_rmdir_dotdot_entry_check
26133 0000873D 807F0120                <1>       cmp   byte [edi+1], 20h
26134 00008741 EB06                    <1>       jmp   short pass_rmdir_dot_entry_check
26135                                  <1>
26136                                  <1> loc_rmdir_dotdot_entry_check:
26137 00008743 66817F012E20            <1>       cmp   word [edi+1], '. '
26138                                  <1> pass_rmdir_dot_entry_check:
26139 00008749 7504                    <1>       jne   short loc_rmdir_directory_not_empty_1
26140 0000874B FEC3                    <1>       inc   bl
26141 0000874D EBD2                    <1>       jmp   short loc_rmdir_find_last_dir_entry_next
26142                                  <1>
26143                                  <1>
26144                                  <1> loc_rmdir_directory_not_empty_1:
26145 0000874F 58                      <1>       pop   eax ; pushed esi
26146                                  <1>
26147                                  <1> loc_rmdir_directory_not_empty_2:
26148 00008750 BE[50070100]            <1>       mov   esi, Msg_Dir_Not_Empty
26149 00008755 E803DCFFFF              <1>       call  print_msg
26150                                  <1>       ; 01/03/2016
26151 0000875A A1[AE560100]            <1>       mov   eax, [FAT_ClusterCounter]
26152 0000875F 09C0                    <1>       or    eax, eax ; 0 ?
26153 00008761 0F84B8FCFFFF            <1>       jz    loc_file_rw_restore_retn
26154                                  <1>       ; ESI = Logical DOS Drive Description Table address
26155                                  <1>
26156 00008767 66BB01FF                <1>       mov   bx, 0FF01h ; BH = FFh -> use ESI for Drive parameters
26157                                  <1>                        ; BL = 1 -> add free clusters
26158 0000876B E829380000              <1>       call  calculate_fat_freespace
26159 00008770 09C9                    <1>       or    ecx, ecx
26160 00008772 0F84A7FCFFFF            <1>         jz      loc_file_rw_restore_retn ; ecx = 0 -> OK
26161                                  <1>       ; ecx > 0 -> Error (Recalculation is neeeded)
26162 00008778 EB0E                    <1>       jmp   short loc_rmdir_cmd_return
26163                                  <1>
26164                                  <1>
26165                                  <1> loc_rmdir_cmd_failed:
26166 0000877A 833D[AE560100]01        <1>       cmp   dword [FAT_ClusterCounter], 1
26167 00008781 0F8298FCFFFF            <1>       jb    loc_file_rw_cmd_failed
26168 00008787 F9                      <1>       stc
26169                                  <1> loc_rmdir_cmd_return:
26170                                  <1>       ; 01/03/2016
26171 00008788 9C                      <1>       pushf
26172                                  <1>       ; ESI = Logical DOS Drive Description Table address
26173 00008789 66BB00FF                <1>       mov   bx, 0FF00h ; BH = FFh -> use ESI for Drive parameters
26174                                  <1>                        ; BL = 0 -> Recalculate free cluster count
26175 0000878D 50                      <1>       push  eax
26176 0000878E E806380000              <1>       call  calculate_fat_freespace
26177 00008793 58                      <1>       pop   eax
26178 00008794 9D                      <1>       popf
26179 00008795 0F8284FCFFFF            <1>       jc    loc_file_rw_cmd_failed
26180 0000879B E97FFCFFFF              <1>       jmp   loc_file_rw_restore_retn
26181                                  <1>
26182                                  <1>
26183                                  <1> loc_rmdir_empty_dir_cluster:
26184 000087A0 5E                      <1>       pop   esi
26185                                  <1>
26186                                  <1> loc_rmdir_set_prev_cluster_dir_last_cluster:
26187 000087A1 803D[1F590100]00        <1>       cmp   byte [RmDir_MultiClusters], 0
26188 000087A8 761D                    <1>       jna   short loc_rmdir_unlink_dir_last_cluster
26189                                  <1>
26190 000087AA A1[2C590100]            <1>       mov   eax, [RmDir_PreviousCluster]
26191                                  <1>       ;xor  ecx, ecx
26192 000087AF 49                      <1>       dec   ecx ; FFFFFFFFh
26193 000087B0 E892340000              <1>       call  update_cluster
26194 000087B5 7310                    <1>       jnc   short loc_rmdir_unlink_dir_last_cluster
26195                                  <1>
26196                                  <1> loc_rmdir_unlink_stc_retn:
26197                                  <1>       ; 01/03/2016
26198 000087B7 83F801                  <1>       cmp   eax, 1  ; eax = 0 -> end of cluster chain
26199 000087BA F5                      <1>       cmc
26200 000087BB 72BD                    <1>       jc    short loc_rmdir_cmd_failed
26201 000087BD EB1D                    <1>       jmp   short loc_rmdir_save_fat_buffer
26202                                  <1>
26203                                  <1> loc_rmdir_unlink_stc_retn_0Bh:
26204 000087BF B81C000000              <1>       mov   eax, 28 ; Invalid format ; 15/10/2016
26205 000087C4 F9                      <1>       stc
26206 000087C5 EBB3                    <1>       jmp   short loc_rmdir_cmd_failed
26207                                  <1>
26208                                  <1> loc_rmdir_unlink_dir_last_cluster:
```

```
26209 000087C7 A1[28590100]        <1>       mov    eax, [RmDir_DirLastCluster]
26210 000087CC 31C9                <1>       xor    ecx, ecx ; 0
26211 000087CE E874340000          <1>       call   update_cluster
26212 000087D3 73EA                <1>       jnc    short loc_rmdir_unlink_stc_retn_0Bh
26213                              <1>       ; Because of it is the last cluster
26214                              <1>       ; 'update_cluster' must return with eocc error
26215 000087D5 09C0                <1>       or     eax, eax
26216 000087D7 7403                <1>       jz     short loc_rmdir_save_fat_buffer ; eocc
26217 000087D9 F9                  <1>       stc
26218 000087DA EB9E                <1>         jmp     short loc_rmdir_cmd_failed
26219                              <1>
26220                              <1> loc_rmdir_save_fat_buffer:
26221 000087DC 803D[A6560100]02    <1>       cmp    byte [FAT_BuffValidData], 2
26222 000087E3 7525                <1>       jne    short loc_rmdir_calculate_FAT_freespace
26223 000087E5 E81A370000          <1>       call   save_fat_buffer
26224 000087EA 728E                <1>       jc     short loc_rmdir_cmd_failed
26225                              <1>
26226                              <1>       ; 01/03/2016
26227 000087EC 803D[1F590100]00    <1>       cmp    byte [RmDir_MultiClusters], 0
26228 000087F3 7615                <1>       jna    short loc_rmdir_calculate_FAT_freespace
26229                              <1>
26230 000087F5 A1[F4580100]        <1>       mov    eax, [DelFile_FCluster]
26231 000087FA E9B8FEFFFF          <1>         jmp     loc_rmdir_get_last_cluster
26232                              <1>
26233                              <1> loc_rmdir_delete_short_name_invalid_data:
26234 000087FF B81D000000          <1>       mov    eax, 29 ; Invalid data (15/10/2016)
26235 00008804 F9                  <1>       stc
26236 00008805 E970FFFFFF          <1>         jmp     loc_rmdir_cmd_failed
26237                              <1>
26238                              <1> loc_rmdir_calculate_FAT_freespace:
26239 0000880A A1[AE560100]        <1>       mov    eax, [FAT_ClusterCounter]
26240 0000880F 66BB01FF            <1>       mov    bx, 0FF01h
26241                              <1>       ; BL = 1 -> Add EAX to free space count
26242                              <1>       ; BH = FFh ->
26243                              <1>       ; ESI = Logical DOS Drive Description Table address
26244 00008813 E881370000          <1>       call   calculate_fat_freespace
26245                              <1>
26246 00008818 21C9                <1>       and    ecx, ecx ; ecx = 0 -> valid free sector count
26247 0000881A 7409                <1>       jz     short loc_rmdir_delete_short_name_continue
26248                              <1>
26249                              <1> loc_rmdir_recalculate_FAT_freespace:
26250 0000881C 66BB00FF            <1>         mov     bx, 0FF00h ; BL = 0 -> Recalculate free space
26251 00008820 E874370000          <1>       call   calculate_fat_freespace
26252                              <1>
26253                              <1> loc_rmdir_delete_short_name_continue:
26254 00008825 A1[24590100]        <1>       mov    eax, [RmDir_ParentDirCluster]
26255 0000882A 83F802              <1>       cmp    eax, 2
26256 0000882D 730D                <1>       jnb    short loc_rmdir_del_short_name_load_sub_dir
26257 0000882F E844320000          <1>       call   load_FAT_root_directory
26258 00008834 0F82E5FBFFFF        <1>       jc     loc_file_rw_cmd_failed
26259 0000883A EB0B                <1>       jmp    short loc_rmdir_del_short_name_ld_chk_fclust
26260                              <1>
26261                              <1> loc_rmdir_del_short_name_load_sub_dir:
26262 0000883C E8C2320000          <1>       call   load_FAT_sub_directory
26263 00008841 0F82D8FBFFFF        <1>       jc     loc_file_rw_cmd_failed
26264                              <1>
26265                              <1> loc_rmdir_del_short_name_ld_chk_fclust:
26266 00008847 0FB73D[20590100]    <1>       movzx  edi, word [RmDir_DirEntryOffset]
26267 0000884E 81C700000800        <1>       add    edi, Directory_Buffer
26268                              <1>
26269 00008854 668B4714            <1>       mov    ax, [edi+20] ; First Cluster High Word
26270 00008858 C1E010              <1>       shl    eax, 16
26271 0000885B 668B471A            <1>       mov    ax, [edi+26] ; First Cluster Low Word
26272                              <1>         ; Not necessary...
26273 0000885F 3B05[F4580100]      <1>       cmp    eax, [DelFile_FCluster]
26274 00008865 7598                <1>       jne    short loc_rmdir_delete_short_name_invalid_data
26275                              <1>       ;
26276 00008867 C607E5              <1>       mov    byte [edi], 0E5h ; 'Deleted' sign
26277                              <1>       ; 27/02/2016
26278                              <1>       ; TRDOS v1 has a bug here! it does not set
26279                              <1>       ; 'DirBuff_ValidData' to 2; as result of this bug,
26280                              <1>       ; 'save_directory_buffer' would not save the change !
26281 0000886A C605[B8560100]02    <1>       mov    byte [DirBuff_ValidData], 2 ; change sign
26282                              <1>       ;
26283 00008871 E8F41D0000          <1>       call   save_directory_buffer
26284 00008876 0F82A3FBFFFF        <1>       jc     loc_file_rw_cmd_failed
26285                              <1>
26286                              <1> loc_rmdir_del_long_name:
26287 0000887C 0FB615[FA580100]    <1>       movzx  edx, byte [DelFile_LNEL]
26288 00008883 08D2                <1>       or     dl, dl
26289 00008885 7414                <1>       jz     short loc_rmdir_update_parent_dir_lmdt
26290                              <1>
26291 00008887 0FB705[F8580100]    <1>       movzx  eax, word [DelFile_EntryCounter]
26292 0000888E 29D0                <1>       sub    eax, edx
26293 00008890 0F8289FBFFFF        <1>       jc     loc_file_rw_cmd_failed
26294                              <1>
26295                              <1>       ; EAX = Directory Entry Number of the long name last entry
26296 00008896 E82F1F0000          <1>       call   delete_longname
26297                              <1>       ;jc    short loc_file_rw_cmd_failed
26298                              <1>
26299                              <1> loc_rmdir_update_parent_dir_lmdt:
26300 0000889B E8651E0000          <1>       call   update_parent_dir_lmdt
26301                              <1>       ;jc    short loc_file_rw_cmd_failed
26302                              <1>
26303                              <1> loc_rmdir_ok:
26304 000088A0 BE[29070100]        <1>       mov    esi, Msg_OK
26305 000088A5 E8B3DAFFFF          <1>       call   print_msg
26306 000088AA E970FBFFFF          <1>       jmp    loc_file_rw_restore_retn
26307                              <1>
26308                              <1>
26309                              <1> delete_file:
26310                              <1>       ; 29/02/2016
```

```
26311                                  <1>       ; 28/02/2016 (TRDOS 386 =  TRDOS v2.0)
26312                                  <1>       ; 09/08/2010 (CMD_INTR.ASM, 'cmp_cmd_del')
26313                                  <1>       ; 28/02/2010
26314                                  <1>
26315                                  <1> get_delfile_fchar:
26316                                  <1>       ; esi = file name
26317 000088AF 803E20                 <1>       cmp    byte [esi], 20h
26318 000088B2 7701                   <1>         ja   short loc_delfile_parse_path_name
26319                                  <1>
26320                                  <1> loc_delfile_nofilename_retn:
26321 000088B4 C3                     <1>       retn
26322                                  <1>
26323                                  <1> loc_delfile_parse_path_name:
26324 000088B5 BF[32580100]           <1>       mov    edi, FindFile_Drv
26325 000088BA E847190000             <1>       call   parse_path_name
26326 000088BF 0F8228F2FFFF           <1>       jc     loc_cmd_failed
26327                                  <1>
26328                                  <1> loc_delfile_check_filename_exists:
26329 000088C5 BE[74580100]           <1>       mov    esi, FindFile_Name
26330 000088CA 803E20                 <1>       cmp    byte [esi], 20h
26331 000088CD 0F861AF2FFFF           <1>       jna    loc_cmd_failed
26332 000088D3 8935[F0580100]         <1>       mov    [DelFile_FNPointer], esi
26333                                  <1>
26334                                  <1> loc_delfile_drv:
26335 000088D9 8A15[32580100]         <1>       mov    dl, [FindFile_Drv]
26336 000088DF 8A35[8E4E0100]         <1>       mov    dh, [Current_Drv]
26337 000088E5 8835[EE560100]         <1>       mov    [RUN_CDRV], dh
26338 000088EB 38F2                   <1>       cmp    dl, dh
26339 000088ED 740B                   <1>       je     short loc_delfile_change_directory
26340                                  <1>
26341 000088EF E869E3FFFF             <1>       call   change_current_drive
26342 000088F4 0F8225FBFFFF           <1>       jc     loc_file_rw_cmd_failed
26343                                  <1>
26344                                  <1> loc_delfile_change_directory:
26345 000088FA 803D[33580100]20       <1>       cmp    byte [FindFile_Directory], 20h
26346 00008901 7618                   <1>       jna    short loc_delfile_find
26347                                  <1>
26348 00008903 FE05[98020100]         <1>       inc    byte [Restore_CDIR]
26349 00008909 BE[33580100]           <1>       mov    esi, FindFile_Directory
26350 0000890E 30E4                   <1>       xor    ah, ah ; CD_COMMAND sign -> 0
26351 00008910 E8DD120000             <1>       call   change_current_directory
26352 00008915 0F8204FBFFFF           <1>       jc     loc_file_rw_cmd_failed
26353                                  <1>
26354                                  <1> ;loc_delfile_change_prompt_dir_string:
26355                                  <1>       ;call  change_prompt_dir_string
26356                                  <1>
26357                                  <1> loc_delfile_find:
26358                                  <1>       ;mov   esi, FindFile_Name
26359 0000891B 8B35[F0580100]         <1>       mov    esi, [DelFile_FNPointer]
26360 00008921 66B80018               <1>       mov    ax, 1800h ; Except volume label and dirs
26361 00008925 E8C5F6FFFF             <1>       call   find_first_file
26362 0000892A 0F82EFFAFFFF           <1>       jc     loc_file_rw_cmd_failed
26363                                  <1>
26364                                  <1> loc_delfile_ambgfn_check:
26365 00008930 6621D2                 <1>       and    dx, dx ; Ambiguous filename chars used sign (DX>0)
26366 00008933 740B                   <1>       jz     short loc_delfile_found
26367                                  <1>
26368                                  <1> loc_file_not_found:
26369 00008935 B802000000             <1>       mov    eax, 2 ; File not found sign
26370 0000893A F9                     <1>       stc
26371 0000893B E9DFFAFFFF             <1>       jmp    loc_file_rw_cmd_failed
26372                                  <1>
26373                                  <1> loc_delfile_found:
26374 00008940 80E307                 <1>       and    bl, 07h ; Attributes
26375 00008943 0F85E0FAFFFF           <1>         jnz     loc_permission_denied
26376                                  <1>
26377                                  <1> ;loc_delfile_found_save_lnel:
26378                                  <1> ;     mov    [DelFile_LNEL], bh ; Long name entry length (if > 0)
26379                                  <1>
26380                                  <1> loc_delfile_ask_for_delete:
26381 00008949 57                     <1>       push   edi ; * (29/02/2016)
26382                                  <1>
26383 0000894A BE[67070100]           <1>       mov    esi, Msg_DoYouWantDelete
26384 0000894F E809DAFFFF             <1>       call   print_msg
26385 00008954 8B35[F0580100]         <1>       mov    esi, [DelFile_FNPointer]
26386 0000895A E8FED9FFFF             <1>       call   print_msg
26387 0000895F BE[1B070100]           <1>       mov    esi, Msg_YesNo
26388 00008964 E8F4D9FFFF             <1>       call   print_msg
26389                                  <1>
26390                                  <1> loc_delfile_ask_again:
26391 00008969 30E4                   <1>       xor    ah, ah
26392 0000896B E8A682FFFF             <1>       call   int16h
26393 00008970 3C1B                   <1>       cmp    al, 1Bh
26394                                  <1>       ;je    short loc_do_not_delete_file
26395 00008972 7457                   <1>       je     short loc_delfile_y_n_escape ; 06/03/2016
26396 00008974 24DF                   <1>       and    al, 0DFh
26397 00008976 A2[25070100]           <1>       mov    [Y_N_nextline], al
26398 0000897B 3C59                   <1>       cmp    al, 'Y'
26399 0000897D 7404                   <1>       je     short loc_yes_delete_file
26400 0000897F 3C4E                   <1>       cmp    al, 'N'
26401 00008981 75E6                   <1>       jne    short loc_delfile_ask_again
26402                                  <1>
26403                                  <1> loc_do_not_delete_file:
26404                                  <1> loc_yes_delete_file:
26405 00008983 A2[25070100]           <1>       mov    [Y_N_nextline], al
26406 00008988 6650                   <1>       push   ax
26407 0000898A BE[25070100]           <1>       mov    esi, Y_N_nextline
26408 0000898F E8C9D9FFFF             <1>       call   print_msg
26409 00008994 6658                   <1>       pop    ax
26410 00008996 5F                     <1>       pop    edi ; * (29/02/2016)
26411                                  <1>       ;cmp   al, 'Y' ; 'yes'
26412                                  <1>       ;cmc
```

```
26413                                  <1>         ;jnc loc_file_rw_restore_retn
26414 00008997 3C4E                    <1>       cmp   al, 'N' ; 'no'
26415 00008999 0F8480FAFFFF            <1>         je  loc_file_rw_restore_retn
26416                                  <1>
26417                                  <1> loc_delete_file:
26418 0000899F 8A3D[32580100]          <1>       mov   bh, [FindFile_Drv]
26419                                  <1>       ;mov  bl, [DelFile_LNEL]
26420 000089A5 8A1D[81580100]          <1>       mov   bl, [FindFile_LongNameEntryLength]
26421                                  <1>       ;mov  cx, [DirBuff_EntryCounter]
26422 000089AB 668B0D[AC580100]        <1>       mov   cx, [FindFile_DirEntryNumber]
26423                                  <1>       ; (*) EDI = Directory buffer entry offset/address
26424 000089B2 E8FD1F0000              <1>       call  remove_file ; (FILE.ASM, 'proc_delete_file')
26425 000089B7 0F8358FAFFFF            <1>       jnc   loc_print_deleted_message
26426                                  <1>
26427 000089BD 3C05                    <1>       cmp   al, 05h
26428 000089BF 0F8464FAFFFF            <1>       je    loc_permission_denied
26429 000089C5 F9                      <1>       stc
26430 000089C6 E954FAFFFF              <1>       jmp   loc_file_rw_cmd_failed
26431                                  <1>
26432                                  <1> loc_delfile_y_n_escape:
26433 000089CB B04E                    <1>       mov   al, 'N' ; 'no'
26434 000089CD EBB4                    <1>       jmp   short loc_do_not_delete_file
26435                                  <1>
26436                                  <1> set_file_attributes:
26437                                  <1>       ; 06/03/2016
26438                                  <1>       ; 04/03/2016 (TRDOS 386 =  TRDOS v2.0)
26439                                  <1>       ; 10/07/2010 (TRDOS v1, CMD_INTR.ASM, 'cmp_cmd_attrib')
26440                                  <1>       ; 23/05/2010
26441                                  <1>       ; 17/12/2000 (P2000.ASM)
26442                                  <1>
26443                                  <1>       ; esi = file or directory name
26444 000089CF 6631C0                  <1>       xor   ax, ax
26445 000089D2 66A3[B8070100]          <1>       mov   [Attr_Chars], ax
26446 000089D8 A2[48590100]            <1>       mov   [Attributes], al
26447                                  <1>
26448                                  <1> get_attrib_fchar:
26449                                  <1>       ; esi = file name
26450 000089DD 8A06                    <1>       mov   al, [esi]
26451 000089DF 3C20                    <1>       cmp   al, 20h
26452 000089E1 7623                    <1>       jna   short loc_attr_file_nofilename_retn
26453                                  <1>
26454                                  <1> loc_scan_attrib_params:
26455 000089E3 3C2D                    <1>       cmp   al, '-'
26456 000089E5 0F871C010000            <1>       ja    loc_attr_file_parse_path_name
26457 000089EB 7408                    <1>       je    short loc_attr_space
26458                                  <1>
26459 000089ED 3C2B                    <1>       cmp   al, '+'
26460 000089EF 0F85F8F0FFFF            <1>       jne   loc_cmd_failed
26461                                  <1>
26462                                  <1> loc_attr_space:
26463 000089F5 8A6601                  <1>       mov   ah, [esi+1]
26464 000089F8 80FC20                  <1>       cmp   ah, 20h
26465 000089FB 770A                    <1>       ja    short pass_attr_space
26466 000089FD 0F82EAF0FFFF            <1>       jb    loc_cmd_failed
26467 00008A03 46                      <1>       inc   esi
26468 00008A04 EBEF                    <1>       jmp   short loc_attr_space
26469                                  <1>
26470                                  <1> loc_attr_file_nofilename_retn:
26471 00008A06 C3                      <1>       retn
26472                                  <1>
26473                                  <1> pass_attr_space:
26474 00008A07 80E4DF                  <1>       and   ah, 0DFh
26475 00008A0A 80FC53                  <1>       cmp   ah, 'S'
26476 00008A0D 0F87DAF0FFFF            <1>       ja    loc_cmd_failed
26477 00008A13 7204                    <1>       jb    short pass_attr_system
26478 00008A15 B404                    <1>       mov   ah, 04h   ; System
26479 00008A17 EB21                    <1>       jmp   short pass_attr_archive
26480                                  <1>
26481                                  <1> pass_attr_system:
26482 00008A19 80FC48                  <1>       cmp   ah, 'H'
26483 00008A1C 7706                    <1>       ja    short pass_attr_hidden
26484 00008A1E 7213                    <1>       jb    short pass_attr_read_only
26485 00008A20 B402                    <1>       mov   ah, 02h    ; Hidden
26486 00008A22 EB16                    <1>       jmp   short pass_attr_archive
26487                                  <1>
26488                                  <1> pass_attr_hidden:
26489 00008A24 80FC52                  <1>       cmp   ah, 'R'
26490 00008A27 0F87C0F0FFFF            <1>       ja    loc_cmd_failed
26491 00008A2D 7204                    <1>       jb    short pass_attr_read_only ; Read only
26492 00008A2F B401                    <1>       mov   ah, 01h
26493 00008A31 EB07                    <1>       jmp   short pass_attr_archive
26494                                  <1>
26495                                  <1> pass_attr_read_only:
26496 00008A33 80FC41                  <1>       cmp   ah, 'A'
26497 00008A36 753B                    <1>       jne   short loc_chk_attr_enter
26498 00008A38 B420                    <1>       mov   ah, 20h    ; Archive
26499                                  <1>
26500                                  <1> pass_attr_archive:
26501 00008A3A 3C2D                    <1>       cmp   al, '-'
26502 00008A3C 7508                    <1>       jne   short pass_reducing_attributes
26503 00008A3E 0825[B8070100]          <1>       or    [Attr_Chars], ah
26504 00008A44 EB06                    <1>       jmp   short loc_change_attributes_inc
26505                                  <1>
26506                                  <1> pass_reducing_attributes:
26507 00008A46 0825[B9070100]          <1>       or    [Attr_Chars+1], ah
26508                                  <1>
26509                                  <1> loc_change_attributes_inc:
26510 00008A4C 46                      <1>       inc   esi
26511 00008A4D 8A6601                  <1>       mov   ah, [esi+1]
26512 00008A50 80FC20                  <1>       cmp   ah, 20h
26513 00008A53 7227                    <1>       jb    short pass_change_attr
26514 00008A55 74F5                    <1>       je    short loc_change_attributes_inc
```

```
26515 00008A57 80FC2D              <1>       cmp    ah, '-'
26516 00008A5A 770D                <1>       ja     short loc_chk_next_attr_char1
26517 00008A5C 7405                <1>       je     short loc_chk_next_attr_char0
26518 00008A5E 80FC2B              <1>       cmp    ah, '+'
26519 00008A61 7506                <1>       jne    short loc_chk_next_attr_char1
26520                              <1>
26521                              <1> loc_chk_next_attr_char0:
26522 00008A63 46                  <1>       inc    esi
26523 00008A64 668B06              <1>       mov    ax, [esi]
26524 00008A67 EB9E                <1>       jmp    short pass_attr_space
26525                              <1>
26526                              <1> loc_chk_next_attr_char1:
26527 00008A69 803E2D              <1>       cmp    byte [esi], '-'
26528 00008A6C 7799                <1>       ja     short pass_attr_space
26529 00008A6E E988000000          <1>         jmp     loc_attr_file_check_fname_fchar
26530                              <1>
26531                              <1> loc_chk_attr_enter:
26532 00008A73 80FC0D              <1>       cmp    ah, 0Dh
26533 00008A76 0F8571F0FFFF        <1>       jne    loc_cmd_failed
26534                              <1>
26535                              <1> pass_change_attr:
26536 00008A7C A0[B8070100]        <1>       mov    al, [Attr_Chars]
26537 00008A81 F6D0                <1>       not    al
26538 00008A83 2005[48590100]      <1>       and    [Attributes], al
26539 00008A89 A0[B9070100]        <1>       mov    al, [Attr_Chars+1]
26540 00008A8E 0805[48590100]      <1>       or     [Attributes], al
26541                              <1>
26542                              <1> loc_show_attributes:
26543 00008A94 BE[030F0100]        <1>       mov    esi, nextline
26544 00008A99 E8BFD8FFFF          <1>       call   print_msg
26545                              <1>
26546                              <1> loc_show_attributes_no_nextline:
26547 00008A9E C705[B8070100]4E4F- <1>       mov    dword [Attr_Chars], 'NORM'
26548 00008AA6 524D                <1>
26549 00008AA8 66C705[BC070100]41- <1>       mov    word [Attr_Chars+4], 'AL'
26550 00008AB0 4C                  <1>
26551 00008AB1 BE[B8070100]        <1>       mov    esi, Attr_Chars
26552 00008AB6 A0[48590100]        <1>       mov    al, [Attributes]
26553 00008ABB A804                <1>       test   al, 04h
26554 00008ABD 7406                <1>       jz     short pass_put_attr_s
26555 00008ABF 66C7065300          <1>       mov    word [esi], 0053h     ; S
26556 00008AC4 46                  <1>       inc    esi
26557                              <1>
26558                              <1> pass_put_attr_s:
26559 00008AC5 A802                <1>       test   al, 02h
26560 00008AC7 7406                <1>       jz     short pass_put_attr_h
26561 00008AC9 66C7064800          <1>       mov    word [esi], 0048h     ; H
26562 00008ACE 46                  <1>       inc    esi
26563                              <1>
26564                              <1> pass_put_attr_h:
26565 00008ACF A801                <1>       test   al, 01h
26566 00008AD1 7406                <1>       jz     short pass_put_attr_r
26567 00008AD3 66C7065200          <1>       mov    word [esi], 0052h     ; R
26568 00008AD8 46                  <1>       inc    esi
26569                              <1>
26570                              <1> pass_put_attr_r:
26571 00008AD9 3C20                <1>       cmp    al, 20h
26572 00008ADB 7205                <1>       jb     short pass_put_attr_a
26573 00008ADD 66C7064100          <1>       mov    word [esi], 0041h     ; A
26574                              <1>
26575                              <1> pass_put_attr_a:
26576 00008AE2 BE[AB070100]        <1>       mov    esi, Str_Attributes
26577 00008AE7 E871D8FFFF          <1>       call   print_msg
26578 00008AEC BE[030F0100]        <1>       mov    esi, nextline
26579 00008AF1 E867D8FFFF          <1>       call   print_msg
26580 00008AF6 E924F9FFFF          <1>       jmp    loc_file_rw_restore_retn
26581                              <1>
26582                              <1> loc_attr_file_check_fname_fchar:
26583 00008AFB 46                  <1>       inc    esi
26584 00008AFC 803E20              <1>       cmp    byte [esi], 20h
26585 00008AFF 74FA                <1>       je     short loc_attr_file_check_fname_fchar
26586 00008B01 0F8275FFFFFF        <1>         jb      pass_change_attr
26587                              <1>
26588                              <1> loc_attr_file_parse_path_name:
26589 00008B07 BF[32580100]        <1>       mov    edi, FindFile_Drv
26590 00008B0C E8F5160000          <1>       call   parse_path_name
26591 00008B11 0F82D6EFFFFF        <1>       jc     loc_cmd_failed
26592                              <1>
26593                              <1> loc_attr_file_check_filename_exists:
26594 00008B17 BE[74580100]        <1>       mov    esi, FindFile_Name
26595 00008B1C 803E20              <1>       cmp    byte [esi], 20h
26596 00008B1F 0F86C8EFFFFF        <1>       jna    loc_cmd_failed
26597 00008B25 8935[F0580100]      <1>       mov    [DelFile_FNPointer], esi
26598                              <1>
26599                              <1> loc_attr_file_drv:
26600 00008B2B 8A35[8E4E0100]      <1>       mov    dh, [Current_Drv]
26601 00008B31 8835[EE560100]      <1>       mov    [RUN_CDRV], dh
26602                              <1>
26603 00008B37 8A15[32580100]      <1>       mov    dl, [FindFile_Drv]
26604 00008B3D 38F2                <1>       cmp    dl, dh
26605 00008B3F 740B                <1>       je     short loc_attr_file_change_directory
26606                              <1>
26607 00008B41 E817E1FFFF          <1>       call   change_current_drive
26608 00008B46 0F82D3F8FFFF        <1>       jc     loc_file_rw_cmd_failed
26609                              <1>
26610                              <1> loc_attr_file_change_directory:
26611 00008B4C 803D[33580100]20    <1>         cmp     byte [FindFile_Directory], 20h
26612 00008B53 7618                <1>       jna    short loc_attr_file_find
26613                              <1>
26614 00008B55 FE05[98020100]      <1>       inc    byte [Restore_CDIR]
26615                              <1>
26616 00008B5B BE[33580100]        <1>       mov    esi, FindFile_Directory
```

```
26617 00008B60 30E4                <1>       xor    ah, ah ; CD_COMMAND sign -> 0
26618 00008B62 E88B100000          <1>       call   change_current_directory
26619 00008B67 0F82B2F8FFFF        <1>       jc     loc_file_rw_cmd_failed
26620                              <1>
26621                              <1> ;loc_attr_file_change_prompt_dir_string:
26622                              <1>       ;call  change_prompt_dir_string
26623                              <1>
26624                              <1> loc_attr_file_find:
26625                              <1>       ;mov   esi, FindFile_Name
26626 00008B6D 8B35[F0580100]      <1>       mov    esi, [DelFile_FNPointer]
26627 00008B73 66B80008            <1>       mov    ax, 0800h ; Except volume labels
26628 00008B77 E873F4FFFF          <1>       call   find_first_file
26629 00008B7C 0F829DF8FFFF        <1>       jc     loc_file_rw_cmd_failed
26630                              <1>
26631                              <1> loc_attr_file_ambgfn_check:
26632 00008B82 6609D2              <1>       or     dx, dx ; Ambiguous filename chars used sign (DX>0)
26633                              <1>       ;      (Note: It was BX in TRDOS v1)
26634                              <1>       ;jz    short loc_attr_file_found
26635 00008B85 0F85AAFDFFFF        <1>         jnz     loc_file_not_found ; 06/03/2016
26636                              <1>
26637                              <1>       ;mov   eax, 2 ; File not found sign
26638                              <1>       ;stc
26639                              <1>       ;jmp   loc_file_rw_cmd_failed
26640                              <1>
26641                              <1> loc_attr_file_found:
26642                              <1>       ; EDI = Directory buffer entry offset/address
26643                              <1>       ; BL = File (or Directory) Attributes
26644                              <1>       ;      (Note: It was 'CL' in TRDOS v1)
26645                              <1>       ; mov  bl, [EDI+0Bh]
26646                              <1>
26647 00008B8B 66833D[B8070100]00  <1>       cmp    word [Attr_Chars], 0
26648 00008B93 770B                <1>       ja     short loc_attr_file_change_attributes
26649 00008B95 881D[48590100]      <1>       mov    [Attributes], bl
26650 00008B9B E9F4FEFFFF          <1>       jmp    loc_show_attributes
26651                              <1>
26652                              <1> loc_attr_file_change_attributes:
26653 00008BA0 A0[B8070100]        <1>       mov    al, [Attr_Chars]
26654 00008BA5 F6D0                <1>       not    al
26655 00008BA7 20C3                <1>       and    bl, al
26656 00008BA9 A0[B9070100]        <1>       mov    al, [Attr_Chars+1]
26657 00008BAE 08C3                <1>       or     bl, al
26658                              <1>
26659 00008BB0 66817F0CA101        <1>       cmp    word [edi+DirEntry_NTRes], 01A1h ; Singlix FS
26660 00008BB6 741D                <1>       je     short loc_attr_file_fs_check
26661                              <1>
26662 00008BB8 881D[48590100]      <1>       mov    [Attributes], bl
26663 00008BBE 885F0B              <1>       mov    [edi+0Bh], bl    ; Attributes (New!)
26664                              <1>
26665                              <1>       ; 04/03/2016
26666                              <1>       ; TRDOS v1 has a bug here! it does not set
26667                              <1>       ; 'DirBuff_ValidData' to 2; as result of this bug,
26668                              <1>       ; 'save_directory_buffer' would not save the new attributes !
26669                              <1>
26670 00008BC1 C605[B8560100]02    <1>       mov    byte [DirBuff_ValidData], 2
26671                              <1>
26672 00008BC8 E89D1A0000          <1>       call   save_directory_buffer
26673 00008BCD 0F824CF8FFFF        <1>       jc     loc_file_rw_cmd_failed
26674                              <1>
26675 00008BD3 EB33                <1>       jmp    short loc_print_attr_changed_message
26676                              <1>
26677                              <1> loc_attr_file_fs_check:
26678 00008BD5 29C0                <1>       sub    eax, eax
26679 00008BD7 8A25[B6560100]      <1>         mov     ah, [DirBuff_DRV]
26680 00008BDD BE00010900          <1>       mov    esi, Logical_DOSDisks
26681 00008BE2 01C6                <1>         add     esi, eax
26682 00008BE4 807E04A1            <1>         cmp     byte [esi+LD_FSType], 0A1h
26683 00008BE8 7309                <1>       jnc    short loc_attr_file_change_fs_file_attributes
26684 00008BEA 66B80D00            <1>       mov    ax, 0Dh ; Invalid Data
26685 00008BEE E92CF8FFFF          <1>       jmp    loc_file_rw_cmd_failed
26686                              <1>
26687                              <1> loc_attr_file_change_fs_file_attributes:
26688                              <1>       ; BL = New MS-DOS File Attributes
26689 00008BF3 88D8                <1>       mov    al, bl ; File/Directory Attributes
26690 00008BF5 30E4                <1>       xor    ah, ah ; Attributes in MS-DOS format sign
26691 00008BF7 E8A7050000          <1>       call   change_fs_file_attributes
26692 00008BFC 0F821DF8FFFF        <1>       jc     loc_file_rw_cmd_failed
26693                              <1>
26694 00008C02 881D[48590100]      <1>       mov    [Attributes], bl
26695                              <1>
26696                              <1> loc_print_attr_changed_message:
26697 00008C08 BE[A6070100]        <1>       mov    esi, Msg_New
26698 00008C0D E84BD7FFFF          <1>       call   print_msg
26699 00008C12 E987FEFFFF          <1>       jmp    loc_show_attributes_no_nextline
26700                              <1>
26701                              <1> rename_file:
26702                              <1>       ; 06/11/2016
26703                              <1>       ; 05/11/2016
26704                              <1>       ; 16/10/2016
26705                              <1>       ; 08/03/2016
26706                              <1>       ; 06/03/2016 (TRDOS 386 =  TRDOS v2.0)
26707                              <1>       ; 20/11/2010 (TRDOS v1, CMD_INTR.ASM, 'cmp_cmd_rename')
26708                              <1>       ; 16/11/2010
26709                              <1>
26710                              <1> get_rename_source_fchar:
26711                              <1>       ; esi = file name
26712 00008C17 803E20             <1>       cmp    byte [esi], 20h
26713 00008C1A 7614                <1>         jna  short loc_rename_nofilename_retn
26714                              <1>
26715 00008C1C 8935[70590100]      <1>       mov    [SourceFilePath], esi
26716                              <1>
26717                              <1> rename_scan_source_file:
26718 00008C22 46                  <1>       inc    esi
```

```
26719 00008C23 803E20              <1>       cmp    byte [esi], 20h
26720 00008C26 7409                <1>       je     short rename_scan_destination_file_1
26721                              <1>       ;jb    short loc_rename_nofilename_retn
26722 00008C28 0F82BFEEFFFF        <1>       jb     loc_cmd_failed
26723 00008C2E EBF2                <1>       jmp    short rename_scan_source_file
26724                              <1>
26725                              <1> loc_rename_nofilename_retn: ; 08/03/2016
26726 00008C30 C3                  <1>       retn
26727                              <1>
26728                              <1> rename_scan_destination_file_1:
26729 00008C31 C60600              <1>       mov    byte [esi], 0
26730                              <1>
26731                              <1> rename_scan_destination_file_2:
26732 00008C34 46                  <1>       inc    esi
26733 00008C35 803E20              <1>       cmp    byte [esi], 20h
26734 00008C38 74FA                <1>       je     short rename_scan_destination_file_2
26735                              <1>       ;jb    short loc_rename_nofilename_retn
26736 00008C3A 0F82ADEEFFFF        <1>       jb     loc_cmd_failed
26737                              <1>
26738 00008C40 8935[74590100]      <1>       mov    [DestinationFilePath], esi
26739                              <1>
26740                              <1> rename_scan_destination_file_3:
26741 00008C46 46                  <1>       inc    esi
26742 00008C47 803E20              <1>       cmp    byte [esi], 20h
26743 00008C4A 77FA                <1>       ja     short rename_scan_destination_file_3
26744                              <1>
26745 00008C4C C60600              <1>       mov    byte [esi], 0
26746                              <1>
26747                              <1> loc_rename_save_current_drive:
26748 00008C4F 8A35[8E4E0100]      <1>       mov    dh, [Current_Drv]
26749 00008C55 8835[EE560100]      <1>       mov    byte [RUN_CDRV], dh
26750                              <1>
26751                              <1> loc_rename_sf_parse_path_name:
26752 00008C5B 8B35[70590100]      <1>       mov    esi, [SourceFilePath]
26753 00008C61 BF[32580100]        <1>       mov    edi, FindFile_Drv
26754 00008C66 E89B150000          <1>       call   parse_path_name
26755 00008C6B 0F827CEEFFFF        <1>       jc     loc_cmd_failed
26756                              <1>
26757                              <1> loc_rename_sf_check_filename_exists:
26758 00008C71 BE[74580100]        <1>       mov    esi, FindFile_Name
26759 00008C76 803E20              <1>       cmp    byte [esi], 20h
26760 00008C79 0F866EEEFFFF        <1>       jna    loc_cmd_failed
26761                              <1>
26762                              <1>       ;mov   [DelFile_FNPointer], esi
26763                              <1>
26764                              <1> loc_rename_sf_drv:
26765                              <1>       ;mov   dh, [Current_Drv]
26766                              <1>       ;mov   [RUN_CDRV], dh
26767                              <1>
26768 00008C7F 8A15[32580100]      <1>       mov    dl, [FindFile_Drv]
26769 00008C85 38F2                <1>       cmp    dl, dh ; dh = [Current_Drv]
26770 00008C87 740B                <1>       je     short rename_sf_change_directory
26771                              <1>
26772 00008C89 E8CFDFFFFF          <1>       call   change_current_drive
26773 00008C8E 0F828BF7FFFF        <1>       jc     loc_file_rw_cmd_failed
26774                              <1>
26775                              <1> rename_sf_change_directory:
26776 00008C94 803D[33580100]20    <1>       cmp    byte [FindFile_Directory], 20h
26777 00008C9B 7618                <1>       jna    short rename_sf_find
26778                              <1>
26779 00008C9D FE05[98020100]      <1>       inc    byte [Restore_CDIR]
26780 00008CA3 BE[33580100]        <1>       mov    esi, FindFile_Directory
26781 00008CA8 30E4                <1>       xor    ah, ah ; CD_COMMAND sign -> 0
26782 00008CAA E8430F0000          <1>       call   change_current_directory
26783 00008CAF 0F826AF7FFFF        <1>       jc     loc_file_rw_cmd_failed
26784                              <1>
26785                              <1> ;rename_sf_change_prompt_dir_string:
26786                              <1>       ;call  change_prompt_dir_string
26787                              <1>
26788                              <1> rename_sf_find:
26789                              <1>       ;mov   esi, [DelFile_FNPointer]
26790 00008CB5 BE[74580100]        <1>       mov    esi, FindFile_Name
26791                              <1>
26792 00008CBA 66B80008            <1>       mov    ax, 0800h ; Except volume labels
26793 00008CBE E82CF3FFFF          <1>       call   find_first_file
26794 00008CC3 0F8256F7FFFF        <1>       jc     loc_file_rw_cmd_failed
26795                              <1>
26796                              <1> loc_rename_sf_ambgfn_check:
26797 00008CC9 6621D2              <1>       and    dx, dx ; Ambiguous filename chars used sign (DX>0)
26798                              <1>       ;      (Note: It was BX in TRDOS v1)
26799                              <1>       ;jz    short loc_rename_sf_found
26800 00008CCC 0F8563FCFFFF        <1>       jnz    loc_file_not_found
26801                              <1>
26802                              <1>       ;mov   eax, 2 ; File not found sign
26803                              <1>       ;stc
26804                              <1>       ;jmp   loc_file_rw_cmd_failed
26805                              <1>
26806                              <1> loc_rename_sf_found:
26807                              <1>       ; EDI = Directory buffer entry offset/address
26808                              <1>       ; BL = File (or Directory) Attributes
26809                              <1>       ;      (Note: It was 'CL' in TRDOS v1)
26810                              <1>       ; mov  bl, [EDI+0Bh]
26811                              <1>
26812 00008CD2 F6C307              <1>       test   bl, 07h ; Attributes, S-H-R
26813 00008CD5 0F854EF7FFFF        <1>       jnz    loc_permission_denied
26814                              <1>
26815 00008CDB BE[32580100]        <1>         mov     esi, FindFile_Drv
26816 00008CE0 BF[78590100]        <1>         mov     edi, SourceFile_Drv
26817 00008CE5 B920000000          <1>       mov    ecx, 32
26818 00008CEA F3A5                <1>       rep    movsd
26819                              <1>
26820                              <1> loc_rename_df_parse_path_name:
```

```
26821 00008CEC 8B35[74590100]      <1>       mov    esi, [DestinationFilePath]
26822 00008CF2 BF[32580100]        <1>       mov    edi, FindFile_Drv
26823 00008CF7 E80A150000          <1>       call   parse_path_name
26824 00008CFC 7219                <1>       jc     short loc_rename_df_cmd_failed
26825                              <1>
26826                              <1>       ;mov   dh, [RUN_CDRV]
26827 00008CFE 8A35[8E4E0100]      <1>       mov    dh, [Current_Drv]
26828                              <1>
26829                              <1>       ; 'rename' command is valid only for same dos drive and same dir!
26830                              <1>       ; ('move' command must be used if source file and destination file
26831                              <1>       ; directories are not same!)
26832 00008D04 8A15[32580100]      <1>       mov    dl, [FindFile_Drv]
26833 00008D0A 38F2                <1>       cmp    dl, dh ; are source and destination drives different ?!
26834 00008D0C 7509                <1>       jne    short loc_rename_df_cmd_failed ; yes!
26835                              <1>
26836                              <1> rename_df_check_dirname_exists:
26837 00008D0E 803D[33580100]00    <1>       cmp    byte [FindFile_Directory], 0
26838 00008D15 760B                <1>       jna    short rename_df_check_filename_exists
26839                              <1>
26840                              <1>       ; different source file and destination file directories !
26841                              <1> loc_rename_df_cmd_failed:
26842 00008D17 B801000000          <1>       mov    eax, 1 ; TRDOS 'Bad command or file name' error
26843 00008D1C F9                  <1>       stc
26844 00008D1D E9FDF6FFFF          <1>       jmp    loc_file_rw_cmd_failed
26845                              <1>
26846                              <1> rename_df_check_filename_exists:
26847 00008D22 BE[74580100]        <1>       mov    esi, FindFile_Name
26848 00008D27 E883F6FFFF          <1>       call   check_filename
26849 00008D2C 0F829FF7FFFF        <1>       jc     loc_mkdir_invalid_dir_name_chars
26850                              <1>
26851                              <1>       ;mov   [DelFile_FNPointer], esi
26852                              <1>       ;cmp   byte [esi], 20h
26853                              <1>       ;ja    short loc_rename_df_find
26854                              <1>
26855                              <1>       ;mov   dh, [Current_Drv] ; dh has not been changed
26856                              <1>
26857                              <1> rename_df_drv_check_writable:
26858 00008D32 0FB6F6              <1>       movzx  esi, dh
26859                              <1>       ;movzx esi, byte [Current_Drv]
26860 00008D35 81C600010900        <1>       add    esi, Logical_DOSDisks
26861                              <1>
26862 00008D3B 88F2                <1>       mov    dl, dh ; dl = [Current_Drv]
26863 00008D3D 8A7601              <1>       mov    dh, [esi+LD_DiskType]
26864                              <1>
26865 00008D40 80FE01              <1>       cmp    dh, 1 ; 0 = Invalid
26866 00008D43 7310                <1>       jnb    short rename_df_compare_sf_df_name
26867                              <1>
26868                              <1>       ; 16/10/2016 (13h -> 30)
26869 00008D45 B81E000000          <1>       mov    eax, 30 ; 'Disk write-protected' error
26870 00008D4A 8B1D[74590100]      <1>       mov    ebx, [DestinationFilePath]
26871 00008D50 E9CAF6FFFF          <1>       jmp    loc_file_rw_cmd_failed
26872                              <1>
26873                              <1> rename_df_compare_sf_df_name:
26874 00008D55 BE[74580100]        <1>       mov    esi, FindFile_Name
26875 00008D5A BF[BA590100]        <1>       mov    edi, SourceFile_Name
26876 00008D5F B90C000000          <1>       mov    ecx, 12
26877                              <1> rename_df_compare_sf_df_name_next:
26878 00008D64 AC                  <1>       lodsb
26879 00008D65 AE                  <1>       scasb
26880 00008D66 7506                <1>       jne    short loc_rename_df_find
26881 00008D68 08C0                <1>       or     al, al
26882 00008D6A 74AB                <1>       jz     short loc_rename_df_cmd_failed
26883 00008D6C E2F6                <1>       loop   rename_df_compare_sf_df_name_next
26884                              <1>
26885                              <1> loc_rename_df_find:
26886                              <1>       ;mov   esi, [DelFile_FNPointer]
26887 00008D6E BE[74580100]        <1>       mov    esi, FindFile_Name
26888                              <1>
26889 00008D73 6631C0              <1>       xor    ax, ax ; Any
26890 00008D76 E874F2FFFF          <1>       call   find_first_file
26891 00008D7B 730A                <1>       jnc    short loc_rename_df_found
26892                              <1>
26893                              <1> loc_rename_df_check_error_code:
26894                              <1>       ;cmp   eax, 2
26895 00008D7D 3C02                <1>       cmp    al, 2 ; Not found error
26896 00008D7F 7411                <1>       je     short rename_df_move_find_struct_to_dest
26897 00008D81 F9                  <1>       stc
26898 00008D82 E998F6FFFF          <1>       jmp loc_file_rw_cmd_failed
26899                              <1>
26900                              <1> loc_rename_df_found:
26901                              <1>       ; 05/11/2016
26902 00008D87 B805000000          <1>       mov    eax, 05h ; permission denied error
26903 00008D8C F9                  <1>       stc
26904 00008D8D E997F6FFFF          <1>       jmp    loc_permission_denied  ; 06/11/2016
26905                              <1>
26906                              <1> rename_df_move_find_struct_to_dest:
26907 00008D92 BE[32580100]        <1>         mov     esi, FindFile_Drv
26908 00008D97 BF[F8590100]        <1>         mov     edi, DestinationFile_Drv
26909 00008D9C B920000000          <1>       mov    ecx, 32
26910 00008DA1 F3A5                <1>       rep    movsd
26911                              <1>
26912                              <1> loc_rename_df_process_q_sf:
26913                              <1>       ;mov   ecx, 12
26914 00008DA3 B10C                <1>       mov    cl, 12
26915 00008DA5 BE[BA590100]        <1>       mov    esi, SourceFile_Name
26916 00008DAA BF[E7070100]        <1>       mov    edi, Rename_OldName
26917                              <1> rename_df_process_q_nml_1_sf:
26918 00008DAF AC                  <1>       lodsb
26919 00008DB0 3C20                <1>         cmp  al, 20h
26920 00008DB2 7603                <1>         jna  short rename_df_process_q_nml_2_sf
26921 00008DB4 AA                  <1>       stosb
26922 00008DB5 E2F8                <1>       loop   rename_df_process_q_nml_1_sf
```

```
26923                                   <1>
26924                                   <1> rename_df_process_q_nml_2_sf:
26925 00008DB7 C60700                    <1>       mov    byte [edi], 0
26926                                   <1>
26927                                   <1> loc_rename_df_process_q_df:
26928                                   <1>       ;mov   ecx, 12
26929 00008DBA B10C                      <1>       mov    cl, 12
26930 00008DBC BE[3A5A0100]              <1>       mov    esi, DestinationFile_Name
26931 00008DC1 BF[F8070100]              <1>       mov    edi, Rename_NewName
26932                                   <1> rename_df_process_q_nml_1_df:
26933 00008DC6 AC                        <1>       lodsb
26934 00008DC7 3C20                      <1>       cmp    al, 20h
26935 00008DC9 7603                      <1>       jna    short loc_rename_df_process_q_nml_2_df
26936 00008DCB AA                        <1>       stosb
26937 00008DCC E2F8                      <1>       loop   rename_df_process_q_nml_1_df
26938                                   <1>
26939                                   <1> loc_rename_df_process_q_nml_2_df:
26940 00008DCE C60700                    <1>       mov    byte [edi], 0
26941                                   <1>
26942                                   <1> loc_rename_confirmation_question:
26943 00008DD1 BE[BF070100]              <1>       mov    esi, Msg_DoYouWantRename
26944 00008DD6 E882D5FFFF                <1>       call   print_msg
26945                                   <1>
26946 00008DDB A0[D5590100]              <1>       mov    al, [SourceFile_DirEntry+11] ; Attributes
26947 00008DE0 2410                      <1>       and    al, 10h
26948 00008DE2 750C                      <1>       jnz    short rename_confirmation_question_dir
26949                                   <1>
26950                                   <1> rename_confirmation_question_file:
26951 00008DE4 BE[D6070100]              <1>       mov    esi, Rename_File
26952 00008DE9 E86FD5FFFF                <1>       call   print_msg
26953 00008DEE EB0A                      <1>       jmp    short rename_confirmation_question_as
26954                                   <1>
26955                                   <1> rename_confirmation_question_dir:
26956 00008DF0 BE[DC070100]              <1>       mov    esi, Rename_Directory
26957 00008DF5 E863D5FFFF                <1>       call   print_msg
26958                                   <1>
26959                                   <1> rename_confirmation_question_as:
26960 00008DFA BE[E7070100]              <1>       mov    esi, Rename_OldName
26961 00008DFF E859D5FFFF                <1>       call   print_msg
26962 00008E04 BE[F4070100]              <1>       mov    esi, Msg_File_rename_as
26963 00008E09 E84FD5FFFF                <1>       call   print_msg
26964 00008E0E BE[1B070100]              <1>       mov    esi, Msg_YesNo
26965 00008E13 E845D5FFFF                <1>       call   print_msg
26966                                   <1>
26967                                   <1> loc_rename_ask_again:
26968 00008E18 30E4                      <1>       xor    ah, ah
26969 00008E1A E8F77DFFFF                <1>       call   int16h
26970 00008E1F 3C1B                      <1>       cmp    al, 1Bh
26971 00008E21 740F                      <1>       je     short loc_do_not_rename_file
26972 00008E23 24DF                      <1>       and    al, 0DFh
26973 00008E25 A2[25070100]              <1>       mov    [Y_N_nextline], al
26974 00008E2A 3C59                      <1>       cmp    al, 'Y'
26975 00008E2C 7404                      <1>       je     short loc_yes_rename_file
26976 00008E2E 3C4E                      <1>       cmp    al, 'N'
26977 00008E30 75E6                      <1>       jne    short loc_rename_ask_again
26978                                   <1>
26979                                   <1> loc_do_not_rename_file:
26980                                   <1> loc_yes_rename_file:
26981 00008E32 A2[25070100]              <1>       mov    [Y_N_nextline], al
26982 00008E37 6650                      <1>       push   ax
26983 00008E39 BE[25070100]              <1>       mov    esi, Y_N_nextline
26984 00008E3E E81AD5FFFF                <1>       call   print_msg
26985 00008E43 6658                      <1>       pop    ax
26986                                   <1>       ;cmp   al, 'Y' ; 'yes'
26987                                   <1>       ;cmc
26988                                   <1>         ;jnc loc_file_rw_restore_retn
26989 00008E45 3C4E                      <1>       cmp    al, 'N' ; 'no'
26990 00008E47 0F84D2F5FFFF              <1>         je   loc_file_rw_restore_retn
26991                                   <1>
26992 00008E4D BE[F8070100]              <1>       mov    esi, Rename_NewName
26993 00008E52 668B0D[F2590100]          <1>       mov    cx, [SourceFile_DirEntryNumber]
26994 00008E59 66A1[DE590100]            <1>       mov    ax, [SourceFile_DirEntry+20] ; First Cluster, HW
26995 00008E5F 66C1E010                  <1>       shl    ax, 16
26996 00008E63 66A1[E4590100]            <1>       mov    ax, [SourceFile_DirEntry+26] ; First Cluster, LW
26997                                   <1>
26998 00008E69 0FB61D[C7590100]          <1>       movzx  ebx, byte [SourceFile_LongNameEntryLength]
26999 00008E70 E8DB1B0000                <1>       call   rename_directory_entry
27000 00008E75 E9D3F6FFFF                <1>       jmp    loc_rename_file_ok
27001                                   <1> ;loc_rename_file_ok:
27002                                   <1> ;     jc     loc_run_cmd_failed
27003                                   <1> ;     mov    esi, Msg_OK
27004                                   <1> ;     call   proc_printmsg
27005                                   <1> ;     jmp    loc_file_rw_restore_retn
27006                                   <1>
27007                                   <1> move_file:
27008                                   <1>       ; 11/03/2016
27009                                   <1>       ; 09/03/2016
27010                                   <1>       ; 08/03/2016 (TRDOS 386 =  TRDOS v2.0)
27011                                   <1>       ; 21/05/2011 (TRDOS v1, CMD_INTR.ASM, 'cmp_cmd_move')
27012                                   <1>       ; 23/04/2011
27013                                   <1>
27014                                   <1> get_move_source_fchar:
27015                                   <1>       ; esi = file name
27016 00008E7A 803E20                    <1>       cmp    byte [esi], 20h
27017 00008E7D 7614                      <1>         jna  short loc_move_nofilename_retn
27018                                   <1>
27019 00008E7F 8935[70590100]            <1>       mov    [SourceFilePath], esi
27020                                   <1>
27021                                   <1> move_scan_source_file:
27022 00008E85 46                        <1>       inc    esi
27023 00008E86 803E20                    <1>       cmp    byte [esi], 20h
27024 00008E89 7409                      <1>         je      short move_scan_destination_1
```

```
27025                                  <1>       ;jb    short loc_move_nofilename_retn
27026 00008E8B 0F825CECFFFF            <1>       jb     loc_cmd_failed
27027 00008E91 EBF2                    <1>       jmp    short move_scan_source_file
27028                                  <1>
27029                                  <1> loc_move_nofilename_retn:
27030 00008E93 C3                      <1>       retn
27031                                  <1>
27032                                  <1> move_scan_destination_1:
27033 00008E94 C60600                  <1>       mov    byte [esi], 0
27034                                  <1>
27035                                  <1> move_scan_destination_2:
27036 00008E97 46                      <1>       inc    esi
27037 00008E98 803E20                  <1>       cmp    byte [esi], 20h
27038 00008E9B 74FA                    <1>       je     short move_scan_destination_2
27039                                  <1>       ;jb    short loc_move_nofilename_retn
27040 00008E9D 0F824AECFFFF            <1>       jb     loc_cmd_failed
27041                                  <1>
27042 00008EA3 8935[74590100]          <1>       mov    [DestinationFilePath], esi
27043                                  <1>
27044                                  <1> move_scan_destination_3:
27045 00008EA9 46                      <1>       inc    esi
27046 00008EAA 803E20                  <1>       cmp    byte [esi], 20h
27047 00008EAD 77FA                    <1>       ja     short move_scan_destination_3
27048 00008EAF C60600                  <1>       mov    byte [esi], 0
27049                                  <1>
27050                                  <1> loc_move_scan_destination_OK:
27051 00008EB2 8B35[70590100]          <1>       mov    esi, [SourceFilePath]
27052 00008EB8 8B3D[74590100]          <1>       mov    edi, [DestinationFilePath]
27053                                  <1>
27054 00008EBE B001                    <1>       mov    al, 1  ; move procedure Phase 1
27055 00008EC0 E8081C0000              <1>       call   move_source_file_to_destination_file
27056 00008EC5 7328                    <1>       jnc    short move_source_file_to_destination_question
27057                                  <1>
27058                                  <1> loc_move_cmd_failed_1:
27059 00008EC7 08C0                    <1>       or     al, al
27060 00008EC9 0F841EECFFFF            <1>       jz     loc_cmd_failed
27061 00008ECF 3C11                    <1>       cmp    al, 11h
27062 00008ED1 740D                    <1>       je     short loc_msg_not_same_device
27063 00008ED3 3C05                    <1>       cmp    al, 05h
27064 00008ED5 0F853DECFFFF            <1>       jne    loc_run_cmd_failed
27065                                  <1>
27066 00008EDB E949F5FFFF              <1>       jmp    loc_permission_denied
27067                                  <1>
27068                                  <1>       ;mov   esi, Msg_Permission_denied
27069                                  <1>       ;call  print_msg
27070                                  <1>       ;jmp   loc_file_rw_restore_retn
27071                                  <1>
27072                                  <1> loc_msg_not_same_device:
27073 00008EE0 BE[05080100]            <1>       mov    esi, msg_not_same_drv
27074 00008EE5 E873D4FFFF              <1>       call   print_msg
27075 00008EEA E930F5FFFF              <1>       jmp    loc_file_rw_restore_retn
27076                                  <1>
27077                                  <1> move_source_file_to_destination_question:
27078 00008EEF A0[78590100]            <1>         mov     al, [SourceFile_Drv]
27079 00008EF4 0441                    <1>       add    al, 'A'
27080 00008EF6 A2[67080100]            <1>       mov    [msg_source_file_drv], al
27081 00008EFB A0[F8590100]            <1>         mov     al, [DestinationFile_Drv]
27082 00008F00 0441                    <1>       add    al, 'A'
27083 00008F02 A2[86080100]            <1>       mov    [msg_destination_file_drv], al
27084                                  <1>
27085 00008F07 57                      <1>       push   edi ; *
27086                                  <1>
27087 00008F08 BE[4B080100]            <1>       mov    esi, msg_source_file
27088 00008F0D E84BD4FFFF              <1>       call   print_msg
27089 00008F12 BE[79590100]            <1>       mov    esi, SourceFile_Directory
27090 00008F17 803E20                  <1>       cmp    byte [esi], 20h
27091 00008F1A 7605                    <1>       jna    short msftdfq_sfn
27092 00008F1C E83CD4FFFF              <1>       call   print_msg
27093                                  <1> msftdfq_sfn:
27094 00008F21 BE[BA590100]            <1>       mov    esi, SourceFile_Name
27095 00008F26 E832D4FFFF              <1>       call   print_msg
27096 00008F2B BE[6A080100]            <1>       mov    esi, msg_destination_file
27097 00008F30 E828D4FFFF              <1>       call   print_msg
27098 00008F35 BE[F9590100]            <1>       mov    esi, DestinationFile_Directory
27099 00008F3A 803E20                  <1>       cmp    byte [esi], 20h
27100 00008F3D 7605                    <1>       jna    short msftdfq_dfn
27101 00008F3F E819D4FFFF              <1>       call   print_msg
27102                                  <1> msftdfq_dfn:
27103 00008F44 BE[3A5A0100]            <1>       mov    esi, DestinationFile_Name
27104 00008F49 E80FD4FFFF              <1>       call   print_msg
27105 00008F4E BE[89080100]            <1>       mov    esi, msg_copy_nextline
27106 00008F53 E805D4FFFF              <1>       call   print_msg
27107 00008F58 BE[89080100]            <1>       mov    esi, msg_copy_nextline
27108 00008F5D E8FBD3FFFF              <1>       call   print_msg
27109                                  <1>
27110                                  <1> loc_move_ask_for_new_file_yes_no:
27111 00008F62 BE[17080100]            <1>       mov    esi, Msg_DoYouWantMoveFile
27112 00008F67 E8F1D3FFFF              <1>       call   print_msg
27113 00008F6C BE[1B070100]            <1>       mov    esi, Msg_YesNo
27114 00008F71 E8E7D3FFFF              <1>       call   print_msg
27115                                  <1> loc_move_ask_for_new_file_again:
27116 00008F76 30E4                    <1>       xor    ah, ah
27117 00008F78 E8997CFFFF              <1>       call   int16h
27118 00008F7D 3C1B                    <1>       cmp    al, 1Bh
27119                                  <1>       ;je    short loc_do_not_move_file
27120 00008F7F 744F                    <1>       je     short loc_move_y_n_escape
27121 00008F81 24DF                    <1>       and    al, 0DFh
27122 00008F83 A2[25070100]            <1>         mov     [Y_N_nextline], al
27123 00008F88 3C59                    <1>       cmp    al, 'Y'
27124 00008F8A 7404                    <1>       je     short loc_yes_move_file
27125 00008F8C 3C4E                    <1>       cmp    al, 'N'
27126 00008F8E 75E6                    <1>       jne    short loc_move_ask_for_new_file_again
```

```
27127                                  <1>
27128                                  <1> loc_do_not_move_file:
27129                                  <1> loc_yes_move_file:
27130 00008F90 A2[25070100]            <1>       mov    [Y_N_nextline], al
27131 00008F95 6650                    <1>       push   ax
27132 00008F97 BE[25070100]            <1>       mov    esi, Y_N_nextline
27133 00008F9C E8BCD3FFFF              <1>       call   print_msg
27134 00008FA1 6658                    <1>       pop    ax
27135 00008FA3 5F                      <1>       pop    edi ; *
27136                                  <1>       ;cmp   al, 'Y' ; 'yes'
27137                                  <1>       ;cmc
27138                                  <1>         ;jnc loc_file_rw_restore_retn
27139 00008FA4 3C4E                    <1>       cmp    al, 'N' ; 'no'
27140 00008FA6 0F8473F4FFFF            <1>         je   loc_file_rw_restore_retn
27141                                  <1>
27142                                  <1> loc_move_yes_move_file:
27143 00008FAC B002                    <1>       mov    al, 2 ; move procedure Phase 2
27144 00008FAE E81A1B0000              <1>       call   move_source_file_to_destination_file
27145                                  <1>       ;jc    short loc_move_cmd_failed_2
27146 00008FB3 0F839AF5FFFF            <1>         jnc     move_source_file_to_destination_OK
27147                                  <1>
27148                                  <1> ;move_source_file_to_destination_OK:
27149                                  <1> ;     mov    esi, Msg_OK
27150                                  <1> ;     call   print_msg
27151                                  <1> ;     jmp    loc_file_rw_restore_retn
27152                                  <1>
27153                                  <1> loc_move_cmd_failed_2:
27154 00008FB9 3C27                    <1>       cmp    al, 27h
27155 00008FBB 0F8557EBFFFF            <1>       jne    loc_run_cmd_failed
27156                                  <1>
27157 00008FC1 BE[30080100]            <1>       mov    esi, msg_insufficient_disk_space
27158 00008FC6 E892D3FFFF              <1>       call   print_msg
27159                                  <1>
27160 00008FCB E94FF4FFFF              <1>       jmp    loc_file_rw_restore_retn
27161                                  <1>
27162                                  <1> loc_move_y_n_escape:
27163 00008FD0 B04E                    <1>       mov    al, 'N' ; 'no'
27164 00008FD2 EBBC                    <1>       jmp    short loc_do_not_move_file
27165                                  <1>
27166                                  <1> copy_file:
27167                                  <1>       ; 15/10/2016
27168                                  <1>       ; 24/03/2016
27169                                  <1>       ; 21/03/2016
27170                                  <1>       ; 15/03/2016 (TRDOS 386 =  TRDOS v2.0)
27171                                  <1>       ; 21/05/2011 (TRDOS v1, CMD_INTR.ASM, 'cmp_cmd_copy')
27172                                  <1>       ; 01/08/2010
27173                                  <1>
27174                                  <1> get_copy_source_fchar:
27175                                  <1>       ; esi = file name
27176 00008FD4 803E20                 <1>       cmp    byte [esi], 20h
27177 00008FD7 7614                    <1>         jna     short loc_copy_nofilename_retn
27178                                  <1>
27179 00008FD9 8935[70590100]          <1>       mov    [SourceFilePath], esi
27180                                  <1>
27181                                  <1> copy_scan_source_file:
27182 00008FDF 46                      <1>       inc    esi
27183 00008FE0 803E20                  <1>       cmp    byte [esi], 20h
27184 00008FE3 7409                    <1>       je     short copy_scan_destination_1
27185                                  <1>       ;jb    short loc_copy_nofilename_retn
27186 00008FE5 0F8202EBFFFF            <1>       jb     loc_cmd_failed
27187 00008FEB EBF2                    <1>       jmp    short copy_scan_source_file
27188                                  <1>
27189                                  <1> loc_copy_nofilename_retn:
27190 00008FED C3                      <1>       retn
27191                                  <1>
27192                                  <1> copy_scan_destination_1:
27193 00008FEE C60600                  <1>       mov    byte [esi], 0
27194                                  <1>
27195                                  <1> copy_scan_destination_2:
27196 00008FF1 46                      <1>       inc    esi
27197 00008FF2 803E20                  <1>       cmp    byte [esi], 20h
27198 00008FF5 74FA                    <1>       je     short copy_scan_destination_2
27199                                  <1>       ;jb    short loc_copy_nofilename_retn
27200 00008FF7 0F82F0EAFFFF            <1>       jb     loc_cmd_failed
27201                                  <1>
27202 00008FFD 8935[74590100]          <1>       mov    [DestinationFilePath], esi
27203                                  <1>
27204                                  <1> copy_scan_destination_3:
27205 00009003 46                      <1>       inc    esi
27206 00009004 803E20                  <1>       cmp    byte [esi], 20h
27207 00009007 77FA                    <1>       ja     short copy_scan_destination_3
27208 00009009 C60600                  <1>       mov    byte [esi], 0
27209                                  <1>
27210                                  <1> loc_copy_save_current_drive:
27211 0000900C 8A35[8E4E0100]          <1>       mov    dh, [Current_Drv]
27212 00009012 8835[EE560100]          <1>       mov    [RUN_CDRV], dh
27213                                  <1>
27214                                  <1> copy_source_file_to_destination_phase_1:
27215 00009018 8B35[70590100]          <1>       mov    esi, [SourceFilePath]
27216 0000901E 8B3D[74590100]          <1>       mov    edi, [DestinationFilePath]
27217                                  <1>
27218 00009024 B001                    <1>       mov    al, 1  ; copy procedure Phase 1
27219 00009026 E83F1D0000              <1>       call   copy_source_file_to_destination_file
27220 0000902B 732B                    <1>       jnc    short copy_source_file_to_destination_question
27221                                  <1>
27222                                  <1> loc_copy_cmd_failed_1:
27223                                  <1>       ; 18/03/2016 (restore current drive and directory)
27224 0000902D 08C0                    <1>       or     al, al
27225 0000902F 7507                    <1>       jnz    short loc_copy_cmd_failed_2
27226                                  <1>
27227 00009031 FEC0                    <1>         inc     al ; mov al, 1 ; Bad command or file name !
27228 00009033 E9E0EAFFFF              <1>       jmp    loc_run_cmd_failed
```

```
27229                                  <1>
27230                                  <1> loc_copy_cmd_failed_2:
27231 00009038 3C27                    <1>       cmp    al, 27h ; Insufficient disk space
27232 0000903A 740D                    <1>       je     short loc_file_write_insuff_disk_space_msg
27233                                  <1>
27234 0000903C 3C05                    <1>       cmp    al, 05h
27235 0000903E 0F85D4EAFFFF            <1>       jne    loc_run_cmd_failed
27236                                  <1>
27237 00009044 E9E0F3FFFF              <1>       jmp    loc_permission_denied
27238                                  <1>
27239                                  <1> loc_file_write_insuff_disk_space_msg:
27240 00009049 BE[30080100]            <1>       mov    esi, msg_insufficient_disk_space
27241 0000904E E80AD3FFFF              <1>       call   print_msg
27242 00009053 E9C7F3FFFF              <1>         jmp     loc_file_rw_restore_retn
27243                                  <1>
27244                                  <1> copy_source_file_to_destination_question:
27245 00009058 57                      <1>       push   edi ; *
27246                                  <1>
27247                                  <1>       ; dh = source file attributes
27248                                  <1>       ; dl > 0 -> destination file found
27249 00009059 20D2                    <1>       and    dl, dl
27250 0000905B 7449                    <1>       jz     short copy_source_file_to_destination_pass_owrq
27251                                  <1>
27252                                  <1> loc_copy_ask_for_owr_yes_no:
27253 0000905D BE[8C080100]            <1>       mov    esi, Msg_DoYouWantOverWriteFile
27254 00009062 E8F6D2FFFF              <1>       call   print_msg
27255 00009067 BE[3A5A0100]            <1>       mov    esi, DestinationFile_Name
27256 0000906C E8ECD2FFFF              <1>       call   print_msg
27257 00009071 BE[1B070100]            <1>       mov    esi, Msg_YesNo
27258 00009076 E8E2D2FFFF              <1>       call   print_msg
27259                                  <1>
27260                                  <1> loc_copy_ask_for_owr_again:
27261 0000907B 30E4                    <1>       xor    ah, ah
27262 0000907D E8947BFFFF              <1>       call   int16h
27263 00009082 3C1B                    <1>       cmp    al, 1Bh
27264                                  <1>         ;je     loc_do_not_copy_file
27265 00009084 7419                    <1>         je      short loc_copy_y_n_escape
27266 00009086 24DF                    <1>       and    al, 0DFh
27267 00009088 A2[25070100]            <1>         mov     [Y_N_nextline], al
27268 0000908D 3C59                    <1>       cmp    al, 'Y'
27269 0000908F 0F84B1000000            <1>         je      loc_yes_copy_file
27270 00009095 3C4E                    <1>       cmp    al, 'N'
27271 00009097 0F84A9000000            <1>         je      loc_do_not_copy_file
27272 0000909D EBDC                    <1>       jmp    short loc_copy_ask_for_owr_again
27273                                  <1>
27274                                  <1> loc_copy_y_n_escape:
27275 0000909F B04E                    <1>       mov    al, 'N' ; 'no'
27276 000090A1 E9A0000000              <1>         jmp     loc_do_not_copy_file
27277                                  <1>
27278                                  <1> copy_source_file_to_destination_pass_owrq:
27279 000090A6 A0[78590100]            <1>       mov     al, [SourceFile_Drv]
27280 000090AB 0441                    <1>       add    al, 'A'
27281 000090AD A2[67080100]            <1>       mov    [msg_source_file_drv], al
27282 000090B2 A0[F8590100]            <1>         mov     al, [DestinationFile_Drv]
27283 000090B7 0441                    <1>       add    al, 'A'
27284 000090B9 A2[86080100]            <1>       mov    [msg_destination_file_drv], al
27285                                  <1>
27286 000090BE BE[4B080100]            <1>       mov    esi, msg_source_file
27287 000090C3 E895D2FFFF              <1>       call   print_msg
27288 000090C8 BE[79590100]            <1>       mov    esi, SourceFile_Directory
27289 000090CD 803E20                  <1>       cmp    byte [esi], 20h
27290 000090D0 7605                    <1>       jna    short csftdfq_sfn
27291 000090D2 E886D2FFFF              <1>       call   print_msg
27292                                  <1> csftdfq_sfn:
27293 000090D7 BE[BA590100]            <1>       mov    esi, SourceFile_Name
27294 000090DC E87CD2FFFF              <1>       call   print_msg
27295 000090E1 BE[6A080100]            <1>       mov    esi, msg_destination_file
27296 000090E6 E872D2FFFF              <1>       call   print_msg
27297 000090EB BE[F9590100]            <1>       mov    esi, DestinationFile_Directory
27298 000090F0 803E20                  <1>       cmp    byte [esi], 20h
27299 000090F3 7605                    <1>       jna    short csftdfq_dfn
27300 000090F5 E863D2FFFF              <1>       call   print_msg
27301                                  <1> csftdfq_dfn:
27302 000090FA BE[3A5A0100]            <1>       mov    esi, DestinationFile_Name
27303 000090FF E859D2FFFF              <1>       call   print_msg
27304 00009104 BE[89080100]            <1>       mov    esi, msg_copy_nextline
27305 00009109 E84FD2FFFF              <1>       call   print_msg
27306 0000910E BE[89080100]            <1>       mov    esi, msg_copy_nextline
27307 00009113 E845D2FFFF              <1>       call   print_msg
27308                                  <1>
27309                                  <1> loc_copy_ask_for_new_file_yes_no:
27310 00009118 BE[AB080100]            <1>       mov    esi, Msg_DoYouWantCopyFile
27311 0000911D E83BD2FFFF              <1>       call   print_msg
27312 00009122 BE[1B070100]            <1>       mov    esi, Msg_YesNo
27313 00009127 E831D2FFFF              <1>       call   print_msg
27314                                  <1>
27315                                  <1> loc_copy_ask_for_new_file_again:
27316 0000912C 30E4                    <1>       xor    ah, ah
27317 0000912E E8E37AFFFF              <1>       call   int16h
27318 00009133 3C1B                    <1>       cmp    al, 1Bh
27319 00009135 740F                    <1>       je     short loc_do_not_copy_file
27320 00009137 24DF                    <1>       and    al, 0DFh
27321 00009139 A2[25070100]            <1>         mov     [Y_N_nextline], al
27322 0000913E 3C59                    <1>       cmp    al, 'Y'
27323 00009140 7404                    <1>       je     short loc_yes_copy_file
27324 00009142 3C4E                    <1>       cmp    al, 'N'
27325 00009144 75E6                    <1>       jne    short loc_copy_ask_for_new_file_again
27326                                  <1>
27327                                  <1> loc_do_not_copy_file:
27328                                  <1> loc_yes_copy_file:
27329 00009146 A2[25070100]            <1>       mov    [Y_N_nextline], al
27330 0000914B 6650                    <1>       push   ax
```

```
27331 0000914D BE[25070100]        <1>       mov    esi, Y_N_nextline
27332 00009152 E806D2FFFF          <1>       call   print_msg
27333 00009157 6658                <1>       pop    ax
27334 00009159 5F                  <1>       pop    edi ; *
27335                              <1>       ;cmp   al, 'Y' ; 'yes'
27336                              <1>       ;cmc
27337                              <1>         ;jnc loc_file_rw_restore_retn
27338 0000915A 3C4E                <1>       cmp    al, 'N' ; 'no'
27339 0000915C 0F84BDF2FFFF        <1>         je   loc_file_rw_restore_retn
27340                              <1>
27341                              <1> copy_source_file_to_destination_pass_q:
27342 00009162 B002                <1>       mov    al, 2  ; copy procedure Phase 2
27343 00009164 E8011C0000          <1>       call   copy_source_file_to_destination_file
27344                              <1>       ;jc    short loc_file_write_check_disk_space_err
27345                              <1>
27346                              <1>       ; 24/03/2016
27347 00009169 6651                <1>       push   cx
27348 0000916B BE[89080100]        <1>       mov    esi, msg_copy_nextline
27349 00009170 E8E8D1FFFF          <1>       call   print_msg
27350                              <1>       ;pop   cx
27351 00009175 6658                <1>       pop    ax
27352                              <1>
27353                              <1>       ;or    cl, cl
27354 00009177 08C0                <1>       or     al, al
27355 00009179 7419                <1>       jz     short copy_source_file_to_destination_OK
27356                              <1>
27357                              <1>       ; 15/10/2016 (1Dh -> 18)
27358                              <1>       ; 18/03/2016 (1Dh)
27359                              <1>       ;cmp   cl, 18 ; write error
27360 0000917B 3C12                <1>       cmp    al, 18
27361 0000917D 7506                <1>       jne    short copy_source_file_to_destination_not_OK
27362                              <1>       ;
27363                              <1>       ;mov   al, cl ; error number (write fault!)
27364 0000917F F9                  <1>       stc
27365 00009180 E99AF2FFFF          <1>       jmp    loc_file_rw_cmd_failed
27366                              <1>
27367                              <1> copy_source_file_to_destination_not_OK:
27368 00009185 BE[C4080100]        <1>       mov    esi, Msg_read_file_error_before_EOF
27369 0000918A E8CED1FFFF          <1>       call   print_msg
27370 0000918F E98BF2FFFF          <1>       jmp    loc_file_rw_restore_retn
27371                              <1>
27372                              <1> copy_source_file_to_destination_OK:
27373 00009194 BE[29070100]        <1>       mov    esi, Msg_OK
27374 00009199 E8BFD1FFFF          <1>       call   print_msg
27375                              <1>
27376 0000919E E97CF2FFFF          <1>       jmp    loc_file_rw_restore_retn
27377                              <1>
27378                              <1> ;loc_file_write_check_disk_space_err:
27379                              <1>       ;cmp   al, 27h ; Insufficient disk space
27380                              <1>       ;je    loc_file_write_insuff_disk_space_msg
27381                              <1>         ;jb  loc_file_rw_cmd_failed
27382                              <1>
27383                              <1>       ;call  print_misc_error_msg ; 15/03/2016
27384                              <1>         ;jmp loc_file_rw_restore_retn
27385                              <1>
27386                              <1> change_fs_file_attributes:
27387                              <1>       ; 04/03/2016 ; Temporary
27388                              <1>       ; AL = File or directory attributes
27389                              <1>       ; AH = 0 -> Attributes are in MS-DOS format
27390                              <1>       ; AH > 0 -> Attributes are in SINGLIX format
27391                              <1>       ;push  ebx
27392                              <1>       ; ... do somethings here ...
27393                              <1>       ;pop   ebx
27394                              <1>       ; BL = File or directory attributes
27395 000091A3 C3                  <1>       retn
27396                              <1>
27397                              <1> set_get_env:
27398                              <1>       ; 11/04/2016 (TRDOS 386 =  TRDOS v2.0)
27399                              <1>       ; 02/09/2011 (TRDOS v1, CMD_INTR.ASM, 'cmp_cmd_set')
27400                              <1>       ; 2005 - 28/08/2011
27401                              <1> get_setenv_fchar:
27402                              <1>       ; esi = environment variable/string
27403 000091A4 8A06                <1>       mov    al, [esi]
27404 000091A6 3C20                <1>       cmp    al, 20h
27405 000091A8 771E                <1>       ja     short loc_find_env
27406                              <1>
27407 000091AA BE00300900          <1>       mov    esi, Env_Page
27408                              <1> loc_print_setline:
27409 000091AF 803E00              <1>       cmp    byte [esi], 0
27410 000091B2 7613                <1>       jna    short loc_setenv_retn
27411 000091B4 E8A4D1FFFF          <1>       call   print_msg
27412 000091B9 56                  <1>       push   esi
27413 000091BA BE[030F0100]        <1>       mov    esi, nextline
27414 000091BF E899D1FFFF          <1>       call   print_msg
27415 000091C4 5E                  <1>       pop    esi
27416 000091C5 EBE8                <1>       jmp    short loc_print_setline
27417                              <1>
27418                              <1> loc_setenv_retn:
27419 000091C7 C3                  <1>       retn
27420                              <1>
27421                              <1> loc_find_env:
27422 000091C8 3C3D                <1>       cmp    al, '='
27423 000091CA 0F841DE9FFFF        <1>       je     loc_cmd_failed
27424                              <1>
27425 000091D0 56                  <1>       push   esi
27426                              <1> loc_repeat_env_equal_check:
27427 000091D1 46                  <1>       inc    esi
27428 000091D2 803E3D              <1>       cmp    byte [esi], '='
27429 000091D5 7431                <1>       je     short pass_env_equal_check
27430 000091D7 803E20              <1>       cmp    byte [esi], 20h
27431 000091DA 73F5                <1>       jnb    short loc_repeat_env_equal_check
27432 000091DC C60600              <1>       mov    byte [esi], 0
```

```
27433 000091DF 5E                <1>       pop    esi
27434 000091E0 BF[8E4F0100]      <1>       mov    edi, TextBuffer ; out buffer
27435 000091E5 B9FF000000        <1>       mov    ecx, 255 ; maximum size (limit)
27436 000091EA 30C0              <1>       xor    al, al ; 0 -> use [ESI]
27437 000091EC E89E000000        <1>       call   get_environment_string
27438 000091F1 72D4              <1>       jc     short loc_setenv_retn
27439                            <1>
27440 000091F3 BE[8E4F0100]      <1>       mov    esi, TextBuffer
27441 000091F8 E860D1FFFF        <1>       call   print_msg
27442 000091FD BE[030F0100]      <1>       mov    esi, nextline
27443 00009202 E856D1FFFF        <1>       call   print_msg
27444                            <1>
27445 00009207 C3                <1>       retn
27446                            <1>
27447                            <1> pass_env_equal_check:
27448 00009208 46                <1>       inc    esi
27449 00009209 803E20            <1>       cmp    byte [esi], 20h
27450 0000920C 73FA              <1>       jnb    short pass_env_equal_check
27451 0000920E C60600            <1>       mov    byte [esi], 0
27452                            <1>
27453                            <1> loc_call_set_env_string:
27454 00009211 5E                <1>       pop    esi
27455 00009212 E83B010000        <1>       call   set_environment_string
27456 00009217 73AE              <1>       jnc    short loc_setenv_retn
27457                            <1>
27458                            <1> loc_set_cmd_failed:
27459 00009219 3C08              <1>       cmp    al, 08h
27460 0000921B 0F85CCE8FFFF      <1>       jne    loc_cmd_failed
27461                            <1>
27462 00009221 BE[04090100]      <1>       mov    esi, Msg_No_Set_Space
27463 00009226 E832D1FFFF        <1>       call   print_msg
27464                            <1>
27465 0000922B C3                <1>       retn
27466                            <1>
27467                            <1> set_get_path:
27468                            <1>       ; 11/04/2016 (TRDOS 386 =  TRDOS v2.0)
27469                            <1>       ; 03/09/2011 (TRDOS v1, CMD_INTR.ASM, 'cmp_cmd_path')
27470                            <1>       ; 2005
27471                            <1> get_path_fchar:
27472                            <1>       ; esi = path
27473 0000922C 803E20            <1>       cmp    byte [esi], 20h
27474 0000922F 7737              <1>       ja     short loc_set_path
27475                            <1>
27476 00009231 BE00300900        <1>       mov    esi, Env_Page
27477                            <1> loc_print_path:
27478 00009236 803E00            <1>       cmp    byte [esi], 0
27479 00009239 762C              <1>       jna    short loc_path_retn
27480                            <1>
27481 0000923B BE[64030100]      <1>       mov    esi, Cmd_Path ; 'PATH' address
27482 00009240 BF[8E4F0100]      <1>       mov    edi, TextBuffer ; oout buffer
27483 00009245 30C0              <1>       xor    al, al  ; use [ESI]
27484 00009247 B9FF000000        <1>       mov    ecx, 255 ; maximum size (limit)
27485 0000924C E83E000000        <1>       call   get_environment_string
27486 00009251 7214              <1>       jc     short loc_path_retn
27487                            <1>
27488 00009253 BE[8E4F0100]      <1>       mov    esi, TextBuffer
27489 00009258 E800D1FFFF        <1>       call   print_msg
27490 0000925D BE[030F0100]      <1>       mov    esi, nextline
27491 00009262 E8F6D0FFFF        <1>       call   print_msg
27492                            <1>
27493                            <1> loc_path_retn:
27494 00009267 C3                <1>       retn
27495                            <1>
27496                            <1> loc_set_path:
27497 00009268 56                <1>       push   esi
27498                            <1> loc_set_path_find_end:
27499 00009269 46                <1>       inc    esi
27500 0000926A 803E20            <1>       cmp    byte [esi], 20h
27501 0000926D 73FA              <1>       jnb    short loc_set_path_find_end
27502 0000926F C60600            <1>       mov    byte [esi], 0
27503                            <1> loc_set_path_header:
27504 00009272 5E                <1>       pop    esi
27505 00009273 4E                <1>       dec    esi
27506 00009274 C6063D            <1>       mov    byte [esi], '='
27507 00009277 4E                <1>       dec    esi
27508 00009278 C60648            <1>       mov    byte [esi], 'H'
27509 0000927B 4E                <1>       dec    esi
27510 0000927C C60654            <1>       mov    byte [esi], 'T'
27511 0000927F 4E                <1>       dec    esi
27512 00009280 C60641            <1>       mov    byte [esi], 'A'
27513 00009283 4E                <1>       dec    esi
27514 00009284 C60650            <1>       mov    byte [esi], 'P'
27515                            <1>
27516                            <1> loc_path_call_set_env_string:
27517 00009287 E8C6000000        <1>       call   set_environment_string
27518 0000928C 728B              <1>         jc   short loc_set_cmd_failed
27519                            <1>
27520 0000928E C3                <1>       retn
27521                            <1>
27522                            <1> get_environment_string:
27523                            <1>       ; 12/04/2016
27524                            <1>       ; 11/04/2016
27525                            <1>       ; 05/04/2016 (TRDOS 386 =  TRDOS v2.0)
27526                            <1>       ; 02/09/2011 (TRDOS v1, MAINPROG.ASM)
27527                            <1>       ; 28/08/2011
27528                            <1>       ; INPUT->
27529                            <1>       ;      EDI = Output buffer
27530                            <1>       ;      CX = Buffer length (<= ENV_PAGE_SIZE)
27531                            <1>       ;
27532                            <1>       ;      AL > 0 = AL = String sequence number
27533                            <1>       ;      AL = 0 -> ESI = ASCIIZ Set word
27534                            <1>       ;             (environment variable)
```

```
                                <1>       ; OUTPUT ->
                                <1>       ;      ESI is not changed
                                <1>       ;      EDI is not changed
                                <1>       ;      EAX = String length (with zero tail)
                                <1>       ;      EDX = Environment variables page address
                                <1>       ;      CF = 1 -> Not found (EAX not valid)
                                <1>       ;
                                <1>       ; (Modified registers: EAX, EDX)
                                <1>
0000928F BA00300900             <1>       mov    edx, Env_Page
00009294 803A00                 <1>       cmp    byte [edx], 0
00009297 7474                   <1>       jz     short get_env_string_with_word_stc_retn
                                <1>
00009299 66890D[FC5A0100]       <1>       mov    [env_var_length], cx
                                <1>
000092A0 51                     <1>       push   ecx ; *
000092A1 56                     <1>       push   esi ; **
                                <1>
000092A2 08C0                   <1>       or     al, al
000092A4 7449                   <1>       jz     short get_env_string_with_word
                                <1>
                                <1> get_env_string_with_seq_number:
000092A6 B101                   <1>       mov    cl, 1
000092A8 88C5                   <1>       mov    ch, al
000092AA 31C0                   <1>       xor    eax, eax
000092AC 89D6                   <1>       mov    esi, edx ; Env_Page
                                <1>
                                <1> get_env_string_seq_number_check:
000092AE 38CD                   <1>       cmp    ch, cl
000092B0 7726                   <1>       ja     short get_env_string_seq_number_next
                                <1>
                                <1> get_env_string_move_to_buff:
000092B2 57                     <1>       push   edi ; ***
                                <1>
000092B3 29D2                   <1>       sub    edx, edx
                                <1>
                                <1> get_env_string_seq_number_repeat1:
000092B5 42                     <1>       inc    edx
000092B6 AC                     <1>       lodsb
000092B7 AA                     <1>       stosb
                                <1>
000092B8 66FF0D[FC5A0100]       <1>       dec    word [env_var_length]
000092BF 7508                   <1>       jnz    short get_env_string_seq_number_repeat3
                                <1>
                                <1> get_env_string_seq_number_repeat2:
000092C1 20C0                   <1>       and    al, al
000092C3 7408                   <1>       jz     short get_env_string_seq_number_ok
000092C5 42                     <1>       inc    edx
000092C6 AC                     <1>       lodsb
000092C7 EBF8                   <1>       jmp    short get_env_string_seq_number_repeat2
                                <1>
                                <1> get_env_string_seq_number_repeat3:
000092C9 08C0                   <1>       or     al, al
000092CB 75E8                   <1>       jnz    short get_env_string_seq_number_repeat1
                                <1>
                                <1> get_env_string_seq_number_ok:
000092CD 5F                     <1>       pop    edi ; ***
000092CE 89D0                   <1>       mov    eax, edx ; Length of the environment string
                                <1>                       ; (ASCIIZ, includes ZERO tail)
000092D0 BA00300900             <1>       mov    edx, Env_Page
                                <1>
                                <1> get_env_string_stc_retn:
000092D5 5E                     <1>       pop    esi ; **
000092D6 59                     <1>       pop    ecx ; *
000092D7 C3                     <1>       retn
                                <1>
                                <1> get_env_string_seq_number_next:
000092D8 AC                     <1>       lodsb
000092D9 08C0                   <1>       or     al, al
000092DB 75FB                   <1>       jnz    short get_env_string_seq_number_next
                                <1>
000092DD 81FE00320900           <1>       cmp    esi, Env_Page + Env_Page_Size ; +512 (+4096)
000092E3 F5                     <1>       cmc
000092E4 72EF                   <1>       jc     short get_env_string_stc_retn
                                <1>
000092E6 AC                     <1>       lodsb
000092E7 3C01                   <1>       cmp    al, 1
000092E9 72EA                   <1>       jb     short get_env_string_stc_retn
000092EB FEC1                   <1>       inc    cl
000092ED EBBF                   <1>       jmp    short get_env_string_seq_number_check
                                <1>
                                <1> get_env_string_with_word:
000092EF 31C9                   <1>       xor    ecx, ecx
                                <1>
                                <1> get_env_string_calc_word_length:
000092F1 AC                     <1>       lodsb
000092F2 3C20                   <1>       cmp    al, 20h
000092F4 7211                   <1>       jb     short get_env_string_calc_word_length_ok
                                <1>       ;inc   cx
000092F6 FEC1                   <1>       inc    cl
                                <1>
000092F8 3C61                   <1>       cmp    al, 'a'
000092FA 72F5                   <1>       jb     short get_env_string_calc_word_length
000092FC 3C7A                   <1>       cmp    al, 'z'
000092FE 77F1                   <1>       ja     short get_env_string_calc_word_length
00009300 24DF                   <1>       and    al, 0DFh
00009302 8846FF                 <1>       mov    [esi-1], al
00009305 EBEA                   <1>       jmp    short get_env_string_calc_word_length
                                <1>
                                <1> get_env_string_calc_word_length_ok:
00009307 08C9                   <1>       or     cl, cl
00009309 7506                   <1>       jnz    short get_env_string_calc_word_length_save
```

```
27637                                  <1>
27638 0000930B 5E                      <1>       pop    esi ; **
27639                                  <1>
27640                                  <1> get_env_string_stc_retn1:
27641 0000930C 59                      <1>       pop    ecx ; *
27642                                  <1>
27643                                  <1> get_env_string_with_word_stc_retn:
27644 0000930D 31C0                    <1>       xor    eax, eax
27645 0000930F F9                      <1>       stc
27646 00009310 C3                      <1>       retn
27647                                  <1>
27648                                  <1> get_env_string_calc_word_length_save:
27649 00009311 871C24                  <1>       xchg   ebx, [esp] ; **
27650 00009314 89DE                    <1>       mov    esi, ebx
27651                                  <1>              ; Start of the env string (to be searched)
27652                                  <1>
27653 00009316 57                      <1>       push   edi ; ***
27654 00009317 89D7                    <1>       mov    edi, edx ; Env_Page
27655                                  <1>
27656                                  <1> get_env_string_compare:
27657 00009319 57                      <1>       push   edi ; ****
27658 0000931A 51                      <1>       push   ecx ; ***** ; Variable name length
27659                                  <1>
27660                                  <1> get_env_string_compare_rep:
27661 0000931B AC                      <1>       lodsb
27662 0000931C AE                      <1>       scasb
27663 0000931D 7511                    <1>       jne    short get_env_string_compare_next1
27664 0000931F E2FA                    <1>       loop   get_env_string_compare_rep
27665                                  <1>
27666 00009321 803F3D                  <1>       cmp    byte [edi], '='
27667 00009324 750A                    <1>       jne    short get_env_string_compare_next1
27668                                  <1>
27669 00009326 59                      <1>       pop    ecx ; *****
27670 00009327 5F                      <1>       pop    edi ; ****
27671 00009328 89FE                    <1>       mov    esi, edi
27672 0000932A 5F                      <1>       pop    edi ; ***
27673 0000932B 871C24                  <1>       xchg   ebx, [esp] ; **
27674 0000932E EB82                    <1>       jmp    short get_env_string_move_to_buff
27675                                  <1>
27676                                  <1> get_env_string_compare_next1:
27677 00009330 89FE                    <1>       mov    esi, edi
27678 00009332 59                      <1>       pop    ecx ; *****
27679 00009333 5F                      <1>       pop    edi ; ****
27680                                  <1> get_env_string_compare_next2:
27681 00009334 81FEFF310900            <1>       cmp    esi, Env_Page + Env_Page_Size - 1 ; +511 (+4095)
27682 0000933A 7310                    <1>       jnb    short get_env_string_compare_not_ok
27683 0000933C 20C0                    <1>       and    al, al
27684 0000933E AC                      <1>       lodsb
27685 0000933F 75F3                    <1>       jnz    short get_env_string_compare_next2
27686 00009341 08C0                    <1>       or     al, al
27687 00009343 7407                    <1>       jz     short get_env_string_compare_not_ok
27688 00009345 4E                      <1>       dec    esi ; 12/04/2016
27689 00009346 89F7                    <1>       mov    edi, esi
27690 00009348 89DE                    <1>       mov    esi, ebx
27691 0000934A EBCD                    <1>       jmp    short get_env_string_compare
27692                                  <1>
27693                                  <1> get_env_string_compare_not_ok:
27694 0000934C 5F                      <1>       pop    edi ; ***
27695 0000934D 89DE                    <1>       mov    esi, ebx
27696 0000934F 5B                      <1>       pop    ebx ; **
27697 00009350 EBBA                    <1>       jmp    short get_env_string_stc_retn1
27698                                  <1>
27699                                  <1> set_environment_string:
27700                                  <1>       ; 13/04/2016
27701                                  <1>       ; 12/04/2016
27702                                  <1>       ; 11/04/2016
27703                                  <1>       ; 06/04/2016
27704                                  <1>       ; 05/04/2016 (TRDOS 386 = TRDOS v2.0)
27705                                  <1>       ; 02/09/2011 (TRDOS v1, MAINPROG.ASM)
27706                                  <1>       ; 29/08/2011
27707                                  <1>       ; 29/08/2011
27708                                  <1>       ; INPUT->
27709                                  <1>       ;      ESI = ASCIIZ environment string
27710                                  <1>       ; OUTPUT ->
27711                                  <1>       ;      ESI is not changed
27712                                  <1>       ;      CF = 1 -> Could not set,
27713                                  <1>       ;           insufficient environment space
27714                                  <1>       ;
27715                                  <1>       ; (EAX, EDX will be changed)
27716                                  <1>       ;
27717                                  <1>       ;    (EAX = Start address of the env string if > 0)
27718                                  <1>       ;    (EDX = Environment string length)
27719                                  <1>
27720 00009352 56                      <1>       push   esi ; *
27721                                  <1>
27722 00009353 31C0                    <1>       xor    eax, eax
27723                                  <1>
27724                                  <1> set_env_chk_validation1:
27725 00009355 FEC4                    <1>       inc    ah ; variable (string) length
27726 00009357 AC                      <1>       lodsb
27727 00009358 3C3D                    <1>       cmp    al, '='
27728 0000935A 7415                    <1>       je     short set_env_chk_validation2
27729 0000935C 3C20                    <1>       cmp    al, 20h
27730 0000935E 720F                    <1>       jb     short set_env_string_stc
27731                                  <1>
27732                                  <1>       ; 06/04/2016
27733 00009360 3C61                    <1>       cmp    al, 'a'
27734 00009362 72F1                    <1>       jb     short set_env_chk_validation1
27735 00009364 3C7A                    <1>       cmp    al, 'z'
27736 00009366 77ED                    <1>       ja     short set_env_chk_validation1
27737 00009368 2C20                    <1>       sub    al, 'a'-'A'
27738 0000936A 8846FF                  <1>       mov    [esi-1], al
```

```
27739 0000936D EBE6               <1>       jmp    short set_env_chk_validation1
27740                             <1>
27741                             <1> set_env_string_stc:
27742 0000936F 5E                 <1>       pop    esi ; *
27743                             <1>       ;stc
27744 00009370 C3                 <1>       retn
27745                             <1>
27746                             <1> set_env_chk_validation2:
27747 00009371 51                 <1>       push   ecx ; **
27748 00009372 53                 <1>       push   ebx ; ***
27749 00009373 57                 <1>       push   edi ; ****
27750                             <1>
27751                             <1>       ; 12/04/2016
27752 00009374 8B5C240C           <1>       mov    ebx, [esp+12]
27753                             <1>
27754                             <1> set_env_chk_validation2w:
27755 00009378 89F7               <1>       mov    edi, esi
27756 0000937A 4F                 <1>       dec    edi
27757                             <1>
27758 0000937B 807FFF20           <1>       cmp    byte [edi-1], 20h
27759 0000937F 771A               <1>       ja     short set_env_chk_validation2z
27760                             <1>
27761 00009381 56                 <1>       push   esi
27762 00009382 89FE               <1>       mov    esi, edi
27763 00009384 4E                 <1>       dec    esi
27764                             <1>
27765                             <1> set_env_chk_validation2x:
27766 00009385 4E                 <1>       dec    esi
27767                             <1>
27768 00009386 39DE               <1>       cmp    esi, ebx
27769 00009388 7207               <1>       jb     short set_env_chk_validation2y
27770                             <1>
27771 0000938A 4F                 <1>       dec    edi
27772                             <1>
27773 0000938B 8A06               <1>       mov    al, [esi]
27774 0000938D 8807               <1>       mov    [edi], al
27775                             <1>
27776 0000938F EBF4               <1>       jmp    short set_env_chk_validation2x
27777                             <1>
27778                             <1> set_env_chk_validation2y:
27779 00009391 5E                 <1>       pop    esi
27780                             <1>
27781                             <1>       ;mov   byte [ebx], 20h
27782                             <1>
27783 00009392 43                 <1>       inc    ebx
27784 00009393 895C240C           <1>       mov    [esp+12], ebx
27785                             <1>
27786 00009397 FECC               <1>       dec    ah ; 13/04/2016
27787                             <1>
27788 00009399 EBDD               <1>       jmp    short set_env_chk_validation2w
27789                             <1>
27790                             <1> set_env_chk_validation2z:
27791 0000939B BA00300900         <1>       mov    edx, Env_Page
27792 000093A0 89D7               <1>       mov    edi, edx
27793                             <1>
27794                             <1> set_env_chk_validation3:
27795 000093A2 AC                 <1>       lodsb
27796 000093A3 3C20               <1>       cmp    al, 20h
27797 000093A5 74FB               <1>       je     short set_env_chk_validation3
27798                             <1>
27799 000093A7 9C                 <1>       pushf
27800                             <1>
27801                             <1>       ; 12/04/2016
27802                             <1> set_env_chk_validation3n:
27803 000093A8 3C61               <1>       cmp    al, 'a'
27804 000093AA 720C               <1>       jb     short set_env_chk_validation3c
27805 000093AC 3C7A               <1>       cmp    al, 'z'
27806 000093AE 7705               <1>       ja     short set_env_chk_validation3x
27807 000093B0 2C20               <1>       sub    al, 'a'-'A'
27808 000093B2 8846FF             <1>       mov    [esi-1], al
27809                             <1>
27810                             <1> set_env_chk_validation3x:
27811 000093B5 AC                 <1>       lodsb
27812 000093B6 EBF0               <1>       jmp    short set_env_chk_validation3n
27813                             <1>
27814                             <1> set_env_chk_validation3c:
27815 000093B8 3C20               <1>       cmp    al, 20h
27816 000093BA 73F9               <1>       jnb    short set_env_chk_validation3x
27817                             <1>
27818 000093BC 803F00             <1>       cmp    byte [edi], 0
27819 000093BF 7731               <1>       ja     short set_env_chk_validation4
27820                             <1>
27821 000093C1 9D                 <1>       popf
27822 000093C2 7228               <1>       jb     short set_env_string_nothing
27823                             <1>
27824 000093C4 B900020000         <1>       mov    ecx, Env_Page_Size ; 512 (4096)
27825                             <1>
27826 000093C9 89DE               <1>       mov    esi, ebx ; 12/04/2016
27827                             <1>
27828                             <1> set_env_string_copy_to_envb:
27829 000093CB AC                 <1>       lodsb
27830 000093CC 3C20               <1>       cmp    al, 20h
27831 000093CE 720A               <1>       jb     short set_env_string_copy_to_envb_z
27832 000093D0 AA                 <1>       stosb
27833 000093D1 E2F8               <1>       loop   set_env_string_copy_to_envb
27834                             <1>
27835                             <1>       ; 11/04/2016
27836 000093D3 89D7               <1>       mov    edi, edx ; Env_Page
27837 000093D5 B900020000         <1>       mov    ecx, Env_Page_Size
27838                             <1>
27839                             <1> set_env_string_copy_to_envb_z:
27840 000093DA 52                 <1>       push   edx  ; Start address of the variable
```

```
27841 000093DB BA00020000          <1>       mov    edx, Env_Page_Size
27842 000093E0 29CA                <1>       sub    edx, ecx ; variable (string) length
27843                              <1>
27844 000093E2 28C0                <1>       sub    al, al ; 0
27845 000093E4 F3AA                <1>       rep    stosb ; clear remain bytes of the env page
27846                              <1>
27847 000093E6 58                  <1>       pop    eax  ; Start address of the variable
27848                              <1>
27849                              <1> set_env_string_allocate_envb_retn:  ; stc or clc return
27850 000093E7 5F                  <1>       pop    edi ; ****
27851 000093E8 5B                  <1>       pop    ebx ; ***
27852 000093E9 59                  <1>       pop    ecx ; **
27853 000093EA 5E                  <1>       pop    esi ; *
27854 000093EB C3                  <1>       retn
27855                              <1>
27856                              <1> set_env_string_nothing:
27857 000093EC 31C0                <1>       xor    eax, eax
27858 000093EE 31D2                <1>       xor    edx, edx ; 11/04/2016
27859 000093F0 EBF5                <1>       jmp    short set_env_string_allocate_envb_retn
27860                              <1>
27861                              <1> set_env_chk_validation4:
27862                              <1>       ; 11/04/2016
27863 000093F2 9D                  <1>       popf
27864                              <1>
27865 000093F3 89D6                <1>       mov    esi, edx  ; Env_Page
27866                              <1>
27867                              <1> set_env_chk_validation5:
27868 000093F5 89DF                <1>       mov    edi, ebx  ; ASCIIZ environment string address
27869 000093F7 0FB6CC              <1>       movzx  ecx, ah ; Variable (string) length (with '=')
27870                              <1>
27871                              <1> set_env_chk_validation5_loop:
27872 000093FA AC                  <1>       lodsb
27873 000093FB AE                  <1>       scasb
27874 000093FC 750A                <1>       jne    short set_env_chk_validation6
27875 000093FE E2FA                <1>       loop   set_env_chk_validation5_loop
27876                              <1>
27877 00009400 3C3D                <1>       cmp    al, '='
27878 00009402 0F8483000000        <1>         je      set_env_change_variable
27879                              <1>
27880                              <1> set_env_chk_validation6:
27881 00009408 08C0                <1>       or     al, al ; 0
27882 0000940A 7403                <1>       jz     short set_env_chk_validation7
27883                              <1>
27884 0000940C AC                  <1>       lodsb
27885 0000940D EBF9                <1>       jmp    short set_env_chk_validation6
27886                              <1>
27887                              <1> set_env_chk_validation7:
27888 0000940F 88E1                <1>       mov    cl, ah
27889 00009411 01F1                <1>       add    ecx, esi
27890 00009413 81F9FF310900        <1>       cmp    ecx, Env_Page + Env_Page_Size - 1
27891                              <1>              ; 511 (4095)
27892                              <1>              ; strlen + '=' + 0
27893 00009419 72DA                <1>       jb     short set_env_chk_validation5
27894                              <1>
27895                              <1> set_env_chk_validation8: ; variable not found
27896 0000941B 0FB6F4              <1>       movzx  esi, ah  ; variable name length (with '=')
27897 0000941E 01DE                <1>       add    esi, ebx ; position just after of the '='
27898                              <1>
27899                              <1> set_env_chk_validation8_loop:
27900 00009420 AC                  <1>       lodsb
27901 00009421 3C20                <1>       cmp    al, 20h
27902 00009423 74FB                <1>       je     short set_env_chk_validation8_loop
27903 00009425 72C5                <1>       jb     short set_env_string_nothing
27904                              <1>
27905                              <1> set_env_chk_validation9:
27906 00009427 AC                  <1>       lodsb
27907 00009428 3C20                <1>       cmp    al, 20h
27908 0000942A 73FB                <1>       jnb    short set_env_chk_validation9
27909                              <1>
27910                              <1>       ; End of ASCIIZ environment string
27911                              <1>
27912                              <1> set_env_add_variable:
27913 0000942C 29DE                <1>       sub    esi, ebx ; variable+definition length
27914                              <1>
27915 0000942E 56                  <1>       push   esi ; *****
27916                              <1>
27917 0000942F 89D6                <1>       mov    esi, edx ; Environment page address
27918                              <1>
27919 00009431 B900020000          <1>       mov    ecx, Env_Page_Size ; 512 (4096)
27920                              <1>
27921                              <1> set_env_add_variable_loop:
27922 00009436 AC                  <1>       lodsb
27923 00009437 20C0                <1>       and    al, al
27924 00009439 7406                <1>       jz     short set_env_add_variable_chk1 ; 0
27925 0000943B E2F9                <1>       loop   set_env_add_variable_loop
27926                              <1>
27927                              <1>       ; 11/04/2016
27928 0000943D 884EFF              <1>       mov    [esi-1], cl ; 0
27929 00009440 41                  <1>       inc    ecx
27930                              <1>
27931                              <1> set_env_add_variable_chk1:
27932 00009441 49                  <1>       dec    ecx
27933 00009442 7408                <1>       jz     short set_env_add_variable_nspc
27934 00009444 AC                  <1>       lodsb
27935 00009445 08C0                <1>       or     al, al
27936 00009447 740C                <1>       jz     short set_env_add_variable_chk2 ; 00
27937 00009449 49                  <1>       dec    ecx
27938 0000944A 75EA                <1>       jnz    short set_env_add_variable_loop
27939                              <1>
27940                              <1> set_env_add_variable_nspc: ; no space on environment page
27941 0000944C 58                  <1>       pop    eax ; *****
27942 0000944D B808000000          <1>       mov    eax, 8 ; No space for new environment string
```

```
27943 00009452 F9                <1>       stc
27944 00009453 EB92              <1>         jmp     short set_env_string_allocate_envb_retn
27945                            <1>
27946                            <1> set_env_add_variable_chk2:
27947 00009455 8B0C24            <1>       mov    ecx, [esp] ; *****
27948 00009458 4E                <1>       dec    esi ; beginning address of the new variable
27949 00009459 89F0              <1>       mov    eax, esi
27950 0000945B 01C8              <1>       add    eax, ecx ; string length (with CR)
27951 0000945D 81C200020000      <1>       add    edx, Env_Page_Size ; 512 (4096)
27952 00009463 39D0              <1>       cmp    eax, edx
27953 00009465 77E5              <1>       ja     short set_env_add_variable_nspc
27954 00009467 49                <1>       dec    ecx ; except CR at the end
27955 00009468 89CA              <1>       mov    edx, ecx ; 12/04/2016
27956 0000946A 89F7              <1>       mov    edi, esi
27957 0000946C 893C24            <1>       mov    [esp], edi ; ***** ; Start address of new variable
27958 0000946F 89DE              <1>       mov    esi, ebx ; ASCIIZ environment string address
27959 00009471 F3A4              <1>       rep    movsb
27960 00009473 28C0              <1>       sub    al, al
27961 00009475 AA                <1>       stosb
27962 00009476 58                <1>       pop    eax ; ***** ; Beginning address of new variable
27963 00009477 81FF00320900      <1>         cmp     edi, Env_Page + Env_Page_Size ; 12/04/2016
27964 0000947D 0F8364FFFFFF      <1>         jnb     set_env_string_allocate_envb_retn ; OK !
27965 00009483 880F              <1>       mov    [edi], cl ; 0
27966 00009485 F8                <1>       clc    ; 13/04/2016
27967 00009486 E95CFFFFFF        <1>         jmp     set_env_string_allocate_envb_retn ; OK !
27968                            <1>
27969                            <1> set_env_change_variable:
27970                            <1>       ; 06/04/2016
27971                            <1>       ; esi = Variable's address in environment page (after '=')
27972                            <1>       ; edi = ASCIIZ environment string address (after '=')
27973                            <1>
27974                            <1>       ; ah = variable length from start to the '='
27975 0000948B 8825[FC5A0100]    <1>       mov    [env_var_length], ah
27976                            <1>
27977 00009491 28C9              <1>       sub    cl, cl ; ecx = 0
27978                            <1>
27979 00009493 57                <1>       push   edi ; *****
27980                            <1>
27981 00009494 89F7              <1>       mov    edi, esi ; 11/04/2016
27982                            <1>
27983                            <1> set_env_change_variable_calc1:
27984 00009496 AC                <1>       lodsb
27985 00009497 08C0              <1>       or     al, al
27986 00009499 7403              <1>       jz     short set_env_change_variable_calc2
27987                            <1>
27988 0000949B 41                <1>       inc    ecx ; length of environment string (after the '=')
27989                            <1>
27990 0000949C EBF8              <1>       jmp    short set_env_change_variable_calc1
27991                            <1>
27992                            <1> set_env_change_variable_calc2:
27993 0000949E 8B3424            <1>       mov    esi, [esp] ; ASCIIZ environment string address
27994                            <1>
27995 000094A1 29D2              <1>       sub    edx, edx
27996                            <1>
27997                            <1> set_env_change_variable_calc3:
27998 000094A3 AC                <1>       lodsb
27999 000094A4 3C20              <1>       cmp    al, 20h
28000 000094A6 7203              <1>       jb     short set_env_change_variable_calc4
28001                            <1>
28002 000094A8 42                <1>       inc    edx ; length of ASCIIZ string (after the '=')
28003                            <1>
28004 000094A9 EBF8              <1>       jmp    short set_env_change_variable_calc3
28005                            <1>
28006                            <1> set_env_change_variable_calc4:
28007 000094AB C646FF00          <1>       mov    byte [esi-1], 0  ; put ZERO instead of CR
28008                            <1>
28009 000094AF 5E                <1>       pop    esi ; ***** ; ASCIIZ string address (after '=')
28010                            <1>
28011                            <1>       ; EDI = Old variable's address (after '=')
28012                            <1>
28013                            <1>       ; compare the new string with the old string
28014 000094B0 39CA              <1>       cmp    edx, ecx
28015 000094B2 7717              <1>       ja     short set_env_change_variable_calc5 ; longer
28016 000094B4 0F828F000000      <1>         jb      set_env_change_variable_calc9 ; shorter
28017                            <1>
28018                            <1>       ;same length (simple copy)
28019 000094BA 0FB6C4            <1>       movzx  eax, ah
28020 000094BD 01C2              <1>       add    edx, eax
28021 000094BF F7D8              <1>       neg    eax
28022 000094C1 01F8              <1>       add    eax, edi
28023                            <1>       ; EAX = Start address of the variable
28024                            <1>       ; EDX = Variable length (without ZERO at the end of variable)
28025                            <1>
28026 000094C3 F3A4              <1>       rep    movsb
28027 000094C5 F8                <1>       clc    ; 13/04/2016
28028 000094C6 E91CFFFFFF        <1>         jmp     set_env_string_allocate_envb_retn ; OK !
28029                            <1>
28030                            <1> set_env_change_variable_calc5:
28031                            <1>       ; 11/04/2016
28032 000094CB 52                <1>       push   edx ; *****
28033 000094CC 29CA              <1>       sub    edx, ecx ; difference ; (the new string is longer)
28034 000094CE 89F3              <1>       mov    ebx, esi
28035 000094D0 89FE              <1>       mov    esi, edi
28036                            <1>
28037                            <1> set_env_change_variable_calc6:
28038 000094D2 AC                <1>       lodsb
28039 000094D3 20C0              <1>       and    al, al
28040 000094D5 75FB              <1>       jnz    short set_env_change_variable_calc6
28041                            <1>
28042 000094D7 81FE00320900      <1>       cmp    esi, Env_Page + Env_Page_Size ; 512 (4096)
28043 000094DD 0F8369FFFFFF      <1>         jnb     set_env_add_variable_nspc
28044                            <1>
```

```
28045 000094E3 89F9              <1>       mov    ecx, edi  ; current (old) variable's address
28046 000094E5 89F7              <1>       mov    edi, esi  ; next variable's address
28047                            <1>
28048 000094E7 AC                <1>       lodsb
28049 000094E8 08C0              <1>       or     al, al
28050 000094EA 7416              <1>       jz     short set_env_change_variable_calc8 ; 00
28051                            <1>
28052                            <1> set_env_change_variable_calc7:
28053 000094EC AC                <1>       lodsb
28054 000094ED 20C0              <1>       and    al, al
28055 000094EF 75FB              <1>       jnz    short set_env_change_variable_calc7
28056                            <1>
28057 000094F1 81FE00320900      <1>       cmp    esi, Env_Page + Env_Page_Size ; 512 (4096)
28058 000094F7 0F834FFFFFFF      <1>         jnb     set_env_add_variable_nspc
28059                            <1>
28060 000094FD AC                <1>       lodsb
28061 000094FE 08C0              <1>       or     al, al
28062 00009500 75EA              <1>       jnz    short set_env_change_variable_calc7
28063                            <1>
28064                            <1> set_env_change_variable_calc8:
28065 00009502 4E                <1>       dec    esi ; address of the second (last) 0 of the 00
28066                            <1>
28067 00009503 01F2              <1>       add    edx, esi ; final position of the last 0
28068                            <1>
28069 00009505 81FA00320900      <1>       cmp    edx, Env_Page + Env_Page_Size ; 512 (4096)
28070 0000950B 0F833BFFFFFF      <1>         jnb     set_env_add_variable_nspc
28071                            <1>
28072 00009511 89C8              <1>       mov    eax, ecx ; old variable's address (after '=')
28073                            <1>
28074 00009513 89F1              <1>       mov    ecx, esi
28075 00009515 29F9              <1>       sub    ecx, edi ; count of bytes to move forward
28076                            <1>
28077                            <1>       ; 13/04/2016
28078 00009517 C60200            <1>       mov    byte [edx], 0
28079 0000951A 89D7              <1>       mov    edi, edx
28080 0000951C 29F2              <1>       sub    edx, esi ; difference (additional byte count)
28081 0000951E 4F                <1>       dec    edi ; the last zero address (first byte of the 00)
28082 0000951F 89FE              <1>       mov    esi, edi
28083 00009521 29D6              <1>       sub    esi, edx ; - displacement
28084                            <1>
28085 00009523 FA                <1>       cli    ; disable interrupts
28086 00009524 FD                <1>       std    ; backward
28087                            <1>
28088 00009525 F3A4              <1>       rep    movsb ; move ECX bytes from DS:ESI to ES:EDI
28089                            <1>
28090 00009527 FC                <1>       cld    ; forward (default)
28091 00009528 FB                <1>       sti    ; enable interrupts
28092                            <1>
28093 00009529 89C7              <1>       mov    edi, eax
28094 0000952B 59                <1>       pop    ecx ; ***** ; byte count (after '=')
28095 0000952C 89CA              <1>       mov    edx, ecx
28096 0000952E 89DE              <1>       mov    esi, ebx ; ASCIIZ string address (after '=')
28097 00009530 89FB              <1>       mov    ebx, edi
28098                            <1>
28099 00009532 F3A4              <1>       rep    movsb
28100                            <1>
28101 00009534 880F              <1>       mov    [edi], cl ; 0 ; end of variable
28102                            <1>
28103 00009536 0FB605[FC5A0100]  <1>       movzx  eax, byte [env_var_length]
28104 0000953D 01C2              <1>       add    edx, eax ; variable length (total)
28105 0000953F F7D8              <1>       neg    eax
28106 00009541 01D8              <1>       add    eax, ebx ; start address of the variable
28107 00009543 F8                <1>       clc    ; 13/04/2016
28108 00009544 E99EFEFFFF        <1>         jmp     set_env_string_allocate_envb_retn ; OK !
28109                            <1>
28110                            <1> set_env_change_variable_calc9:
28111                            <1>       ; 11/04/2016
28112 00009549 21D2              <1>       and    edx, edx ; is empty ?
28113 0000954B 753B              <1>       jnz    short set_env_change_variable_calc15
28114                            <1>
28115 0000954D 0FB6DC            <1>       movzx  ebx, ah
28116 00009550 F7DB              <1>       neg    ebx
28117 00009552 01FB              <1>       add    ebx, edi
28118                            <1>
28119                            <1>       ; EBX = Start address of the variable (in env page)
28120                            <1>       ; EDX = Variable length = 0
28121                            <1>
28122 00009554 89FE              <1>       mov    esi, edi
28123                            <1>
28124                            <1> set_env_change_variable_calc10:
28125 00009556 AC                <1>       lodsb
28126 00009557 08C0              <1>       or     al, al
28127 00009559 75FB              <1>       jnz    short set_env_change_variable_calc10
28128                            <1>
28129 0000955B B9FF310900        <1>       mov    ecx, Env_Page + Env_Page_Size - 1
28130                            <1>
28131 00009560 39CE              <1>       cmp    esi, ecx ; +511 (+4095)
28132 00009562 7604              <1>       jna    short set_env_change_variable_calc11
28133                            <1>
28134 00009564 89CE              <1>       mov    esi, ecx
28135 00009566 8806              <1>       mov    [esi], al ; 0
28136                            <1>
28137                            <1> set_env_change_variable_calc11:
28138 00009568 89DF              <1>       mov    edi, ebx ; old variable's start address
28139                            <1>
28140                            <1> set_env_change_variable_calc12:
28141 0000956A AC                <1>       lodsb
28142 0000956B AA                <1>       stosb
28143 0000956C 20C0              <1>       and    al, al
28144 0000956E 75FA              <1>       jnz    short set_env_change_variable_calc12
28145 00009570 39CE              <1>       cmp    esi, ecx
28146 00009572 7706              <1>       ja     short set_env_change_variable_calc13
```

```
28147 00009574 AC                <1>       lodsb
28148 00009575 AA                <1>       stosb
28149 00009576 20C0              <1>       and    al, al
28150 00009578 75F0              <1>       jnz    short set_env_change_variable_calc12
28151                            <1>
28152                            <1> set_env_change_variable_calc13:
28153 0000957A 29F9              <1>       sub    ecx, edi
28154 0000957C 7203              <1>       jb     short set_env_change_variable_calc14
28155 0000957E 41                <1>       inc    ecx ; 1-512 (1-4096)
28156 0000957F F3AA              <1>       rep    stosb ; al = 0
28157                            <1>
28158                            <1> set_env_change_variable_calc14:
28159 00009581 29C0              <1>       sub    eax, eax ; Start address of the variable
28160                            <1>       ; EAX = 0 -> Variable is removed
28161                            <1>       ; EDX = Variable length = 0
28162                            <1>
28163 00009583 E95FFEFFFF        <1>         jmp     set_env_string_allocate_envb_retn ; OK !
28164                            <1>
28165                            <1> set_env_change_variable_calc15:
28166 00009588 52                <1>       push   edx ; *****
28167 00009589 F7DA              <1>       neg    edx
28168 0000958B 01CA              <1>       add    edx, ecx ; difference (the old string is longer)
28169 0000958D 89F3              <1>       mov    ebx, esi
28170 0000958F 89FE              <1>       mov    esi, edi
28171                            <1>
28172                            <1> set_env_change_variable_calc16:
28173 00009591 AC                <1>       lodsb
28174 00009592 20C0              <1>       and    al, al
28175 00009594 75FB              <1>       jnz    short set_env_change_variable_calc16
28176                            <1>
28177 00009596 B900320900        <1>       mov    ecx, Env_Page + Env_Page_Size
28178                            <1>
28179 0000959B 39CE              <1>       cmp    esi, ecx ; +512 (+4096)
28180 0000959D 7605              <1>       jna    short set_env_change_variable_calc17
28181                            <1>
28182 0000959F 89CE              <1>       mov    esi, ecx
28183 000095A1 8846FF            <1>       mov    [esi-1], al ; 0
28184                            <1>
28185                            <1> set_env_change_variable_calc17:
28186 000095A4 89F9              <1>       mov    ecx, edi  ; current (old) variable's address
28187 000095A6 89F7              <1>       mov    edi, esi  ; next variable's address
28188                            <1>
28189 000095A8 AC                <1>       lodsb
28190 000095A9 08C0              <1>       or     al, al
28191 000095AB 741D              <1>       jz     short set_env_change_variable_calc20
28192                            <1>
28193                            <1> set_env_change_variable_calc18:
28194 000095AD AC                <1>       lodsb
28195 000095AE 20C0              <1>       and    al, al
28196 000095B0 75FB              <1>       jnz    short set_env_change_variable_calc18
28197                            <1>
28198 000095B2 81FE00320900      <1>       cmp    esi, Env_Page + Env_Page_Size
28199 000095B8 720B              <1>       jb     short set_env_change_variable_calc19
28200 000095BA 740E              <1>       je     short set_env_change_variable_calc20
28201                            <1>
28202 000095BC BEFF310900        <1>       mov    esi, Env_Page + Env_Page_Size - 1
28203 000095C1 8806              <1>       mov    [esi], al ; 0
28204 000095C3 EB06              <1>       jmp    short set_env_change_variable_calc21
28205                            <1>
28206                            <1> set_env_change_variable_calc19:
28207 000095C5 AC                <1>       lodsb
28208 000095C6 08C0              <1>       or     al, al
28209 000095C8 75E3              <1>       jnz    short set_env_change_variable_calc18
28210                            <1>
28211                            <1> set_env_change_variable_calc20:
28212 000095CA 4E                <1>       dec    esi ; address of the second (last) 0 of the 00
28213                            <1>
28214                            <1> set_env_change_variable_calc21:
28215                            <1>       ; edx = difference (byte count)
28216                            <1>
28217 000095CB 89C8              <1>       mov    eax, ecx ; old variable's address (after '=')
28218                            <1>
28219 000095CD 89F1              <1>       mov    ecx, esi
28220 000095CF 29F9              <1>       sub    ecx, edi ; count of bytes to move backward
28221                            <1>
28222 000095D1 89FE              <1>       mov    esi, edi ; next variable's address
28223 000095D3 29D7              <1>       sub    edi, edx ; (displacement)
28224                            <1>
28225 000095D5 F3A4              <1>       rep    movsb
28226                            <1>
28227 000095D7 880F              <1>       mov    [edi], cl ; 0 ; 00 ; end of environment variables
28228                            <1>
28229 000095D9 89C7              <1>       mov    edi, eax
28230 000095DB 5A                <1>       pop    edx ; ***** ; byte count (after '=')
28231 000095DC 89D1              <1>       mov    ecx, edx
28232 000095DE 89DE              <1>       mov    esi, ebx ; ASCIIZ string address (after '=')
28233 000095E0 89FB              <1>       mov    ebx, edi
28234                            <1>
28235 000095E2 F3A4              <1>       rep    movsb
28236                            <1>
28237 000095E4 880F              <1>       mov    [edi], cl ; 0 ; end of variable
28238                            <1>
28239 000095E6 0FB605[FC5A0100]  <1>       movzx  eax, byte [env_var_length]
28240 000095ED 01C2              <1>       add    edx, eax ; variable length (total)
28241 000095EF F7D8              <1>       neg    eax
28242 000095F1 01D8              <1>       add    eax, ebx ; start address of the variable
28243 000095F3 F8                <1>       clc    ; 13/04/2016
28244 000095F4 E9EEFDFFFF        <1>         jmp     set_env_string_allocate_envb_retn ; OK !
28245                            <1>
28246                            <1> mainprog_startup_configuration:
28247                            <1>       ; 06/05/2016
28248                            <1>       ; 14/04/2016 (TRDOS 386 = TRDOS v2.0)
```

```
28249                                  <1>       ; 17/09/2011 (TRDOS v1, MAINPROG.ASM)
28250                                  <1>       ;
28251                                  <1> loc_load_mainprog_cfg_file:
28252 000095F9 BE[DE020100]           <1>       mov    esi, MainProgCfgFile
28253 000095FE 66B80018               <1>       mov    ax, 1800h ; Except volume label and dirs
28254 00009602 E8E8E9FFFF             <1>       call   find_first_file
28255 00009607 7256                   <1>       jc     short loc_load_mainprog_cfg_exit
28256                                  <1>
28257                                  <1>       ;or    eax, eax
28258                                  <1>       ;jz    short loc_load_mainprog_cfg_exit
28259                                  <1>
28260                                  <1> loc_start_mainprog_configuration:
28261                                  <1>       ; ESI = FindFile_DirEntry Location
28262                                  <1>       ; EAX = File Size
28263                                  <1>
28264 00009609 A3[7C4E0100]           <1>       mov    [MainProgCfg_FileSize], eax
28265                                  <1>
28266 0000960E 668B5614               <1>       mov    dx, [esi+DirEntry_FstClusHI]
28267 00009612 C1E210                 <1>       shl    edx, 16
28268 00009615 668B561A               <1>       mov    dx, [esi+DirEntry_FstClusLO]
28269 00009619 8915[B05A0100]         <1>       mov    [csftdf_sf_cluster], edx
28270                                  <1>
28271 0000961F 89C1                   <1>       mov    ecx, eax
28272 00009621 29C0                   <1>       sub    eax, eax
28273                                  <1>
28274                                  <1>       ; TRDOS 386 (TRDOS v2.0)
28275                                  <1>       ; Allocate contiguous memory block for loading the file
28276                                  <1>
28277                                  <1>       ; eax = 0 (Allocate memory from the beginning)
28278                                  <1>       ; ecx = File (Allocation) size in bytes
28279                                  <1>
28280 00009623 E8FBBDFFFF             <1>       call   allocate_memory_block
28281 00009628 7235                   <1>       jc     short loc_load_mainprog_cfg_exit
28282                                  <1>
28283 0000962A A3[A85A0100]           <1>       mov    [csftdf_sf_mem_addr], eax ; loading address
28284 0000962F 890D[AC5A0100]         <1>       mov    [csftdf_sf_mem_bsize], ecx ; block size
28285                                  <1>
28286 00009635 31DB                   <1>       xor    ebx, ebx
28287                                  <1>       ;mov   [csftdf_sf_rbytes], ebx ; 0, reset
28288                                  <1>
28289 00009637 8A3D[8E4E0100]         <1>       mov    bh, [Current_Drv] ; [FindFile_Drv]
28290 0000963D BE00010900             <1>       mov    esi, Logical_DOSDisks
28291 00009642 01DE                   <1>       add    esi, ebx
28292                                  <1>
28293 00009644 8B1D[A85A0100]         <1>       mov    ebx, [csftdf_sf_mem_addr] ; memory block address
28294                                  <1>
28295 0000964A 807E0300               <1>       cmp    byte [esi+LD_FATType], 0
28296 0000964E 7710                   <1>         ja   short loc_mcfg_load_fat_file
28297                                  <1>
28298 00009650 C705[B85A0100]0000-    <1>       mov    dword [csftdf_r_size], 65536
28299 00009658 0100                   <1>
28300 0000965A E992010000             <1>         jmp     loc_mcfg_load_fs_file
28301                                  <1>
28302                                  <1> loc_load_mainprog_cfg_exit:
28303 0000965F C3                     <1>       retn
28304                                  <1>
28305                                  <1> loc_mcfg_load_fat_file:
28306 00009660 0FB74611               <1>       movzx  eax, word [esi+LD_BPB+BytesPerSec]
28307 00009664 0FB64E13               <1>       movzx  ecx, byte [esi+LD_BPB+SecPerClust]
28308 00009668 F7E1                   <1>       mul    ecx
28309 0000966A A3[B85A0100]           <1>       mov    [csftdf_r_size], eax
28310                                  <1>
28311                                  <1> loc_mcfg_load_fat_file_next:
28312 0000966F E813010000             <1>       call   mcfg_read_fat_file_sectors
28313 00009674 0F82F7000000           <1>         jc      mcfg_deallocate_mem
28314                                  <1>
28315 0000967A 09D2                   <1>       or     edx, edx ; edx > 0 -> EOF
28316 0000967C 74F1                   <1>       jz     short loc_mcfg_load_fat_file_next
28317                                  <1>
28318                                  <1> loc_mcfg_load_fat_file_ok:
28319                                  <1>       ; 06/05/2016
28320 0000967E C705[4C5B0100]-        <1>       mov    dword [mainprog_return_addr], loc_mcfg_ci_return_addr
28321 00009684 [32970000]             <1>
28322                                  <1>       ;
28323 00009688 8B35[A85A0100]         <1>       mov    esi, [csftdf_sf_mem_addr]
28324 0000968E 8935[804E0100]         <1>       mov    [MainProgCfg_LineOffset], esi
28325                                  <1>
28326 00009694 A1[7C4E0100]           <1>       mov    eax, [MainProgCfg_FileSize]
28327 00009699 89C2                   <1>       mov    edx, eax
28328 0000969B 01F2                   <1>       add    edx, esi
28329                                  <1>
28330                                  <1> loc_mcfg_process_next_line_check:
28331 0000969D 89C1                   <1>       mov    ecx, eax
28332                                  <1>
28333 0000969F 803E2A                 <1>       cmp    byte [esi], "*" ; Remark sign
28334 000096A2 7503                   <1>       jne    short loc_mcfg_process_next_line
28335 000096A4 46                     <1>       inc    esi
28336 000096A5 EB17                   <1>       jmp    short loc_move_mainprog_cfg_nl1
28337                                  <1>
28338                                  <1> loc_mcfg_process_next_line:
28339 000096A7 83F94F                 <1>       cmp    ecx, 79
28340 000096AA 7605                   <1>       jna    short loc_start_mainprog_cfg_process
28341                                  <1>
28342 000096AC B94F000000             <1>       mov    ecx, 79
28343                                  <1>
28344                                  <1> loc_start_mainprog_cfg_process:
28345 000096B1 BF[3E4F0100]           <1>       mov    edi, CommandBuffer
28346                                  <1>
28347                                  <1> loc_move_mainprog_cfg_line:
28348 000096B6 AC                     <1>       lodsb
28349 000096B7 3C20                   <1>       cmp    al, 20h
28350 000096B9 720C                   <1>       jb     short loc_move_mainprog_cfg_nl2
```

```
28351 000096BB AA                <1>       stosb
28352 000096BC E2F8              <1>       loop   loc_move_mainprog_cfg_line
28353                            <1>
28354                            <1> loc_move_mainprog_cfg_nl1:
28355 000096BE 39D6              <1>       cmp    esi, edx ; + configuration file size
28356 000096C0 7312              <1>       jnb    short loc_end_of_mainprog_cfg_line
28357 000096C2 AC                <1>       lodsb
28358 000096C3 3C20              <1>       cmp    al, 20h
28359 000096C5 73F7              <1>       jnb    short loc_move_mainprog_cfg_nl1
28360                            <1>
28361                            <1> loc_move_mainprog_cfg_nl2:
28362 000096C7 39D6              <1>       cmp    esi, edx
28363 000096C9 7309              <1>       jnb    short loc_end_of_mainprog_cfg_line
28364 000096CB 8A06              <1>       mov    al, [esi]
28365 000096CD 3C20              <1>       cmp    al, 20h
28366 000096CF 7703              <1>       ja     short loc_end_of_mainprog_cfg_line
28367 000096D1 46                <1>       inc    esi
28368 000096D2 EBF3              <1>       jmp    short loc_move_mainprog_cfg_nl2
28369                            <1>
28370                            <1> loc_end_of_mainprog_cfg_line:
28371 000096D4 C60700            <1>       mov    byte [edi], 0
28372                            <1>
28373 000096D7 8935[804E0100]    <1>       mov    [MainProgCfg_LineOffset], esi
28374                            <1>
28375                            <1> loc_move_mainprog_cfg_command:
28376 000096DD BE[3E4F0100]      <1>       mov    esi, CommandBuffer
28377 000096E2 89F7              <1>       mov    edi, esi
28378 000096E4 31DB              <1>       xor    ebx, ebx
28379                            <1>       ;xor   ecx, ecx
28380 000096E6 30C9              <1>       xor    cl, cl
28381                            <1>
28382                            <1> loc_move_mcfg_first_cmd_char:
28383 000096E8 8A041E            <1>       mov    al, [esi+ebx]
28384 000096EB FEC3              <1>       inc    bl
28385 000096ED 3C20              <1>       cmp    al, 20h
28386 000096EF 7712              <1>       ja     short loc_move_mcfg_cmd_capitalizing
28387 000096F1 7237              <1>       jb     short loc_move_mcfg_cmd_arguments_ok
28388 000096F3 80FB4F            <1>       cmp    bl, 79
28389 000096F6 72F0              <1>       jb     short loc_move_mcfg_first_cmd_char
28390 000096F8 EB30              <1>       jmp    short loc_move_mcfg_cmd_arguments_ok
28391                            <1>
28392                            <1> loc_move_mcfg_next_cmd_char:
28393 000096FA 8A041E            <1>       mov    al, [esi+ebx]
28394 000096FD FEC3              <1>       inc    bl
28395 000096FF 3C20              <1>       cmp    al, 20h
28396 00009701 7614              <1>       jna    short loc_move_mcfg_cmd_ok
28397                            <1>
28398                            <1> loc_move_mcfg_cmd_capitalizing:
28399 00009703 3C61              <1>       cmp    al, 61h ; 'a'
28400 00009705 7206              <1>       jb     short loc_move_mcfg_cmd_caps_ok
28401 00009707 3C7A              <1>       cmp    al, 7Ah ; 'z'
28402 00009709 7702              <1>       ja     short loc_move_mcfg_cmd_caps_ok
28403 0000970B 24DF              <1>       and    al, 0DFh ; sub      al, 'a'-'A'
28404                            <1>
28405                            <1> loc_move_mcfg_cmd_caps_ok:
28406 0000970D AA                <1>       stosb
28407 0000970E FEC1              <1>       inc    cl
28408 00009710 80FB4F            <1>       cmp    bl, 79
28409 00009713 72E5              <1>       jb     short loc_move_mcfg_next_cmd_char
28410 00009715 EB13              <1>       jmp    short loc_move_mcfg_cmd_arguments_ok
28411                            <1>
28412                            <1> loc_move_mcfg_cmd_ok:
28413 00009717 30C0              <1>       xor    al, al ; 0
28414                            <1>
28415                            <1> loc_move_mcfg_cmd_arguments:
28416 00009719 8807              <1>       mov    [edi], al
28417 0000971B 47                <1>       inc    edi
28418 0000971C 80FB4F            <1>       cmp    bl, 79
28419 0000971F 7309              <1>       jnb    short loc_move_mcfg_cmd_arguments_ok
28420 00009721 8A041E            <1>       mov    al, [esi+ebx]
28421 00009724 FEC3              <1>       inc    bl
28422 00009726 3C20              <1>       cmp    al, 20h
28423 00009728 73EF              <1>       jnb    short loc_move_mcfg_cmd_arguments
28424                            <1>
28425                            <1> loc_move_mcfg_cmd_arguments_ok:
28426 0000972A C60700            <1>       mov    byte [edi], 0
28427                            <1>
28428                            <1> loc_mcfg_process_cmd_interpreter:
28429 0000972D E8F4DFFFFF        <1>       call    command_interpreter
28430                            <1>
28431                            <1> loc_mcfg_ci_return_addr:
28432 00009732 A1[7C4E0100]      <1>       mov    eax, [MainProgCfg_FileSize]
28433 00009737 89C2              <1>       mov    edx, eax
28434 00009739 8B35[804E0100]    <1>       mov    esi, [MainProgCfg_LineOffset]
28435 0000973F 01F2              <1>       add    edx, esi
28436 00009741 0305[A85A0100]    <1>       add    eax, [csftdf_sf_mem_addr]
28437 00009747 29F0              <1>       sub    eax, esi
28438 00009749 0F874EFFFFFF      <1>         ja      loc_mcfg_process_next_line_check
28439                            <1>
28440 0000974F E81D000000        <1>       call   mcfg_deallocate_mem
28441                            <1>
28442 00009754 B94F000000        <1>       mov    ecx, 79 ; 80 ?
28443 00009759 BF[3E4F0100]      <1>       mov    edi, CommandBuffer
28444 0000975E 30C0              <1>       xor    al, al
28445 00009760 F3AA              <1>       rep    stosb
28446                            <1>
28447                            <1>       ; 06/05/2016
28448 00009762 BE[030F0100]      <1>       mov    esi, nextline
28449 00009767 E8F1CBFFFF        <1>       call   print_msg
28450 0000976C E92DD6FFFF        <1>       jmp    dos_prompt
28451                            <1>
28452                            <1> mcfg_deallocate_mem:
```

```
28453 00009771 A1[A85A0100]        <1>       mov    eax, [csftdf_sf_mem_addr] ; start address
28454 00009776 8B0D[AC5A0100]      <1>       mov    ecx, [csftdf_sf_mem_bsize] ; block size
28455                              <1>       ;call  deallocate_memory_block
28456                              <1>       ;retn
28457 0000977C E9AFBEFFFF          <1>       jmp    deallocate_memory_block
28458                              <1>
28459                              <1> mcfg_read_file_sectors:
28460                              <1>       ; 14/04/2016
28461 00009781 807E0300            <1>       cmp    byte [esi+LD_FATType], 0
28462 00009785 7669                <1>         jna  short mcfg_read_fs_file_sectors
28463                              <1>
28464                              <1> mcfg_read_fat_file_sectors:
28465                              <1>       ; return:
28466                              <1>       ;   CF = 0 & EDX > 0 -> END OF FILE
28467                              <1>       ;   CF = 0 & EDX = 0 -> not EOF
28468                              <1>       ;   CF = 1 -> read error (error code in AL)
28469                              <1>
28470                              <1> mcfg_read_fat_file_secs_0:
28471 00009787 8B15[7C4E0100]      <1>       mov    edx, [MainProgCfg_FileSize]
28472 0000978D 2B15[C05A0100]      <1>       sub    edx, [csftdf_sf_rbytes]
28473 00009793 3B15[B85A0100]      <1>       cmp    edx, [csftdf_r_size]
28474 00009799 7306                <1>       jnb    short mcfg_read_fat_file_secs_1
28475 0000979B 8915[B85A0100]      <1>       mov    [csftdf_r_size], edx
28476                              <1>
28477                              <1> mcfg_read_fat_file_secs_1:
28478 000097A1 A1[B85A0100]        <1>       mov    eax, [csftdf_r_size]
28479 000097A6 29D2                <1>       sub    edx, edx
28480 000097A8 0FB74E11            <1>       movzx  ecx, word [esi+LD_BPB+BytesPerSec]
28481 000097AC 01C8                <1>       add    eax, ecx
28482 000097AE 48                  <1>       dec    eax
28483 000097AF F7F1                <1>       div    ecx
28484 000097B1 89C1                <1>       mov    ecx, eax ; sector count
28485 000097B3 A1[B05A0100]        <1>       mov    eax, [csftdf_sf_cluster]
28486                              <1>
28487                              <1>       ; EBX = memory block address (current)
28488                              <1>
28489 000097B8 E88E230000          <1>       call   read_fat_file_sectors
28490 000097BD 7230                <1>       jc     short mcfg_read_fat_file_secs_3
28491                              <1>
28492                              <1>       ; EBX = next memory address
28493                              <1>
28494 000097BF A1[C05A0100]        <1>       mov    eax, [csftdf_sf_rbytes]
28495 000097C4 0305[B85A0100]      <1>       add    eax, [csftdf_r_size]
28496 000097CA 8B15[7C4E0100]      <1>       mov    edx, [MainProgCfg_FileSize]
28497 000097D0 39D0                <1>       cmp    eax, edx
28498 000097D2 731B                <1>       jnb    short mcfg_read_fat_file_secs_3 ; edx > 0
28499 000097D4 A3[C05A0100]        <1>       mov    [csftdf_sf_rbytes], eax
28500                              <1>
28501 000097D9 53                  <1>       push   ebx ; *
28502                              <1>       ; get next cluster (csftdf_r_size! bytes)
28503 000097DA A1[B05A0100]        <1>       mov    eax, [csftdf_sf_cluster]
28504 000097DF E839210000          <1>       call   get_next_cluster
28505 000097E4 5B                  <1>       pop    ebx ; *
28506 000097E5 7301                <1>       jnc    short mcfg_read_fat_file_secs_2
28507                              <1>
28508                              <1>       ;mov   eax, 17; Read error !
28509 000097E7 C3                  <1>       retn
28510                              <1>
28511                              <1> mcfg_read_fat_file_secs_2:
28512 000097E8 29D2                <1>       sub    edx, edx ; 0
28513 000097EA A3[B05A0100]        <1>       mov    [csftdf_sf_cluster], eax ; next cluster
28514                              <1>
28515                              <1> mcfg_read_fat_file_secs_3:
28516 000097EF C3                  <1>       retn
28517                              <1>
28518                              <1> mcfg_read_fs_file_sectors:
28519 000097F0 C3                  <1>       retn
28520                              <1>
28521                              <1> loc_mcfg_load_fs_file:
28522 000097F1 C3                  <1>       retn
28523                              <1>
28524                              <1> load_and_execute_file:
28525                              <1>       ; 04/01/2017
28526                              <1>       ; 06/05/2016, 07/05/2016, 11/05/2016
28527                              <1>       ; 23/04/2016, 24/04/2016
28528                              <1>       ; 22/04/2016 (TRDOS 386 = TRDOS v2.0)
28529                              <1>       ; 05/11/2011
28530                              <1>       ; (TRDOS v1, CMDINTR.ASM, 'cmp_cmd_run', 'cmp_cmd_external')
28531                              <1>       ; ('loc_run_check_filename')
28532                              <1>       ; 29/08/2011
28533                              <1>       ; 10/09/2011
28534                              <1>       ; INPUT->
28535                              <1>       ;     ESI = Path Name address (CommandBuffer address)
28536                              <1>       ; OUTPUT ->
28537                              <1>       ;     none (error message will be shown if an error will occur)
28538                              <1>       ;
28539                              <1>       ; (EAX, EBX, ECX, EDX, ESI, EDI, EBP will be changed)
28540                              <1>       ;
28541                              <1> loc_run_check_filename:
28542 000097F2 803E20              <1>       cmp    byte [esi], 20h
28543 000097F5 0F82F2E2FFFF        <1>       jb     loc_cmd_failed
28544 000097FB 7703                <1>       ja     short loc_run_check_filename_ok
28545 000097FD 46                  <1>       inc    esi
28546 000097FE EBF2                <1>       jmp    short loc_run_check_filename
28547                              <1>
28548                              <1> loc_run_check_filename_ok:
28549 00009800 C605[EF4E0100]00    <1>       mov    byte [CmdArgStart], 0 ; reset
28550 00009807 56                  <1>       push   esi ; *
28551                              <1> loc_run_get_first_arg_pos:
28552 00009808 46                  <1>       inc    esi
28553 00009809 8A06                <1>       mov    al, [esi]
28554 0000980B 3C20                <1>       cmp    al, 20h
```

```
28555 0000980D 77F9              <1>       ja     short loc_run_get_first_arg_pos
28556 0000980F C60600            <1>       mov    byte [esi], 0
28557                            <1> loc_run_get_external_arg_pos:
28558                            <1>       ; 11/05/2016
28559 00009812 46                <1>       inc    esi
28560 00009813 8A06              <1>       mov    al, [esi]
28561 00009815 3C20              <1>       cmp    al, 20h
28562 00009817 760C              <1>       jna    short loc_run_parse_path_name
28563 00009819 89F0              <1>       mov    eax, esi
28564 0000981B 2D[3E4F0100]      <1>       sub    eax, CommandBuffer
28565 00009820 A2[EF4E0100]      <1>       mov    byte [CmdArgStart], al
28566                            <1> loc_run_parse_path_name:
28567 00009825 5E                <1>       pop    esi ; *
28568 00009826 BF[32580100]      <1>       mov    edi, FindFile_Drv
28569 0000982B E8D6090000        <1>       call   parse_path_name
28570 00009830 0F82B7E2FFFF      <1>       jc     loc_cmd_failed
28571                            <1>
28572                            <1> loc_run_check_filename_exists:
28573 00009836 BE[74580100]      <1>       mov    esi, FindFile_Name
28574 0000983B 803E20            <1>       cmp    byte [esi], 20h
28575 0000983E 0F86A9E2FFFF      <1>       jna    loc_cmd_failed
28576                            <1>
28577                            <1> loc_run_check_exe_filename_ext:
28578 00009844 E891020000        <1>       call   check_prg_filename_ext
28579 00009849 0F829EE2FFFF      <1>       jc     loc_cmd_failed
28580                            <1>
28581                            <1> loc_run_check_exe_filename_ext_ok:
28582 0000984F 66A3[4A5B0100]    <1>       mov    word [EXE_ID], ax
28583                            <1>
28584                            <1> loc_run_drv:
28585 00009855 C605[495B0100]00  <1>       mov    byte [Run_Manual_Path], 0
28586 0000985C A1[884E0100]      <1>       mov    eax, [Current_Dir_FCluster]
28587 00009861 A3[445B0100]      <1>         mov     [Run_CDirFC], eax
28588                            <1>       ;
28589 00009866 8A35[8E4E0100]    <1>       mov    dh, [Current_Drv]
28590 0000986C 8835[EE560100]    <1>       mov    [RUN_CDRV], dh
28591                            <1>
28592 00009872 8A15[32580100]    <1>       mov    dl, [FindFile_Drv]
28593 00009878 38F2              <1>       cmp    dl, dh
28594 0000987A 7412              <1>       je     short loc_run_change_directory
28595                            <1>
28596 0000987C 8005[495B0100]02  <1>       add    byte [Run_Manual_Path], 2
28597                            <1>
28598 00009883 E8D5D3FFFF        <1>       call   change_current_drive
28599 00009888 0F828AE2FFFF      <1>       jc     loc_run_cmd_failed
28600                            <1>
28601                            <1> loc_run_change_directory:
28602 0000988E 803D[33580100]20  <1>       cmp    byte [FindFile_Directory], 20h
28603 00009895 7623              <1>       jna    short loc_run_find_executable_file
28604                            <1>
28605 00009897 FE05[495B0100]    <1>       inc    byte [Run_Manual_Path]
28606                            <1>
28607 0000989D FE05[98020100]    <1>       inc    byte [Restore_CDIR]
28608                            <1>
28609 000098A3 BE[33580100]      <1>       mov    esi, FindFile_Directory
28610 000098A8 30E4              <1>       xor    ah, ah ; CD_COMMAND sign -> 0
28611 000098AA E843030000        <1>       call   change_current_directory
28612 000098AF 0F8263E2FFFF      <1>       jc     loc_run_cmd_failed
28613                            <1>
28614                            <1> loc_run_change_prompt_dir_string:
28615 000098B5 E858020000        <1>       call   change_prompt_dir_string
28616                            <1>
28617                            <1> loc_run_find_executable_file:
28618 000098BA 66C705[485B0100]00- <1>     mov    word [Run_Auto_Path], 0
28619 000098C2 00                <1>
28620                            <1>
28621                            <1> loc_run_find_executable_file_next:
28622 000098C3 BE[74580100]      <1>       mov    esi, FindFile_Name
28623                            <1> loc_run_find_program_file_next:
28624 000098C8 66B80018          <1>       mov    ax, 1800h ; Except volume label and dirs
28625 000098CC E81EE7FFFF        <1>       call   find_first_file
28626                            <1>       ; ESI = Directory Entry (FindFile_DirEntry) Location
28627                            <1>       ; EDI = Directory Buffer Directory Entry Location
28628                            <1>       ; EAX = File size
28629 000098D1 0F835C010000      <1>       jnc    loc_load_and_run_file
28630                            <1>
28631 000098D7 3C02              <1>       cmp    al, 2 ; file not found
28632 000098D9 0F8539E2FFFF      <1>       jne    loc_run_cmd_failed
28633                            <1>
28634 000098DF 66A1[4A5B0100]    <1>       mov    ax, word [EXE_ID]
28635 000098E5 80FC2E            <1>       cmp    ah, '.' ; File name has extension sign
28636 000098E8 7424              <1>       je     short loc_run_check_auto_path
28637                            <1>
28638 000098EA 08C0              <1>       or     al, al
28639 000098EC 7520              <1>       jnz    short loc_run_check_auto_path
28640                            <1>
28641 000098EE 80FC08            <1>       cmp    ah, 8 ; count of file name chars
28642 000098F1 771B              <1>       ja     short loc_run_check_auto_path
28643                            <1>
28644                            <1> loc_run_change_file_ext_to_prg:
28645 000098F3 0FB6DC            <1>       movzx  ebx, ah ; count of file name chars
28646 000098F6 BE[74580100]      <1>       mov    esi, FindFile_Name
28647 000098FB 01F3              <1>       add    ebx, esi
28648                            <1>       ; 07/05/2016
28649 000098FD C7032E505247      <1>       mov    dword [ebx],  '.PRG'
28650 00009903 66C705[4A5B0100]50- <1>     mov    word [EXE_ID], 'P.'
28651 0000990B 2E                <1>
28652 0000990C EBBA              <1>       jmp    short loc_run_find_program_file_next
28653                            <1>
28654                            <1> loc_run_check_auto_path:
28655                            <1>       ; NOTE: /// 07/05/2016 ///
28656                            <1>       ; If the path is given, value of byte [Run_Manual_Path]
```

```
28657                                  <1>       ; will not be ZERO. If so, file searching by using
28658                                  <1>       ; Automatic Path (via 'PATH' environment variable)
28659                                  <1>       ; will not be applicable, because the program file
28660                                  <1>       ; is already/absolutely not found.
28661                                  <1>
28662 0000990E A0[495B0100]           <1>       mov    al, [Run_Manual_Path]
28663 00009913 08C0                   <1>       or     al, al
28664 00009915 0F85D2E1FFFF           <1>       jnz    loc_cmd_failed
28665                                  <1>
28666                                  <1> loc_run_check_auto_path_again:
28667 0000991B 66833D[485B0100]FF     <1>       cmp    word [Run_Auto_Path], 0FFFFh
28668                                  <1>              ; 0FFFFh = Not a valid run path (in ENV block)
28669 00009923 0F83C4E1FFFF           <1>       jnb    loc_cmd_failed
28670                                  <1>       ; xor  al, al
28671 00009929 BE[64030100]           <1>       mov    esi, Cmd_Path ; 'PATH'
28672 0000992E BF[8E4F0100]           <1>       mov    edi, TextBuffer
28673 00009933 E857F9FFFF             <1>       call   get_environment_string
28674 00009938 730E                   <1>       jnc    short loc_run_chk_filename_ext_again
28675 0000993A 66C705[485B0100]FF-    <1>       mov    word [Run_Auto_Path], 0FFFFh ; invalid
28676 00009942 FF                     <1>
28677 00009943 E9A5E1FFFF             <1>       jmp    loc_cmd_failed
28678                                  <1>
28679                                  <1> loc_run_chk_filename_ext_again:
28680 00009948 89C1                   <1>       mov    ecx, eax ; string length (with zero tail)
28681 0000994A 49                     <1>       dec    ecx ; without zero tail
28682 0000994B 66A1[4A5B0100]         <1>       mov    ax, [EXE_ID]
28683 00009951 80FC2E                 <1>       cmp    ah, '.'
28684 00009954 740E                   <1>       je     short loc_run_chk_auto_path_pos
28685                                  <1>
28686                                  <1> loc_run_change_file_ext_to_noext_again:
28687 00009956 0FB6DC                 <1>       movzx  ebx, ah
28688 00009959 BE[74580100]           <1>       mov    esi, FindFile_Name
28689 0000995E 01F3                   <1>       add    ebx, esi
28690 00009960 29C0                   <1>       sub    eax, eax
28691 00009962 8903                   <1>       mov    [ebx], eax ; 0 ; erase extension (.PRG)
28692                                  <1>
28693                                  <1> loc_run_chk_auto_path_pos:
28694                                  <1>       ;movzx eax,  word [Run_Auto_Path]
28695 00009964 66A1[485B0100]         <1>       mov    ax, [Run_Auto_Path]
28696 0000996A 39C8                   <1>       cmp    eax, ecx ; ecx = string length (except zero tail)
28697 0000996C 0F837BE1FFFF           <1>       jnb    loc_cmd_failed
28698                                  <1>       ;or    eax, eax
28699 00009972 6609C0                 <1>       or     ax, ax
28700 00009975 7502                   <1>       jnz    short loc_run_auto_path_pos_move
28701 00009977 B005                   <1>       mov    al, 5
28702                                  <1>
28703                                  <1> loc_run_auto_path_pos_move:
28704 00009979 89FE                   <1>       mov    esi, edi ; offset TextBuffer
28705 0000997B 01C6                   <1>       add    esi, eax
28706                                  <1>
28707                                  <1> loc_run_auto_path_pos_space_loop:
28708 0000997D AC                     <1>       lodsb
28709 0000997E 3C20                   <1>       cmp    al, 20h
28710 00009980 74FB                   <1>       je     short loc_run_auto_path_pos_space_loop
28711 00009982 0F8265E1FFFF           <1>       jb     loc_cmd_failed
28712 00009988 AA                     <1>       stosb
28713                                  <1> loc_run_auto_path_pos_move_next:
28714 00009989 AC                     <1>       lodsb
28715 0000998A 3C3B                   <1>       cmp    al, ';'
28716 0000998C 7414                   <1>       je     short loc_run_auto_path_pos_move_last_byte
28717 0000998E 3C20                   <1>       cmp    al, 20h
28718 00009990 74F7                   <1>       je     short loc_run_auto_path_pos_move_next
28719 00009992 7203                   <1>       jb     short loc_byte_ptr_end_of_path
28720 00009994 AA                     <1>       stosb
28721 00009995 EBF2                   <1>       jmp    short loc_run_auto_path_pos_move_next
28722                                  <1>
28723                                  <1> loc_byte_ptr_end_of_path:
28724 00009997 66C705[485B0100]FF-    <1>       mov    word [Run_Auto_Path], 0FFFFh ; end of path
28725 0000999F FF                     <1>
28726 000099A0 EB0D                   <1>       jmp    short loc_run_auto_path_move_ok
28727                                  <1>
28728                                  <1> loc_run_auto_path_pos_move_last_byte:
28729 000099A2 89F0                   <1>       mov    eax, esi
28730 000099A4 2D[8E4F0100]           <1>       sub    eax, TextBuffer
28731 000099A9 66A3[485B0100]         <1>       mov    [Run_Auto_Path], ax ; next path position
28732                                  <1>
28733                                  <1> loc_run_auto_path_move_ok:
28734 000099AF 4F                     <1>       dec    edi
28735 000099B0 B02F                   <1>       mov    al, '/'
28736 000099B2 3807                   <1>       cmp    [edi], al
28737 000099B4 7403                   <1>       je     short loc_run_auto_path_move_file_name
28738 000099B6 47                     <1>       inc    edi
28739 000099B7 8807                   <1>       mov    [edi], al
28740                                  <1>
28741                                  <1> loc_run_auto_path_move_file_name:
28742 000099B9 47                     <1>       inc    edi
28743 000099BA BE[74580100]           <1>       mov    esi, FindFile_Name
28744                                  <1>
28745                                  <1> loc_run_auto_path_move_fn_loop:
28746 000099BF AC                     <1>       lodsb
28747 000099C0 AA                     <1>       stosb
28748 000099C1 08C0                   <1>       or     al, al
28749 000099C3 75FA                   <1>       jnz    short loc_run_auto_path_move_fn_loop
28750                                  <1>
28751 000099C5 BE[8E4F0100]           <1>       mov    esi, TextBuffer
28752 000099CA BF[32580100]           <1>       mov    edi, FindFile_Drv
28753 000099CF E832080000             <1>       call   parse_path_name
28754 000099D4 0F8213E1FFFF           <1>       jc     loc_cmd_failed
28755                                  <1>
28756 000099DA 8A35[8E4E0100]         <1>       mov    dh, [Current_Drv]
28757 000099E0 8A15[32580100]         <1>       mov    dl, [FindFile_Drv]
28758 000099E6 38F2                   <1>       cmp    dl, dh
```

```
28759 000099E8 740B                <1>       je     short loc_run_change_directory_again
28760                              <1>
28761 000099EA E86ED2FFFF          <1>       call   change_current_drive
28762 000099EF 0F8223E1FFFF        <1>       jc     loc_run_cmd_failed
28763                              <1>
28764                              <1> loc_run_change_directory_again:
28765 000099F5 803D[33580100]20    <1>       cmp    byte [FindFile_Directory], 20h
28766 000099FC 761D                <1>       jna    short loc_load_executable_cdir_chk_again
28767                              <1>
28768 000099FE FE05[98020100]      <1>       inc    byte [Restore_CDIR]
28769 00009A04 BE[33580100]        <1>       mov    esi, FindFile_Directory
28770 00009A09 30E4                <1>       xor    ah, ah ; CD_COMMAND sign -> 0
28771 00009A0B E8E2010000          <1>       call   change_current_directory
28772 00009A10 0F8202E1FFFF        <1>       jc     loc_run_cmd_failed
28773                              <1>
28774                              <1> loc_run_chg_prompt_dir_str_again:
28775 00009A16 E8F7000000          <1>       call   change_prompt_dir_string
28776                              <1>
28777                              <1> loc_load_executable_cdir_chk_again:
28778 00009A1B A1[884E0100]        <1>       mov    eax, [Current_Dir_FCluster]
28779 00009A20 3B05[445B0100]      <1>       cmp    eax, [Run_CDirFC]
28780 00009A26 0F8597FEFFFF        <1>       jne    loc_run_find_executable_file_next
28781 00009A2C 30C0                <1>       xor    al, al ; 0
28782 00009A2E E9E8FEFFFF          <1>       jmp    loc_run_check_auto_path_again
28783                              <1>
28784                              <1> loc_load_and_run_file:
28785                              <1>       ; 04/01/2017
28786                              <1>       ; 23/04/2016
28787 00009A33 BE[74580100]        <1>       mov    esi, FindFile_Name
28788 00009A38 BF[8E4F0100]        <1>       mov    edi, TextBuffer
28789                              <1>
28790                              <1>       ; 24/04/2016
28791 00009A3D 31D2                <1>       xor    edx, edx
28792 00009A3F 668915[4A040600]    <1>       mov    word [argc], dx ; 0
28793 00009A46 8915[8C030600]      <1>       mov    dword [u.nread], edx ; 0
28794                              <1>
28795                              <1> loc_load_and_run_file_1:
28796 00009A4C AC                  <1>       lodsb
28797 00009A4D AA                  <1>       stosb
28798 00009A4E FF05[8C030600]      <1>       inc    dword [u.nread]
28799 00009A54 20C0                <1>       and    al, al
28800 00009A56 75F4                <1>       jnz    short loc_load_and_run_file_1
28801                              <1>
28802 00009A58 A0[EF4E0100]        <1>       mov    al, [CmdArgStart]
28803 00009A5D 20C0                <1>       and    al, al
28804 00009A5F 7445                <1>       jz     short loc_load_and_run_file_7
28805                              <1>
28806 00009A61 0FB6F0              <1>       movzx  esi, al ; 11/05/2016
28807 00009A64 B950000000          <1>       mov    ecx, 80
28808 00009A69 29F1                <1>       sub    ecx, esi
28809 00009A6B 81C6[3E4F0100]      <1>       add    esi, CommandBuffer
28810                              <1>
28811 00009A71 66FF05[4A040600]    <1>       inc    word [argc] ; 11/05/2016
28812                              <1>
28813                              <1> loc_load_and_run_file_2:
28814 00009A78 AC                  <1>       lodsb
28815 00009A79 3C20                <1>       cmp    al, 20h
28816 00009A7B 7717                <1>       ja     short loc_load_and_run_file_5
28817 00009A7D 721E                <1>       jb     short loc_load_and_run_file_6
28818                              <1>
28819                              <1> loc_load_and_run_file_3:
28820 00009A7F 803E20              <1>       cmp    byte [esi], 20h
28821 00009A82 7707                <1>       ja     short loc_load_and_run_file_4
28822 00009A84 7217                <1>       jb     short loc_load_and_run_file_6
28823 00009A86 46                  <1>       inc    esi
28824 00009A87 E2F6                <1>       loop   loc_load_and_run_file_3
28825 00009A89 EB12                <1>       jmp    short loc_load_and_run_file_6
28826                              <1>
28827                              <1> loc_load_and_run_file_4:
28828 00009A8B 28C0                <1>       sub    al, al ; 0
28829 00009A8D 66FF05[4A040600]    <1>       inc    word [argc]
28830                              <1> loc_load_and_run_file_5:
28831 00009A94 AA                  <1>       stosb
28832 00009A95 FF05[8C030600]      <1>       inc    dword [u.nread]
28833 00009A9B E2DB                <1>       loop   loc_load_and_run_file_2
28834                              <1>
28835                              <1> loc_load_and_run_file_6:
28836 00009A9D 30C0                <1>       xor    al, al ; 0
28837 00009A9F AA                  <1>       stosb
28838 00009AA0 FF05[8C030600]      <1>       inc    dword [u.nread]
28839                              <1> loc_load_and_run_file_7:
28840 00009AA6 8807                <1>       mov    [edi], al ; 0
28841 00009AA8 66FF05[4A040600]    <1>       inc    word [argc] ; 24/04/2016
28842 00009AAF FF05[8C030600]      <1>       inc    dword [u.nread] ; 24/04/2016
28843 00009AB5 BE[8E4F0100]        <1>       mov    esi, TextBuffer
28844 00009ABA 8B15[A0580100]      <1>       mov    edx, [FindFile_DirEntry+DirEntry_FileSize]
28845 00009AC0 66A1[98580100]      <1>       mov    ax, [FindFile_DirEntry+DirEntry_FstClusHI]
28846 00009AC6 66C1E010            <1>       shl    ax, 16
28847 00009ACA 66A1[9E580100]      <1>       mov    ax, [FindFile_DirEntry+DirEntry_FstClusLO]
28848                              <1>       ; EAX = First Cluster number
28849                              <1>       ; EDX = File Size
28850                              <1>       ; ESI = Argument list address
28851                              <1>       ; [argc] = argument count
28852                              <1>       ; [u.nread] = argument list length
28853 00009AD0 E8AA450000          <1>       call   load_and_run_file ; trdosk6.s
28854                              <1>         ;jc  loc_run_cmd_failed ; 04/01/2017
28855                              <1> loc_load_and_run_file_8: ; 06/05/2016
28856 00009AD5 E945E9FFFF          <1>       jmp    loc_file_rw_restore_retn
28857                              <1>
28858                              <1> check_prg_filename_ext:
28859                              <1>       ; 23/04/2016 (TRDOS 386 = TRDOS v2.0)
28860                              <1>       ; 10/09/2011
```

```
28861                           <1>       ; (TRDOS v1, CMDINTR.ASM, 'proc_check_exe_filename_ext')
28862                           <1>       ; 14/11/2009
28863                           <1>       ; INPUT ->
28864                           <1>       ;     ESI = Dot File Name
28865                           <1>       ; OUTPUT ->
28866                           <1>       ;     cf = 0 -> EXE_ID in AL
28867                           <1>       ;     ESI = Last char + 1 position
28868                           <1>       ;     cf = 1 -> Invalid executable file name
28869                           <1>       ;     or no file name extension if AH<=8
28870                           <1>       ;     AL = Last file name char
28871                           <1>       ;     cf = 0 -> AL='P' (PRG), AL=0 (no extension)
28872                           <1>       ;
28873                           <1>       ; (Modified registers: EAX, ESI)
28874                           <1>
28875 00009ADA 30E4              <1>       xor    ah, ah
28876                           <1> loc_run_check_filename_ext:
28877 00009ADC AC                <1>       lodsb
28878 00009ADD 3C21              <1>       cmp    al, 21h
28879 00009ADF 7229              <1>       jb     short loc_check_exe_fn_retn
28880 00009AE1 FEC4              <1>       inc    ah
28881 00009AE3 3C2E              <1>       cmp    al, '.'
28882 00009AE5 75F5              <1>       jne    short loc_run_check_filename_ext
28883                           <1>
28884                           <1> loc_run_check_filename_ext_dot:
28885 00009AE7 80FC02            <1>       cmp    ah, 2 ; .??? is not valid
28886 00009AEA 88C4              <1>       mov    ah, al ; '.'
28887 00009AEC 7219              <1>       jb     short loc_check_prg_fn_retn
28888                           <1>
28889                           <1> loc_run_check_filename_ext_dot_ok:
28890 00009AEE AC                <1>       lodsb
28891 00009AEF 24DF              <1>       and    al, 0DFh
28892                           <1>
28893                           <1> loc_run_check_filename_ext_prg:
28894 00009AF1 3C50              <1>       cmp    al, 'P'
28895 00009AF3 7212              <1>       jb     short loc_check_prg_fn_retn
28896 00009AF5 7711              <1>       ja     short loc_check_prg_fn_stc
28897 00009AF7 AC                <1>       lodsb
28898 00009AF8 24DF              <1>       and    al, 0DFh
28899 00009AFA 3C52              <1>       cmp    al, 'R'
28900 00009AFC 750A              <1>       jne    short loc_check_prg_fn_stc
28901 00009AFE AC                <1>       lodsb
28902 00009AFF 24DF              <1>       and    al, 0DFh
28903 00009B01 3C47              <1>       cmp    al, 'G'
28904 00009B03 7503              <1>       jne    short loc_check_prg_fn_stc
28905                           <1>
28906 00009B05 B050              <1>       mov    al, 'P'
28907                           <1> loc_check_prg_fn_retn:
28908 00009B07 C3                <1>       retn
28909                           <1>
28910                           <1> loc_check_prg_fn_stc:
28911 00009B08 F9                <1>       stc
28912 00009B09 C3                <1>       retn
28913                           <1>
28914                           <1> loc_check_exe_fn_retn:
28915 00009B0A 28C0              <1>       sub    al, al ; 0
28916 00009B0C C3                <1>       retn
28917                           <1>
28918                           <1> find_and_list_files:
28919 00009B0D C3                <1>       retn
28920                           <1> set_exec_arguments:
28921 00009B0E C3                <1>       retn
28922                           <1> delete_fs_directory:
28923 00009B0F 31C0              <1>       xor eax, eax
28924 00009B11 C3                <1>       retn
28925                               %include 'trdosk4.s' ; 24/01/2016
28926                           <1> ; ****************************************************************************
28927                           <1> ; TRDOS386.ASM (TRDOS 386 Kernel - v2.0.0) - Directory Functions : trdosk4.s
28928                           <1> ; ----------------------------------------------------------------------------
28929                           <1> ; Last Update: 17/10/2016
28930                           <1> ; ----------------------------------------------------------------------------
28931                           <1> ; Beginning: 24/01/2016
28932                           <1> ; ----------------------------------------------------------------------------
28933                           <1> ; Assembler: NASM version 2.11 (trdos386.s)
28934                           <1> ; ----------------------------------------------------------------------------
28935                           <1> ; Derived from TRDOS Operating System v1.0 (8086) source code by Erdogan Tan
28936                           <1> ; DIR.ASM (09/10/2011)
28937                           <1> ; ****************************************************************************
28938                           <1>
28939                           <1> ; DIR.ASM  [ TRDOS KERNEL - COMMAND EXECUTER SECTION - DIRECTORY FUNCTIONS ]
28940                           <1> ; (c) 2004-2010  Erdogan TAN  [ 17/01/2004 ]  Last Update: 09/10/2011
28941                           <1> ; FILE.ASM [ FILE FUNCTIONS ] Last Update: 09/10/2011
28942                           <1>
28943                           <1> change_prompt_dir_string:
28944                           <1>       ; 05/10/2016
28945                           <1>       ; 24/01/2016 (TRDOS 386 = TRDOS v2.0)
28946                           <1>       ; 27/03/2011
28947                           <1>       ; 09/10/2009
28948                           <1>       ; INPUT/OUTPUT => none
28949                           <1>       ; this procedure changes current directory string/text
28950                           <1>       ; 2005
28951                           <1>
28952 00009B12 BE[EF560100]      <1>       mov    esi, PATH_Array
28953                           <1> change_prompt_dir_str: ; 05/10/2016 (call from 'set_working_path')
28954 00009B17 BF[924E0100]      <1>       mov    edi, Current_Directory
28955 00009B1C 8A25[8C4E0100]    <1>       mov    ah, [Current_Dir_Level]
28956 00009B22 E807000000        <1>       call   set_current_directory_string
28957 00009B27 880D[ED4E0100]    <1>       mov    [Current_Dir_StrLen], cl
28958                           <1>
28959 00009B2D C3                <1>       retn
28960                           <1>
28961                           <1> set_current_directory_string:
28962                           <1>       ; 24/01/2016 (TRDOS 386 = TRDOS v2.0)
```

```
28963                                  <1>       ; 27/03/2011
28964                                  <1>       ; 09/10/2009
28965                                  <1>       ; INPUT:
28966                                  <1>       ;    ESI = Path Array Address
28967                                  <1>       ;    EDI = Current Directory String Buffer
28968                                  <1>       ;    AH = Current Directory Level
28969                                  <1>       ; OUTPUT => EAX, EBX, ESI will be changed
28970                                  <1>       ;    EDI will be same with input
28971                                  <1>       ;    ECX = Current Directory String Length
28972                                  <1>
28973 00009B2E 57                      <1>       push    edi
28974 00009B2F 80FC00                  <1>       cmp     ah, 0
28975 00009B32 7652                    <1>       jna    short pass_write_path
28976 00009B34 83C610                  <1>       add    esi, 16
28977 00009B37 89F3                    <1>       mov    ebx, esi
28978                                  <1> loc_write_path:
28979 00009B39 B908000000              <1>       mov    ecx, 8
28980                                  <1> path_write_dirname1:
28981 00009B3E AC                      <1>       lodsb
28982 00009B3F 3C20                    <1>       cmp    al, 20h
28983 00009B41 7612                    <1>       jna    short pass_write_dirname1
28984 00009B43 AA                      <1>       stosb
28985 00009B44 81FF[EC4E0100]          <1>       cmp    edi, End_Of_Current_Dir_Str
28986 00009B4A 733A                    <1>       jnb    short pass_write_path
28987 00009B4C E2F0                    <1>       loop   path_write_dirname1
28988 00009B4E 803E20                  <1>       cmp    byte [esi], 20h
28989 00009B51 7624                    <1>       jna    short pass_write_dirname2
28990 00009B53 EB0A                    <1>       jmp     short loc_put_dot_cont_ext
28991                                  <1> pass_write_dirname1:
28992 00009B55 89DE                    <1>       mov    esi, ebx
28993 00009B57 83C608                  <1>       add    esi, 8
28994 00009B5A 803E20                  <1>       cmp    byte [esi], 20h
28995 00009B5D 7618                    <1>       jna    short pass_write_dirname2
28996                                  <1> loc_put_dot_cont_ext:
28997 00009B5F C6072E                  <1>       mov    byte [edi], "."
28998                                  <1>       ;mov   ecx, 3
28999 00009B62 B103                    <1>       mov    cl, 3
29000                                  <1> loc_check_dir_name_ext:
29001 00009B64 AC                      <1>       lodsb
29002 00009B65 47                      <1>       inc    edi
29003 00009B66 3C20                    <1>       cmp    al, 20h
29004 00009B68 760D                    <1>       jna    short pass_write_dirname2
29005 00009B6A 8807                    <1>       mov    [edi], al
29006 00009B6C 81FF[EC4E0100]          <1>       cmp    edi, End_Of_Current_Dir_Str
29007 00009B72 7312                    <1>       jnb    short pass_write_path
29008 00009B74 E2EE                    <1>       loop    loc_check_dir_name_ext
29009 00009B76 47                      <1>       inc    edi
29010                                  <1> pass_write_dirname2:
29011 00009B77 FECC                    <1>       dec    ah
29012 00009B79 740B                    <1>       jz      short pass_write_path
29013 00009B7B 83C310                  <1>       add    ebx, 16
29014 00009B7E 89DE                    <1>       mov    esi, ebx
29015 00009B80 C6072F                  <1>       mov    byte [edi],"/"
29016 00009B83 47                      <1>       inc    edi
29017 00009B84 EBB3                    <1>       jmp    short loc_write_path
29018                                  <1> pass_write_path:
29019 00009B86 C60700                  <1>       mov    byte [edi], 0
29020 00009B89 47                      <1>       inc    edi
29021 00009B8A 89F9                    <1>       mov    ecx, edi
29022 00009B8C 5F                      <1>       pop    edi
29023 00009B8D 29F9                    <1>       sub    ecx, edi
29024                                  <1>       ; ECX = Current Directory String Length
29025 00009B8F C3                      <1>       retn
29026                                  <1>
29027                                  <1> get_current_directory:
29028                                  <1>       ; 15/10/2016
29029                                  <1>       ; 14/02/2016
29030                                  <1>       ; 24/01/2016 (TRDOS 386 = TRDOS v2.0)
29031                                  <1>       ; 27/03/2011
29032                                  <1>       ;
29033                                  <1>       ; INPUT-> ESI = Current Directory Buffer
29034                                  <1>       ;         DL = TRDOS Logical Dos Drive Number + 1
29035                                  <1>       ;              (0= Default/Current Drive)
29036                                  <1>       ;
29037                                  <1>       ;   Note: Required dir buffer length may be <= 92 bytes
29038                                  <1>       ;         for TRDOS (7*12 name chars + 7 slash + 0)
29039                                  <1>       ; OUTPUT ->  ESI = Current Directory Buffer
29040                                  <1>       ;            EAX, EBX, ECX, EDX, EDI will be changed
29041                                  <1>       ;            CX/CL = Current Directory String Length
29042                                  <1>       ;          DL = Drive Number (0 based)
29043                                  <1>       ;            (If input is 0, output is current drv number)
29044                                  <1>       ;            DH = same with input
29045                                  <1>       ;   cf = 0 -> AL = 0
29046                                  <1>       ;   cf = 1 -> error code in AL
29047                                  <1>
29048                                  <1> loc_get_current_drive_0:
29049 00009B90 80FA00                  <1>       cmp    dl, 0
29050 00009B93 7708                    <1>       ja     short loc_get_current_drive_1
29051 00009B95 8A15[8E4E0100]          <1>       mov    dl, [Current_Drv]
29052 00009B9B EB17                    <1>       jmp    short loc_get_current_drive_2
29053                                  <1> loc_get_current_drive_1:
29054 00009B9D FECA                    <1>       dec    dl
29055 00009B9F 3A15[97020100]          <1>       cmp    dl, [Last_DOS_DiskNo]
29056 00009BA5 760D                    <1>       jna    short loc_get_current_drive_2
29057 00009BA7 B80F000000              <1>       mov    eax, 0Fh ; Invalid drive (Drive not ready!)
29058 00009BAC F5                      <1>       cmc    ; stc
29059 00009BAD C3                      <1>       retn
29060                                  <1>
29061                                  <1> loc_get_current_drive_not_ready_retn:
29062 00009BAE 5E                      <1>       pop    esi
29063                                  <1>       ;mov   eax, 15
29064 00009BAF 66B80F00                <1>       mov    ax, 15 ; Drive not ready
```

```
29065 00009BB3 C3                <1>       retn
29066                            <1>
29067                            <1> loc_get_current_drive_2:
29068 00009BB4 31C0              <1>       xor    eax, eax
29069 00009BB6 88D4              <1>       mov    ah, dl
29070 00009BB8 56                <1>       push   esi
29071 00009BB9 BE00010900        <1>       mov    esi, Logical_DOSDisks
29072 00009BBE 01C6              <1>       add    esi, eax
29073 00009BC0 8A06              <1>       mov    al, [esi+LD_Name]
29074 00009BC2 3C41              <1>       cmp    al, 'A'
29075 00009BC4 72E8              <1>       jb     short loc_get_current_drive_not_ready_retn
29076                            <1>
29077 00009BC6 8A667F            <1>       mov    ah, [esi+LD_CDirLevel]
29078 00009BC9 08E4              <1>       or     ah, ah
29079 00009BCB 7506              <1>       jnz    short loc_get_current_drive_3
29080                            <1>
29081                            <1>       ;xor   ah, ah ; mov ah, 0
29082 00009BCD 8826              <1>       mov    [esi], ah
29083 00009BCF 31C9              <1>       xor    ecx, ecx
29084 00009BD1 EB1C              <1>       jmp    short loc_get_current_drive_4
29085                            <1>
29086                            <1> loc_get_current_drive_3:
29087 00009BD3 BF[EF560100]      <1>         mov     edi, PATH_Array
29088 00009BD8 57                <1>       push   edi
29089 00009BD9 81C680000000      <1>       add    esi, LD_CurrentDirectory
29090 00009BDF B920000000        <1>       mov    ecx, 32
29091 00009BE4 F3A5              <1>       rep    movsd
29092 00009BE6 5E                <1>       pop    esi ; Path Array Address
29093 00009BE7 5F                <1>       pop    edi ; pushed esi (current dir buffer offset)
29094                            <1>       ;
29095 00009BE8 E841FFFFFF        <1>       call   set_current_directory_string
29096 00009BED 89FE              <1>       mov    esi, edi
29097                            <1>
29098                            <1> loc_get_current_drive_4:
29099 00009BEF 30C0              <1>       xor    al, al
29100 00009BF1 C3                <1>       retn
29101                            <1>
29102                            <1> change_current_directory:
29103                            <1>       ; 19/02/2016
29104                            <1>       ; 11/02/2016
29105                            <1>       ; 10/02/2016
29106                            <1>       ; 08/02/2016
29107                            <1>       ; 06/02/2016 (TRDOS 386 = TRDOS v2.0)
29108                            <1>       ; 18/09/2011 (DIR.ASM, 09/10/2011)
29109                            <1>       ; 04/10/2009
29110                            <1>       ; 2005
29111                            <1>       ; INPUT ->
29112                            <1>       ;      ESI = Directory string
29113                            <1>       ;      ah = CD command (CDh = save current dir string)
29114                            <1>       ; OUTPUT ->
29115                            <1>       ;      EDI = DOS Drive Description Table
29116                            <1>       ;      cf = 1 -> error
29117                            <1>       ;         EAX = Error code
29118                            <1>       ;      cf = 0 -> succesful
29119                            <1>       ;         ESI = PATH_Array
29120                            <1>       ;         EAX = Current Directory First Cluster
29121                            <1>       ;
29122                            <1>       ; (EAX, EBX, ECX, EDX, ESI, EDI will be changed)
29123                            <1>
29124 00009BF2 8825[7D570100]    <1>       mov    [CD_COMMAND], ah
29125 00009BF8 803E2F            <1>       cmp    byte [esi], '/'
29126 00009BFB 7505              <1>       jne    short loc_ccd_cdir_level
29127 00009BFD 46                <1>       inc    esi
29128 00009BFE 30C0              <1>       xor    al, al
29129 00009C00 EB05              <1>       jmp    short loc_ccd_parse_path_name
29130                            <1> loc_ccd_cdir_level:
29131 00009C02 A0[8C4E0100]      <1>       mov    al, [Current_Dir_Level]
29132                            <1> loc_ccd_parse_path_name:
29133 00009C07 88C4              <1>       mov    ah, al
29134 00009C09 BF[EF560100]      <1>       mov    edi, PATH_Array
29135                            <1>
29136                            <1> ; Reset directory levels > cdir level
29137                            <1>       ; is this required !?
29138                            <1>       ;
29139                            <1>       ; Relations:
29140                            <1>       ; MAINPROG.ASM (pass_ccdrv_reset_cdir_FAT_fcluster)
29141                            <1>       ; proc_parse_dir_name,
29142                            <1>       ; proc_change_current_directory (this procedure)
29143                            <1>       ; proc_change_prompt_dir_string
29144                            <1>
29145 00009C0E 0FB6C8            <1>       movzx  ecx, al
29146 00009C11 FEC1              <1>       inc    cl
29147 00009C13 C0E104            <1>       shl    cl, 4
29148 00009C16 01CF              <1>       add    edi, ecx
29149 00009C18 B107              <1>       mov    cl, 7
29150 00009C1A 28C1              <1>       sub    cl, al
29151 00009C1C C0E102            <1>       shl    cl, 2
29152 00009C1F 89C3              <1>       mov    ebx, eax
29153 00009C21 31C0              <1>       xor    eax, eax ; 0
29154 00009C23 F3AB              <1>       rep    stosd
29155 00009C25 89D8              <1>       mov    eax, ebx
29156                            <1>
29157 00009C27 BF[EF560100]      <1>       mov    edi, PATH_Array
29158                            <1>
29159 00009C2C 803E20            <1>       cmp    byte [esi], 20h
29160 00009C2F F5                <1>       cmc
29161 00009C30 7305              <1>       jnc    short pass_ccd_parse_dir_name
29162                            <1>
29163                            <1>             ; ESI = Path name
29164                            <1>             ; AL = CCD_Level
29165 00009C32 E872010000        <1>         call    parse_dir_name
29166                            <1>             ; AL = CCD_Level
```

```
                                <1>             ; AH = Last_Dir_Level
                                <1>             ; (EDI = PATH_Array)
                                <1>
                                <1> pass_ccd_parse_dir_name:
00009C37 9C                     <1>       pushf
                                <1>
                                <1>       ;mov   [CCD_Level], al
                                <1>         ;mov [Last_Dir_Level], ah
00009C38 66A3[73570100]         <1>       mov    [CCD_Level], ax
                                <1>
00009C3E 31DB                   <1>       xor    ebx, ebx
00009C40 8A3D[8E4E0100]         <1>       mov    bh, [Current_Drv]
00009C46 BE00010900             <1>       mov    esi, Logical_DOSDisks
00009C4B 01DE                   <1>       add    esi, ebx
                                <1>
00009C4D 9D                     <1>       popf
00009C4E 720A                   <1>       jc     short loc_ccd_bad_path_name_retn
                                <1>
00009C50 8935[6F570100]         <1>       mov    [CCD_DriveDT], esi
                                <1>
00009C56 3C07                   <1>       cmp    al, 7
00009C58 7209                   <1>       jb     short loc_ccd_load_child_dir
                                <1>
                                <1> loc_ccd_bad_path_name_retn:
00009C5A 87F7                   <1>       xchg   esi, edi
00009C5C B813000000             <1>       mov    eax, 19 ; Bad directory/path name
00009C61 F9                     <1>       stc
                                <1> loc_ccd_retn_p:
00009C62 C3                     <1>       retn
                                <1>
                                <1> loc_ccd_load_child_dir:
                                <1>       ; AL = CCD_Level
00009C63 08C0                   <1>       or     al, al
00009C65 7468                   <1>       jz     short loc_ccd_load_root_dir
                                <1>
00009C67 6689C1                 <1>       mov    cx, ax
00009C6A C0E004                 <1>       shl    al, 4
00009C6D 0FB6F0                 <1>       movzx  esi, al
00009C70 01FE                   <1>       add    esi, edi  ; offset PATH_Array
                                <1>
00009C72 8B460C                 <1>       mov    eax, [esi+12]
00009C75 38E9                   <1>       cmp    cl, ch
00009C77 0F84FA000000           <1>         je       loc_ccd_load_sub_directory
00009C7D A3[884E0100]           <1>       mov    [Current_Dir_FCluster], eax
                                <1>
                                <1> loc_ccd_load_child_dir_next:
00009C82 83C610                 <1>       add    esi, 16 ; DOS DirEntry Format FileName Address
                                <1>
                                <1>       ; Directory attribute : 10h
00009C85 B010                   <1>       mov    al, 00010000b ; 10h (Attrib AND mask)
                                <1>       ;mov   ah, 11001000b ; C8h
                                <1>       ; Volume name attribute: 8h
00009C87 B408                   <1>       mov    ah, 00001000b ; 08h (Attrib NAND, AND --> zero mask)
                                <1>
00009C89 6631C9                 <1>       xor    cx, cx
00009C8C E8B5010000             <1>       call   locate_current_dir_file
00009C91 7353                   <1>       jnc    short loc_ccd_set_dir_cluster_ptr
                                <1>
                                <1>        ; 19/02/2016
                                <1>       ;mov   edi, [CCD_DriveDT]
00009C93 8A25[73570100]         <1>       mov    ah, [CCD_Level]
00009C99 803D[7D570100]CD       <1>       cmp    byte [CD_COMMAND], 0CDh ;'CD' command or another
00009CA0 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.)
00009CA2 88E1                   <1>       mov    cl, ah
00009CA4 50                     <1>       push   eax
00009CA5 E8E3000000             <1>       call   loc_ccd_save_current_dir
00009CAA 58                     <1>       pop    eax
                                <1> loc_ccd_load_child_dir_err:
00009CAB 3C03                   <1>       cmp    al, 3  ; AL = 2 => File not found error
00009CAD 7202                   <1>       jb     short loc_ccd_path_not_found_retn
00009CAF F9                     <1>       stc
00009CB0 C3                     <1>       retn
                                <1>
                                <1> loc_ccd_path_not_found_retn:
00009CB1 B003                   <1>       mov    al, 3  ; Path not found
00009CB3 C3                     <1>       retn
                                <1>
                                <1> loc_ccd_load_FAT_root_dir:
00009CB4 803D[8D4E0100]02       <1>       cmp    byte [Current_FATType], 2
00009CBB 776B                   <1>       ja     short loc_ccd_load_FAT32_root_dir
                                <1>
                                <1>       ;mov   esi, [CCD_DriveDT]
                                <1>       ;push  esi
00009CBD E8B61D0000             <1>       call   load_FAT_root_directory
                                <1>       ;pop   edi ; Dos Drv Description Table
                                <1>
00009CC2 89F7                   <1>       mov    edi, esi
00009CC4 BE[EF560100]           <1>       mov    esi, PATH_Array
00009CC9 7297                   <1>       jc     short loc_ccd_retn_p
                                <1>
00009CCB 31C0                   <1>       xor    eax, eax
00009CCD EB78                   <1>         jmp  short loc_ccd_set_cdfc
                                <1>
                                <1> loc_ccd_load_root_dir:
00009CCF 803D[8D4E0100]01       <1>       cmp    byte [Current_FATType], 1
00009CD6 73DC                   <1>       jnb    short loc_ccd_load_FAT_root_dir
                                <1>
                                <1> loc_ccd_load_FS_root_dir:
```

```
29269 00009CD8 E8621E0000          <1>       call   load_FS_root_directory
29270 00009CDD EB5C                <1>       jmp    short pass_ccd_load_FAT_sub_directory
29271                              <1>
29272                              <1> loc_ccd_load_FS_sub_directory_next:
29273 00009CDF E85C1E0000          <1>       call   load_FS_sub_directory
29274 00009CE4 EB1F                <1>       jmp    short pass_ccd_set_dir_cluster_ptr
29275                              <1>
29276                              <1> loc_ccd_set_dir_cluster_ptr:
29277                              <1>       ; EDI = Directory Entry
29278 00009CE6 668B4714            <1>       mov    ax, [edi+20] ; First Cluster High Word
29279 00009CEA C1E010              <1>       shl    eax, 16
29280 00009CED 668B471A            <1>       mov    ax, [edi+26] ; First Cluster Low Word
29281                              <1>
29282 00009CF1 8B35[6F570100]      <1>       mov    esi, [CCD_DriveDT]
29283 00009CF7 803D[8D4E0100]01    <1>       cmp    byte [Current_FATType], 1
29284 00009CFE 72DF                <1>       jb     short loc_ccd_load_FS_sub_directory_next
29285                              <1>       ;push  esi
29286 00009D00 E8FE1D0000          <1>       call   load_FAT_sub_directory
29287                              <1>       ;pop   edi ; Dos Drv Description Table
29288                              <1>
29289                              <1> pass_ccd_set_dir_cluster_ptr:
29290                              <1>       ;mov   edi, esi
29291 00009D05 BE[EF560100]        <1>       mov    esi, PATH_Array
29292 00009D0A 7264                <1>       jc     short loc_ccd_retn_c
29293                              <1>
29294 00009D0C A1[BD560100]        <1>       mov    eax, [DirBuff_Cluster]
29295                              <1>
29296 00009D11 FE05[73570100]      <1>       inc    byte [CCD_Level]
29297 00009D17 0FB61D[73570100]    <1>       movzx  ebx, byte [CCD_Level]
29298 00009D1E C0E304              <1>       shl    bl, 4 ; * 16 (<= 128)
29299 00009D21 01DE                <1>       add    esi, ebx ; 19/02/2016
29300 00009D23 89460C              <1>       mov    [esi+12], eax
29301 00009D26 EB1F                <1>       jmp    short loc_ccd_set_cdfc
29302                              <1>
29303                              <1> loc_ccd_load_FAT32_root_dir:
29304 00009D28 BE[EF560100]        <1>       mov    esi, PATH_Array
29305 00009D2D 8B460C              <1>       mov    eax, [esi+12]
29306 00009D30 8B35[6F570100]      <1>       mov    esi, [CCD_DriveDT]
29307                              <1>
29308                              <1> loc_ccd_load_FAT_sub_directory:
29309                              <1>       ;push  esi
29310 00009D36 E8C81D0000          <1>       call   load_FAT_sub_directory
29311                              <1>       ;pop   edi ; Dos Drv Description Table
29312                              <1>
29313                              <1> pass_ccd_load_FAT_sub_directory:
29314                              <1>       ;mov   edi, esi
29315 00009D3B BE[EF560100]        <1>       mov    esi, PATH_Array
29316 00009D40 722E                <1>       jc     short loc_ccd_retn_c
29317                              <1>
29318 00009D42 A1[BD560100]        <1>       mov    eax, [DirBuff_Cluster]
29319                              <1>
29320                              <1> loc_ccd_set_cdfc:
29321 00009D47 8A0D[73570100]      <1>       mov    cl, [CCD_Level]
29322 00009D4D 880D[8C4E0100]      <1>       mov    [Current_Dir_Level], cl
29323 00009D53 A3[884E0100]        <1>       mov    [Current_Dir_FCluster], eax
29324                              <1>
29325 00009D58 8A2D[74570100]      <1>       mov    ch, [Last_Dir_Level]
29326 00009D5E 38E9                <1>       cmp    cl, ch
29327 00009D60 0F821CFFFFFF        <1>       jb     loc_ccd_load_child_dir_next
29328                              <1>
29329 00009D66 803D[7D570100]CD    <1>       cmp    byte [CD_COMMAND], 0CDh ;'CD' command or another
29330 00009D6D 741E                <1>       je     short loc_ccd_save_current_dir
29331                              <1>
29332                              <1>         ; jne -> don't save, restore (the previous cdir) later !
29333                              <1>         ; (saving the cdir would prevent previous cdir restoration!)
29334                              <1>
29335 00009D6F F8                  <1>       clc
29336                              <1>
29337                              <1> loc_ccd_retn_c:
29338 00009D70 8B3D[6F570100]      <1>       mov    edi, [CCD_DriveDT]
29339 00009D76 C3                  <1>       retn
29340                              <1>
29341                              <1> loc_ccd_load_sub_directory:
29342 00009D77 8B35[6F570100]      <1>       mov    esi, [CCD_DriveDT]
29343 00009D7D 803D[8D4E0100]01    <1>       cmp    byte [Current_FATType], 1
29344 00009D84 73B0                <1>       jnb    short loc_ccd_load_FAT_sub_directory
29345 00009D86 E8B51D0000          <1>       call   load_FS_sub_directory
29346 00009D8B EBAE                <1>       jmp    short pass_ccd_load_FAT_sub_directory
29347                              <1>
29348                              <1> loc_ccd_save_current_dir:
29349 00009D8D BE[EF560100]        <1>       mov    esi, PATH_Array ; 19/02/2016
29350 00009D92 8B3D[6F570100]      <1>       mov    edi, [CCD_DriveDT]
29351 00009D98 57                  <1>       push   edi
29352 00009D99 83C77F              <1>         add     edi, LD_CDirLevel
29353 00009D9C 880F                <1>       mov    [edi], cl
29354 00009D9E 47                  <1>       inc    edi ; LD_CurrentDirectory
29355 00009D9F 56                  <1>       push   esi
29356                              <1>       ;mov   ecx, 32  ; always < 65536 (in this procedure)
29357 00009DA0 66B92000            <1>       mov    cx, 32
29358 00009DA4 F3A5                <1>       rep    movsd
29359                              <1>       ; Current directory has been saved to
29360                              <1>       ; the DOS drive description table, cdir area !
29361 00009DA6 5E                  <1>       pop    esi  ; PATH_Array
29362 00009DA7 5F                  <1>       pop    edi  ; Dos Drv Description Table
29363                              <1>
29364 00009DA8 C3                  <1>       retn
29365                              <1>
29366                              <1> parse_dir_name:
29367                              <1>       ; 11/02/2016
29368                              <1>       ; 10/02/2016
29369                              <1>       ; 07/02/2016 (TRDOS 386 = TRDOS v2.0)
29370                              <1>       ; 18/09/2011
```

```
29371                                  <1>       ; 17/10/2009
29372                                  <1>       ; INPUT ->
29373                                  <1>       ;      ESI = ASCIIZ Directory String Address
29374                                  <1>       ;      AL = Current Directory Level
29375                                  <1>       ;      EDI = Destination Adress
29376                                  <1>       ;           (8 levels, each one 12+4 byte)
29377                                  <1>       ; OUTPUT ->
29378                                  <1>       ;      EDI = Dir Entry Formatted Array
29379                                  <1>       ;           with zero cluster pointer at the last level
29380                                  <1>       ;      AH = Last Dir Level
29381                                  <1>       ;      AL = Current Dir Level
29382                                  <1>       ;
29383                                  <1>       ; (esi, ebx, ecx will be changed)
29384                                  <1>
29385                                  <1>       ;mov   [PATH_Array_Ptr], edi
29386 00009DA9 88C4                   <1>       mov    ah, al
29387 00009DAB 66A3[14580100]        <1>       mov    [PATH_CDLevel], ax
29388                                  <1> repeat_ppdn_check_slash:
29389 00009DB1 AC                     <1>       lodsb
29390 00009DB2 3C2F                   <1>       cmp    al, '/'
29391 00009DB4 74FB                   <1>       je     short repeat_ppdn_check_slash
29392 00009DB6 3C21                   <1>       cmp    al, 21h
29393 00009DB8 7219                   <1>       jb     short loc_ppdn_retn
29394 00009DBA 57                     <1>       push   edi
29395                                  <1> loc_ppdn_get_dir_name:
29396 00009DBB B90C000000             <1>       mov    ecx, 12
29397 00009DC0 BF[16580100]           <1>       mov    edi, Dir_File_Name
29398                                  <1> repeat_ppdn_get_dir_name:
29399 00009DC5 AA                     <1>       stosb
29400 00009DC6 AC                     <1>       lodsb
29401 00009DC7 3C2F                   <1>       cmp    al, '/'
29402 00009DC9 740A                   <1>       je     short loc_check_level_dot_conv_dir_name
29403 00009DCB 3C20                   <1>       cmp    al, 20h
29404 00009DCD 7605                   <1>       jna    short loc_ppdn_end_of_path_scan
29405 00009DCF E2F4                   <1>       loop   repeat_ppdn_get_dir_name
29406 00009DD1 5F                     <1>       pop    edi
29407 00009DD2 F9                     <1>       stc
29408                                  <1> loc_ppdn_retn:
29409 00009DD3 C3                     <1>       retn
29410                                  <1>
29411                                  <1> loc_ppdn_end_of_path_scan:
29412 00009DD4 4E                     <1>       dec    esi
29413                                  <1> loc_check_level_dot_conv_dir_name:
29414 00009DD5 31C0                   <1>       xor    eax, eax
29415 00009DD7 AA                     <1>       stosb
29416 00009DD8 89F3                   <1>       mov    ebx, esi
29417 00009DDA BE[16580100]           <1>       mov    esi, Dir_File_Name
29418 00009DDF AC                     <1>       lodsb
29419                                  <1> repeat_ppdn_name_check_dot:
29420 00009DE0 3C2E                   <1>       cmp    al, '.'
29421 00009DE2 7509                   <1>       jne    short loc_ppdn_convert_sub_dir_name
29422                                  <1> repeat_ppdn_name_dot_dot:
29423 00009DE4 AC                     <1>       lodsb
29424 00009DE5 3C2E                   <1>       cmp    al, '.'
29425 00009DE7 743E                   <1>       je     short loc_ppdn_dot_dot
29426 00009DE9 3C21                   <1>       cmp    al, 21h
29427 00009DEB 7226                   <1>       jb     short pass_ppdn_convert_sub_dir_name
29428                                  <1> loc_ppdn_convert_sub_dir_name:
29429 00009DED 8A25[15580100]         <1>       mov    ah, [PATH_Level]
29430 00009DF3 80FC07                 <1>       cmp    ah, 7
29431 00009DF6 731B                   <1>       jnb    short pass_ppdn_convert_sub_dir_name
29432 00009DF8 FEC4                   <1>       inc    ah
29433 00009DFA 8825[15580100]         <1>       mov    [PATH_Level], ah
29434 00009E00 BE[16580100]           <1>       mov    esi, Dir_File_Name
29435                                  <1>       ;mov   edi, [PATH_Array_Ptr]
29436 00009E05 B010                   <1>       mov    al, 16
29437 00009E07 F6E4                   <1>       mul    ah
29438 00009E09 8B3C24                 <1>       mov    edi, [esp]
29439                                  <1>       ;push  edi
29440 00009E0C 01C7                   <1>       add    edi, eax
29441 00009E0E E828030000             <1>       call   convert_file_name
29442                                  <1>       ;pop   edi
29443                                  <1> pass_ppdn_convert_sub_dir_name:
29444 00009E13 89DE                   <1>       mov    esi, ebx
29445                                  <1> repeat_ppdn_check_last_slash:
29446 00009E15 AC                     <1>       lodsb
29447 00009E16 3C2F                   <1>       cmp    al, '/'
29448 00009E18 74FB                   <1>       je     short repeat_ppdn_check_last_slash
29449 00009E1A 3C21                   <1>       cmp    al, 21h
29450 00009E1C 739D                   <1>       jnb    short loc_ppdn_get_dir_name
29451                                  <1> end_of_parse_dir_name:
29452 00009E1E 5F                     <1>       pop    edi
29453 00009E1F F5                     <1>       cmc
29454                                  <1>       ;mov   al, [PATH_CDLevel]
29455                                  <1>       ;mov   ah, [PATH_Level]
29456 00009E20 66A1[14580100]         <1>       mov    ax, [PATH_CDLevel]
29457 00009E26 C3                     <1>       retn
29458                                  <1>
29459                                  <1> loc_ppdn_dot_dot:
29460 00009E27 AC                     <1>       lodsb
29461 00009E28 3C21                   <1>       cmp    al, 21h
29462 00009E2A 73F2                   <1>       jnb    short end_of_parse_dir_name
29463                                  <1> loc_ppdn_dot_dot_prev_level:
29464 00009E2C 66A1[14580100]         <1>       mov    ax, [PATH_CDLevel]
29465 00009E32 80EC01                 <1>       sub    ah, 1
29466 00009E35 80D400                 <1>       adc    ah, 0
29467 00009E38 38E0                   <1>       cmp    al, ah
29468 00009E3A 7602                   <1>       jna    short pass_ppdn_set_al_to_ah
29469 00009E3C 88E0                   <1>       mov    al, ah
29470                                  <1> pass_ppdn_set_al_to_ah:
29471 00009E3E 66A3[14580100]         <1>       mov    [PATH_CDLevel], ax
29472 00009E44 EBCD                   <1>       jmp    short pass_ppdn_convert_sub_dir_name
```

```
                                <1>
                                <1> locate_current_dir_file:
                                <1>       ; 14/02/2016
                                <1>       ; 13/02/2016
                                <1>       ; 10/02/2016
                                <1>       ; 06/02/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>       ; 14/08/2010
                                <1>       ; 19/09/2009
                                <1>         ; 2005
                                <1>       ; INPUT ->
                                <1>       ;     ESI = DOS DirEntry Format FileName Address
                                <1>       ;     AL = Attributes Mask
                                <1>       ;     (<AL AND EntryAttrib> must be equal to AL)
                                <1>       ;     AH = Negative Attributes Mask (If AH>0)
                                <1>       ;     (<AH AND EntryAttrib> must be ZERO)
                                <1>       ;     CH > 0 Find First Free Dir Entry or Deleted Entry
                                <1>       ;     CL = 0 -> Return the First Free Dir Entry
                                <1>       ;     CL = E5h -> Return the 1st deleted entry
                                <1>       ;     CL = FFh -> Return the 1st deleted or free entry
                                <1>       ;     CL > 0 and CL <> E5h and CL <> FFh -> Return the first
                                <1>       ;          proper entry (which fits with Atributes Masks)
                                <1>       ;     CX = 0 Find Valid File/Directory/VolumeName
                                <1>       ;     ? = Any One Char
                                <1>       ;     * = Every Chars
                                <1>       ; OUTPUT ->
                                <1>       ;     EDI = Directory Entry Address (in Directory Buffer)
                                <1>       ;     ESI = DOS DirEntry Format FileName Address
                                <1>       ;     CF = 0 -> No Error, Proper Entry,
                                <1>       ;     DL = Attributes
                                <1>       ;     DH = Previous Entry Attr (LongName Check)
                                <1>       ;     AL > 0 -> Ambiguous filename wildcard "?" used
                                <1>       ;     AH > 0 -> Ambiguous filename wildcard "*" used
                                <1>       ;     AX = 0 -> Filename full fits with directory entry
                                <1>       ;     CH = The 1st Name Char of Current Dir Entry
                                <1>       ;     CF = 1 -> Proper entry not found, Error Code in EAX/AL
                                <1>       ;     CL = 0 and CH = 0 -> Free Entry (End Of Dir)
                                <1>       ;     CL = 0 and CH = E5h -> Deleted Entry fits with filters
                                <1>       ;     CL > 0 -> Entry not found, CH invalid
                                <1>       ;     CF = 0 ->
                                <1>       ;     EBX = Current Directory Entry Index/Number (BX)
                                <1>
                                <1>       ;mov   word [DirBuff_EntryCounter], 0 ; Zero Based
                                <1>
00009E46 8935[77570100]         <1>       mov    [CDLF_FNAddress], esi
00009E4C 66A3[75570100]         <1>       mov    [CDLF_AttributesMask], ax
00009E52 66890D[7B570100]       <1>       mov    [CDLF_DEType], cx
                                <1>
00009E59 31DB                   <1>       xor    ebx, ebx
00009E5B 881D[8C570100]         <1>       mov    [PreviousAttr], bl ; 0  ; 13/02/2016
                                <1>
00009E61 8A3D[8E4E0100]         <1>       mov    bh, [Current_Drv]
00009E67 381D[B8560100]         <1>       cmp    byte [DirBuff_ValidData], bl ; 0
00009E6D 761D                   <1>       jna    short loc_lcdf_reload_current_dir2
00009E6F 8A1D[B6560100]         <1>         mov     bl, [DirBuff_DRV]
00009E75 80EB41                 <1>       sub    bl, 'A'
00009E78 38DF                   <1>       cmp    bh, bl
00009E7A 750E                   <1>       jne    short loc_lcdf_reload_current_dir1
00009E7C 8B15[BD560100]         <1>       mov    edx, [DirBuff_Cluster]
00009E82 3B15[884E0100]         <1>       cmp    edx, [Current_Dir_FCluster]
00009E88 7412                   <1>       je     short loc_cdir_locatefile_search
                                <1>
                                <1> loc_lcdf_reload_current_dir1:
00009E8A 30DB                   <1>       xor    bl, bl
                                <1> loc_lcdf_reload_current_dir2:
00009E8C 89DE                   <1>       mov    esi, ebx
00009E8E 81C600010900           <1>         add     esi, Logical_DOSDisks
00009E94 E872000000             <1>       call   reload_current_directory
00009E99 735B                   <1>       jnc    short loc_locatefile_search_again
00009E9B C3                     <1>       retn
                                <1>
                                <1> loc_cdir_locatefile_search:
00009E9C 31DB                   <1>       xor    ebx, ebx
00009E9E E8A5000000             <1>       call   find_directory_entry
00009EA3 7349                   <1>       jnc    short loc_cdir_locate_file_retn
                                <1>
                                <1> loc_locatefile_check_stc_reason:
00009EA5 08ED                   <1>       or     ch, ch
00009EA7 7444                   <1>       jz     short loc_cdir_locate_file_stc_retn
                                <1>
                                <1> loc_locatefile_check_next_entryblock:
00009EA9 8A3D[8E4E0100]         <1>       mov    bh, [Current_Drv]
00009EAF 28DB                   <1>       sub    bl, bl
00009EB1 0FB7F3                 <1>       movzx  esi, bx
00009EB4 81C600010900           <1>         add     esi, Logical_DOSDisks
                                <1>
00009EBA 803D[8C4E0100]00       <1>       cmp    byte [Current_Dir_Level], 0
00009EC1 760A                   <1>       jna    short loc_locatefile_check_FAT_type
                                <1>
00009EC3 803D[8D4E0100]01       <1>       cmp    byte [Current_FATType], 1
00009ECA 730A                   <1>       jnb    short loc_locatefile_load_subdir_cluster
00009ECC C3                     <1>       retn
                                <1>
                                <1> loc_locatefile_check_FAT_type:
00009ECD 803D[8D4E0100]03       <1>       cmp    byte [Current_FATType], 3
00009ED4 7218                   <1>       jb     short loc_cdir_locate_file_retn
                                <1>
                                <1> loc_locatefile_load_subdir_cluster:
00009ED6 A1[BD560100]           <1>       mov    eax, [DirBuff_Cluster]
00009EDB E83D1A0000             <1>       call   get_next_cluster
00009EE0 730D                   <1>       jnc    short loc_locatefile_next_cluster
00009EE2 09C0                   <1>       or     eax, eax
00009EE4 7507                   <1>       jnz    short loc_locatefile_drive_not_ready_read_err
```

```
29575 00009EE6 F9                <1>       stc
29576                            <1> loc_locatefile_file_notfound:
29577 00009EE7 B802000000        <1>       mov   eax, 2 ; File/Directory/VolName not found
29578 00009EEC C3                <1>       retn
29579                            <1>
29580                            <1> loc_locatefile_drive_not_ready_read_err:
29581                            <1>       ;mov   eax, 17 ;Drive not ready or read error
29582                            <1> loc_cdir_locate_file_stc_retn:
29583 00009EED F5                <1>       cmc ;stc
29584                            <1> loc_cdir_locate_file_retn:
29585 00009EEE C3                <1>       retn
29586                            <1>
29587                            <1> loc_locatefile_next_cluster:
29588 00009EEF E80F1C0000        <1>       call   load_FAT_sub_directory
29589                            <1>       ;jc    short loc_locatefile_drive_not_ready_read_err
29590 00009EF4 72F8              <1>       jc     short loc_cdir_locate_file_retn
29591                            <1>
29592                            <1> loc_locatefile_search_again:
29593 00009EF6 8B35[77570100]    <1>       mov    esi, [CDLF_FNAddress]
29594 00009EFC 66A1[75570100]    <1>       mov    ax, [CDLF_AttributesMask]
29595 00009F02 668B0D[7B570100]  <1>       mov    cx, [CDLF_DEType]
29596 00009F09 EB91              <1>       jmp    short loc_cdir_locatefile_search
29597                            <1>
29598                            <1> reload_current_directory:
29599                            <1>       ; 06/02/2016 (TRDOS 386 = TRDOS v2.0)
29600                            <1>       ; 13/06/2010
29601                            <1>       ; 22/09/2009
29602                            <1>         ;
29603                            <1>       ; INPUT ->
29604                            <1>       ;      ESI = Dos drive description table address
29605                            <1>
29606                            <1>       ;mov   al, [esi+LD_FATType]
29607 00009F0B A0[8D4E0100]      <1>       mov    al, [Current_FATType]
29608 00009F10 3C02              <1>       cmp    al, 2
29609 00009F12 7729              <1>       ja     short loc_reload_FAT_sub_directory
29610 00009F14 8A25[8C4E0100]    <1>       mov    ah, [Current_Dir_Level]
29611 00009F1A 08C0              <1>       or     al, al
29612 00009F1C 740A              <1>       jz     short loc_reload_FS_directory
29613 00009F1E 08E4              <1>       or     ah, ah
29614 00009F20 751B              <1>       jnz    short loc_reload_FAT_sub_directory
29615                            <1> loc_reload_FAT_12_16_root_directory:
29616 00009F22 E8511B0000        <1>       call   load_FAT_root_directory
29617 00009F27 C3                <1>       retn
29618                            <1> loc_reload_FS_directory:
29619 00009F28 20E4              <1>       and    ah, ah
29620 00009F2A 7506              <1>       jnz    short loc_reload_FS_sub_directory
29621                            <1> loc_reload_FS_root_directory:
29622 00009F2C E80E1C0000        <1>       call   load_FS_root_directory
29623 00009F31 C3                <1>       retn
29624                            <1> loc_reload_FS_sub_directory:
29625 00009F32 A1[884E0100]      <1>       mov    eax, [Current_Dir_FCluster]
29626 00009F37 E8041C0000        <1>       call   load_FS_sub_directory
29627 00009F3C C3                <1>       retn
29628                            <1> loc_reload_FAT_sub_directory:
29629 00009F3D A1[884E0100]      <1>       mov    eax, [Current_Dir_FCluster]
29630 00009F42 E8BC1B0000        <1>       call   load_FAT_sub_directory
29631 00009F47 C3                <1>       retn
29632                            <1>
29633                            <1> find_directory_entry:
29634                            <1>       ; 14/02/2016
29635                            <1>       ; 13/02/2016
29636                            <1>       ; 10/02/2016
29637                            <1>       ; 06/02/2016 (TRDOS 386 = TRDOS v2.0)
29638                            <1>       ; 14/08/2010 (DIR.ASM, "proc_find_direntry")
29639                            <1>       ; 19/09/2009
29640                            <1>       ; 2005
29641                            <1>       ; INPUT ->
29642                            <1>       ;      ESI = Sub Dir or File Name Address
29643                            <1>       ;      AL = Attributes Mask
29644                            <1>       ;      (<AL AND EntryAttrib> must be equal to AL)
29645                            <1>       ;      AH = Negative Attributes Mask (If AH>0)
29646                            <1>       ;      (<AH AND EntryAttrib> must be ZERO)
29647                            <1>       ;      CH > 0 Find First Free Dir Entry or Deleted Entry
29648                            <1>       ;      CL = 0 -> Return the First Free Dir Entry
29649                            <1>       ;      CL = E5h -> Return the 1st deleted entry
29650                            <1>       ;      CL = FFh -> Return the 1st deleted or free entry
29651                            <1>       ;      CL > 0 and CL <> E5h and CL <> FFh -> Return the first
29652                            <1>       ;            proper entry (which fits with Atributes Masks)
29653                            <1>       ;      CX = 0 -> Find Valid File/Directory/VolumeName
29654                            <1>       ;      ? = Any One Char
29655                            <1>       ;      * = Every Chars
29656                            <1>       ;      EBX = Current Dir Entry (BX)
29657                            <1>       ;
29658                            <1>       ; OUTPUT ->
29659                            <1>       ;      EDI = Directory Entry Address (in DirectoryBuffer)
29660                            <1>       ;      ESI = Sub Dir or File Name Address
29661                            <1>       ;      CF = 0 -> No Error, Proper Entry,
29662                            <1>       ;      DL = Attributes
29663                            <1>       ;      DH = Previous Entry Attr (LongName Check)
29664                            <1>       ;      AL > 0 -> Ambiguous filename wildcard "?" used
29665                            <1>       ;      AH > 0 -> Ambiguous filename wildcard "*" used
29666                            <1>       ;      AX = 0 -> Filename full fits with directory entry
29667                            <1>       ;      EBX = CurrentDirEntry (BX)
29668                            <1>       ;      CH = The 1st Name Char of Current Dir Entry
29669                            <1>       ;      CF = 1 -> Proper entry not found, Error Code in AX/AL
29670                            <1>       ;      CL = 0 and CH = 0 -> Free Entry (End Of Dir)
29671                            <1>       ;      CL = 0 and CH = E5h -> Deleted Entry fits with filters
29672                            <1>       ;      CL > 0 -> Entry not found, CH invalid
29673                            <1>       ;
29674                            <1>       ; (EAX, EBX, ECX, EDX, EDI, EBP will be changed)
29675                            <1>
29676 00009F48 663B1D[BB560100]  <1>       cmp    bx, [DirBuff_LastEntry]
```

```
29677 00009F4F 0F8739010000        <1>         ja      loc_ffde_stc_retn_255
29678                              <1>
29679                              <1>       ;mov    [DirBuff_CurrentEntry], bx
29680                              <1>
29681 00009F55 BF00000800          <1>       mov    edi, Directory_Buffer
29682 00009F5A 66A3[88570100]      <1>       mov    [FDE_AttrMask], ax
29683                              <1>
29684 00009F60 29C0                <1>       sub    eax, eax
29685                              <1>
29686                              <1>       ;;mov  [PreviousAttr], al ; 0 ;; 13/02/2016
29687 00009F62 66A3[8A570100]      <1>       mov    [AmbiguousFileName], ax ; 0
29688                              <1>
29689 00009F68 6689D8              <1>       mov    ax, bx
29690 00009F6B 66C1E005            <1>       shl    ax, 5 ; ; * 32 ; Directory entry size
29691 00009F6F 01C7                <1>       add     edi, eax
29692                              <1>
29693 00009F71 08ED                <1>       or     ch, ch
29694 00009F73 0F852C010000        <1>         jnz     loc_find_free_deleted_entry_0
29695                              <1>
29696 00009F79 08C9                <1>       or      cl, cl
29697 00009F7B 0F850D010000        <1>         jnz     loc_ffde_stc_retn_255
29698                              <1>
29699                              <1> check_find_dir_entry:
29700 00009F81 66A1[88570100]      <1>       mov    ax, [FDE_AttrMask]
29701 00009F87 8A2F                <1>       mov    ch, [edi]
29702 00009F89 80FD00              <1>       cmp     ch, 0 ; Is it never used entry?
29703 00009F8C 0F86FF000000        <1>       jna    loc_find_direntry_stc_retn
29704 00009F92 56                  <1>       push   esi
29705 00009F93 8A570B              <1>       mov    dl, [edi+0Bh] ; File attributes
29706 00009F96 80FDE5              <1>       cmp    ch, 0E5h ; Is it a deleted file?
29707 00009F99 746D                <1>       je     short loc_find_dir_next_entry_prevdeleted
29708                              <1>
29709 00009F9B 80FA0F              <1>       cmp     dl, 0Fh ; longname sub component check
29710 00009F9E 7505                <1>       jne     short loc_check_attributes_mask
29711 00009FA0 E8ED010000          <1>       call   save_longname_sub_component
29712                              <1>
29713                              <1> loc_check_attributes_mask:
29714 00009FA5 88C6                <1>       mov    dh, al
29715 00009FA7 20D6                <1>       and    dh, dl
29716 00009FA9 38F0                <1>       cmp    al, dh
29717 00009FAB 0F85BA000000        <1>         jne     loc_find_dir_next_entry
29718 00009FB1 20D4                <1>       and    ah, dl
29719 00009FB3 0F85B2000000        <1>         jnz     loc_find_dir_next_entry
29720 00009FB9 80FA0F              <1>       cmp    dl, 0Fh
29721 00009FBC 751A                <1>       jne    short pass_direntry_attr_check
29722                              <1>
29723 00009FBE 3C0F                <1>       cmp    al, 0Fh ; AL = 0Fh -> find long name
29724 00009FC0 0F85A5000000        <1>         jne     loc_find_dir_next_entry
29725                              <1>
29726 00009FC6 5E                  <1>       pop    esi
29727 00009FC7 6631C0              <1>       xor    ax, ax
29728 00009FCA 8A35[8C570100]      <1>       mov    dh, [PreviousAttr]
29729 00009FD0 66891D[B9560100]    <1>       mov    [DirBuff_CurrentEntry], bx
29730 00009FD7 C3                  <1>       retn
29731                              <1>
29732                              <1> pass_direntry_attr_check:
29733 00009FD8 89FD                <1>       mov    ebp, edi ; 14/02/2016
29734 00009FDA B908000000          <1>       mov    ecx, 8
29735                              <1> loc_lodsb_find_dir:
29736 00009FDF AC                  <1>       lodsb
29737 00009FE0 3C2A                <1>       cmp    al, '*'
29738 00009FE2 7508                <1>       jne    short pass_fde_ambiguous1_check
29739 00009FE4 FE05[8B570100]      <1>         inc     byte [AmbiguousFileName+1]
29740 00009FEA EB28                <1>       jmp    short loc_check_direntry_extension
29741                              <1>
29742                              <1> pass_fde_ambiguous1_check:
29743 00009FEC 3C3F                <1>       cmp    al, '?'
29744 00009FEE 750D                <1>       jne    short pass_fde_ambiguous2_check
29745 00009FF0 FE05[8A570100]      <1>       inc    byte [AmbiguousFileName]
29746 00009FF6 803F20              <1>       cmp    byte [edi], 20h
29747 00009FF9 764E                <1>       jna    short loc_find_dir_next_entry_ebp
29748 00009FFB EB14                <1>       jmp    short loc_scasb_find_dir_inc_di
29749                              <1>
29750                              <1> pass_fde_ambiguous2_check:
29751 00009FFD 3C20                <1>       cmp    al, 20h
29752 00009FFF 750C                <1>       jne    short loc_scasb_find_dir
29753 0000A001 803F20              <1>       cmp    byte [edi], 20h
29754 0000A004 7543                <1>       jne    short loc_find_dir_next_entry_ebp
29755 0000A006 EB0C                <1>       jmp    short loc_check_direntry_extension
29756                              <1>
29757                              <1> loc_find_dir_next_entry_prevdeleted:
29758 0000A008 80CA80              <1>       or     dl, 80h  ; Bit 7 -> deleted entry sign
29759 0000A00B EB5E                <1>       jmp    short loc_find_dir_next_entry
29760                              <1>
29761                              <1> loc_scasb_find_dir:
29762 0000A00D 3A07                <1>       cmp    al, [edi]
29763 0000A00F 7538                <1>       jne    short loc_find_dir_next_entry_ebp
29764                              <1> loc_scasb_find_dir_inc_di:
29765 0000A011 47                  <1>       inc    edi
29766 0000A012 E2CB                <1>       loop   loc_lodsb_find_dir
29767                              <1>
29768                              <1> loc_check_direntry_extension:
29769 0000A014 BE08000000          <1>       mov    esi, 8
29770 0000A019 89F7                <1>       mov    edi, esi ; 8
29771 0000A01B 033424              <1>       add    esi, [esp] ; Sub Dir or File Name Address
29772 0000A01E 01EF                <1>       add    edi, ebp
29773 0000A020 B103                <1>       mov    cl, 3
29774                              <1> loc_lodsb_find_dir_ext:
29775 0000A022 AC                  <1>       lodsb
29776 0000A023 3C2A                <1>       cmp    al, '*'
29777 0000A025 7508                <1>       jne    short pass_fde_ambiguous3_check
29778 0000A027 FE05[8B570100]      <1>       inc    byte [AmbiguousFileName+1]
```

```
29779 0000A02D EB1E               <1>       jmp    short loc_find_dir_proper_direntry
29780                             <1>
29781                             <1> pass_fde_ambiguous3_check:
29782 0000A02F 3C3F               <1>       cmp    al, '?'
29783 0000A031 750D               <1>       jne    short pass_fde_ambiguous4_check
29784 0000A033 FE05[8A570100]     <1>       inc    byte [AmbiguousFileName]
29785 0000A039 803F20             <1>       cmp    byte [edi], 20h
29786 0000A03C 760B               <1>       jna    short loc_find_dir_next_entry_ebp
29787 0000A03E EB49               <1>       jmp    short loc_scasb_find_dir_ext_inc_di
29788                             <1>
29789                             <1> pass_fde_ambiguous4_check:
29790 0000A040 3C20               <1>       cmp    al, 20h
29791 0000A042 7541               <1>       jne    short loc_scasb_find_dir_ext
29792 0000A044 803F20             <1>       cmp    byte [edi], 20h
29793 0000A047 7404               <1>       je     short loc_find_dir_proper_direntry
29794                             <1>
29795                             <1> loc_find_dir_next_entry_ebp:
29796 0000A049 89EF               <1>       mov    edi, ebp ; 14/02/2016
29797 0000A04B EB1E               <1>       jmp    short loc_find_dir_next_entry
29798                             <1>
29799                             <1> loc_find_dir_proper_direntry:
29800 0000A04D 30C9               <1>       xor    cl, cl
29801                             <1> loc_find_dir_proper_direntry_1:
29802 0000A04F 5E                 <1>       pop    esi
29803 0000A050 89EF               <1>         mov     edi, ebp
29804 0000A052 8A2F               <1>       mov    ch, [edi]
29805 0000A054 8A570B             <1>       mov     dl, [edi+0Bh] ; Dir entry attributes
29806 0000A057 66A1[8A570100]     <1>       mov    ax, [AmbiguousFileName]
29807                             <1> loc_find_dir_proper_direntry_2:
29808 0000A05D 8A35[8C570100]     <1>       mov     dh, [PreviousAttr]
29809 0000A063 66891D[B9560100]   <1>       mov    [DirBuff_CurrentEntry], bx
29810 0000A06A C3                 <1>       retn
29811                             <1>
29812                             <1> loc_find_dir_next_entry:
29813 0000A06B 8815[8C570100]     <1>       mov    byte [PreviousAttr], dl ; LongName check
29814                             <1> loc_find_dir_next_entry_1:
29815 0000A071 5E                 <1>       pop    esi
29816 0000A072 83C720             <1>       add    edi, 32
29817                             <1>       ;inc   word [DirBuff_EntryCounter]
29818 0000A075 6643               <1>       inc    bx
29819 0000A077 663B1D[BB560100]   <1>       cmp    bx, [DirBuff_LastEntry]
29820 0000A07E 770E               <1>       ja     short loc_ffde_stc_retn_255
29821 0000A080 E9FCFEFFFF         <1>         jmp     check_find_dir_entry
29822                             <1>
29823                             <1> loc_scasb_find_dir_ext:
29824 0000A085 3A07               <1>       cmp    al, [edi]
29825 0000A087 75C0               <1>       jne    short loc_find_dir_next_entry_ebp
29826                             <1> loc_scasb_find_dir_ext_inc_di:
29827 0000A089 47                 <1>       inc    edi
29828 0000A08A E296               <1>       loop    loc_lodsb_find_dir_ext
29829 0000A08C EBC1               <1>       jmp    short loc_find_dir_proper_direntry_1
29830                             <1>
29831                             <1> loc_ffde_stc_retn_255:
29832                             <1>       ;mov   cx, 0FFFFh
29833 0000A08E 31C9               <1>       xor    ecx, ecx
29834 0000A090 49                 <1>       dec    ecx ; 0FFFFFFFFh
29835                             <1>       ;xor   eax, eax
29836                             <1> loc_find_direntry_stc_retn:
29837                             <1> loc_check_ffde_retn_1:
29838                             <1>       ;mov   ax, 2
29839 0000A091 B802000000         <1>       mov    eax, 2 ; File Not Found
29840 0000A096 8A35[8C570100]     <1>       mov    dh, [PreviousAttr]
29841 0000A09C 66891D[B9560100]   <1>       mov    [DirBuff_CurrentEntry], bx
29842 0000A0A3 F9                 <1>       stc
29843 0000A0A4 C3                 <1>       retn
29844                             <1>
29845                             <1> loc_find_free_deleted_entry_0:
29846 0000A0A5 66A1[88570100]     <1>       mov    ax, [FDE_AttrMask]
29847 0000A0AB 8A2F               <1>       mov    ch, [edi]
29848 0000A0AD 8A570B             <1>       mov    dl, [edi+0Bh] ; File attributes
29849 0000A0B0 08C9               <1>       or     cl, cl
29850 0000A0B2 7407               <1>       jz     short loc_check_ffde_0_repeat
29851                             <1>       ;cmp   cl, 0E5h
29852                             <1>       ;je    short pass_loc_check_ffde_0_err
29853 0000A0B4 80F9FF             <1>       cmp    cl, 0FFh
29854 0000A0B7 7432               <1>       je     short loc_find_free_deleted_entry_1
29855 0000A0B9 EB4D               <1>       jmp    short pass_loc_check_ffde_0_err
29856                             <1>
29857                             <1> loc_check_ffde_0_repeat:
29858 0000A0BB 08ED               <1>       or     ch, ch
29859 0000A0BD 7511               <1>       jnz    short loc_check_ffde_0_next
29860                             <1>
29861                             <1> loc_check_ffde_retn_2:
29862 0000A0BF 6629C0             <1>       sub    ax, ax
29863 0000A0C2 8A35[8C570100]     <1>       mov    dh, [PreviousAttr]
29864 0000A0C8 66891D[B9560100]   <1>       mov    [DirBuff_CurrentEntry], bx
29865 0000A0CF C3                 <1>       retn
29866                             <1>
29867                             <1> loc_check_ffde_0_next:
29868 0000A0D0 6643               <1>       inc    bx
29869 0000A0D2 83C720             <1>       add    edi, 32
29870                             <1>       ;inc   word [DirBuff_EntryCounter]
29871                             <1>
29872 0000A0D5 663B1D[BB560100]   <1>         cmp  bx, [DirBuff_LastEntry]
29873 0000A0DC 77B0               <1>       ja     short loc_ffde_stc_retn_255
29874 0000A0DE 8815[8C570100]     <1>       mov    [PreviousAttr], dl
29875 0000A0E4 8A2F               <1>       mov    ch, [edi]
29876 0000A0E6 8A570B             <1>       mov    dl, [edi+0Bh] ; file attributes
29877 0000A0E9 EBD0               <1>       jmp    short loc_check_ffde_0_repeat
29878                             <1>
29879                             <1> loc_find_free_deleted_entry_1:
29880 0000A0EB 28D2               <1>       sub    dl, dl
```

```
29881                                  <1> loc_find_free_deleted_entry_2:
29882 0000A0ED 20ED                    <1>       and    ch, ch
29883 0000A0EF 74CE                    <1>       jz     short loc_check_ffde_retn_2
29884 0000A0F1 80FDE5                  <1>       cmp    ch, 0E5h
29885 0000A0F4 74C9                    <1>       je     short loc_check_ffde_retn_2
29886 0000A0F6 6643                    <1>       inc    bx
29887 0000A0F8 83C720                  <1>       add    edi, 32
29888 0000A0FB 663B1D[BB560100]        <1>       cmp    bx, [DirBuff_LastEntry]
29889 0000A102 778A                    <1>       ja     short loc_ffde_stc_retn_255
29890 0000A104 8A2F                    <1>       mov    ch, [edi]
29891 0000A106 EBE5                    <1>       jmp    short loc_find_free_deleted_entry_2
29892                                  <1>
29893                                  <1> pass_loc_check_ffde_0_err:
29894 0000A108 38CD                    <1>       cmp    ch, cl
29895 0000A10A 741F                    <1>       je     short loc_check_ffde_attrib
29896                                  <1>
29897 0000A10C 6643                    <1>       inc    bx
29898 0000A10E 83C720                  <1>       add    edi, 32
29899 0000A111 663B1D[BB560100]        <1>       cmp    bx, [DirBuff_LastEntry]
29900 0000A118 0F8770FFFFFF            <1>         ja     loc_ffde_stc_retn_255
29901 0000A11E 8815[8C570100]          <1>       mov    [PreviousAttr], dl
29902 0000A124 8A2F                    <1>       mov    ch, [edi]
29903 0000A126 8A570B                  <1>       mov    dl, [edi+0Bh]
29904 0000A129 EBDD                    <1>       jmp    short pass_loc_check_ffde_0_err
29905                                  <1>
29906                                  <1> loc_check_ffde_attrib:
29907 0000A12B 88C6                    <1>       mov    dh, al
29908 0000A12D 20D6                    <1>       and    dh, dl
29909 0000A12F 38F0                    <1>       cmp    al, dh
29910 0000A131 759D                    <1>       jne    short loc_check_ffde_0_next
29911 0000A133 20D4                    <1>       and    ah, dl
29912 0000A135 7599                    <1>       jnz    short loc_check_ffde_0_next
29913 0000A137 30C9                    <1>       xor    cl, cl
29914 0000A139 EB84                    <1>         jmp     loc_check_ffde_retn_2
29915                                  <1>
29916                                  <1> convert_file_name:
29917                                  <1>       ; 06/03/2016
29918                                  <1>       ; 11/02/2016
29919                                  <1>       ; 07/02/2016 (TRDOS 386 = TRDOS v2.0)
29920                                  <1>       ; 06/10/2009
29921                                  <1>       ; 2005
29922                                  <1>       ;
29923                                  <1>       ; INPUT  ->
29924                                  <1>       ;      ESI = Dot File Name Location
29925                                  <1>       ;      EDI = Dir Entry Format File Name Location
29926                                  <1>       ; OUTPUT ->
29927                                  <1>       ;      EDI = Dir Entry Format File Name Location
29928                                  <1>       ;      ESI = Dot File Name Location (capitalized)
29929                                  <1>       ;
29930                                  <1>       ; (ECX, AL will be changed)
29931                                  <1>
29932 0000A13B 56                      <1>       push   esi
29933 0000A13C 57                      <1>       push   edi
29934                                  <1>
29935 0000A13D B90B000000              <1>       mov    ecx, 11
29936 0000A142 B020                    <1>       mov    al, 20h
29937 0000A144 F3AA                    <1>       rep    stosb
29938                                  <1>
29939 0000A146 8B3C24                  <1>       mov    edi, [esp]
29940                                  <1>
29941 0000A149 B10C                    <1>       mov    cl, 12 ; file name length (max.)
29942                                  <1>       ; 06/03/2016
29943                                  <1>       ; Directory entry name limit (11 bytes) check for
29944                                  <1>       ; 'rename_directory_entry' procedure.
29945                                  <1>       ; (EDI points to Directory Entry)
29946                                  <1>       ; (If the file name would not contain a dot
29947                                  <1>       ; and file name length would be 12, this would cause to
29948                                  <1>       ; overwrite the attributes byte of the directory entry.)
29949                                  <1>       ;
29950 0000A14B B50B                    <1>       mov    ch, 11 ; directory entry's name length
29951                                  <1> loc_check_first_dot:
29952 0000A14D 8A06                    <1>       mov    al, [esi]
29953 0000A14F 3C2E                    <1>       cmp    al, 2Eh
29954 0000A151 750C                    <1>       jne    short pass_check_first_dot
29955 0000A153 8807                    <1>       mov    [edi], al
29956 0000A155 47                      <1>       inc    edi
29957 0000A156 46                      <1>       inc    esi
29958 0000A157 FEC9                    <1>       dec    cl
29959 0000A159 75F2                    <1>       jnz    short loc_check_first_dot
29960                                  <1>       ;;(ecx <= 12)
29961                                  <1>       ;;loop loc_check_first_dot
29962 0000A15B EB30                    <1>       jmp    short stop_convert_file
29963                                  <1>
29964                                  <1> loc_get_fchar:
29965 0000A15D 8A06                    <1>       mov    al, [esi]
29966                                  <1> pass_check_first_dot:
29967 0000A15F 3C61                    <1>       cmp    al, 61h ; 'a'
29968 0000A161 7208                    <1>       jb     short pass_name_capitalize
29969 0000A163 3C7A                    <1>       cmp    al, 7Ah ; 'z'
29970 0000A165 7704                    <1>       ja     short pass_name_capitalize
29971 0000A167 24DF                    <1>       and    al, 0DFh
29972 0000A169 8806                    <1>       mov    [esi], al
29973                                  <1> pass_name_capitalize:
29974 0000A16B 3C21                    <1>       cmp    al, 21h
29975 0000A16D 721E                    <1>       jb     short stop_convert_file
29976 0000A16F 3C2E                    <1>       cmp    al, 2Eh ; '.'
29977 0000A171 750C                    <1>       jne    short pass_dot_space
29978                                  <1> add_dot_space:
29979 0000A173 80F904                  <1>       cmp    cl, 4
29980 0000A176 760E                    <1>       jna    short inc_and_loop
29981 0000A178 47                      <1>       inc    edi
29982 0000A179 FECD                    <1>       dec    ch ; 06/03/2016
```

```
29983 0000A17B FEC9              <1>       dec    cl
29984 0000A17D EBF4              <1>       jmp    short add_dot_space
29985                            <1>
29986                            <1>       ;mov   al, 4
29987                            <1>       ;cmp   cl, al
29988                            <1>       ;jna   short inc_and_loop
29989                            <1>       ;sub   cl, al
29990                            <1>       ;add   edi, ecx
29991                            <1>       ;mov   cl, al
29992                            <1>       ;jmp   short inc_and_loop
29993                            <1>
29994                            <1> pass_dot_space:
29995 0000A17F 8807              <1>       mov    [edi], al
29996                            <1> loc_after_double_dot:
29997                            <1>       ; 06/03/2016
29998 0000A181 FECD              <1>       dec    ch ; count down for 11 bytes dir entry limit
29999 0000A183 740A              <1>       jz     short stop_convert_file_x
30000 0000A185 47                <1>       inc    edi
30001                            <1> inc_and_loop:
30002 0000A186 FEC9              <1>       dec    cl ; count down for 12 bytes filename limit
30003 0000A188 7403              <1>       jz     short stop_convert_file
30004 0000A18A 46                <1>       inc    esi
30005                            <1>       ;;(ecx <= 12)
30006                            <1>       ;;loop loc_get_fchar
30007 0000A18B EBD0              <1>       jmp    short loc_get_fchar
30008                            <1>
30009                            <1> stop_convert_file:
30010                            <1>       ; 06/03/2016
30011 0000A18D 30ED              <1>       xor    ch, ch
30012                            <1>       ; ECX < 256 ; 'find_first_file' -> xor cl, cl
30013                            <1> stop_convert_file_x:
30014 0000A18F 5F                <1>       pop    edi
30015 0000A190 5E                <1>       pop    esi
30016 0000A191 C3                <1>       retn
30017                            <1>
30018                            <1> save_longname_sub_component:
30019                            <1>       ; 13/02/2016
30020                            <1>       ; 06/02/2016 (TRDOS 386 = TRDOS v2.0)
30021                            <1>       ; 28/02/2010
30022                            <1>       ; 17/10/2009
30023                            <1>       ; INPUT ->
30024                            <1>       ;      EDI = Directory Entry
30025                            <1>       ;      // This procedure is called
30026                            <1>       ;      // from 'find_directory_entry' procedure.
30027                            <1>       ;      // If the last entry returns with
30028                            <1>       ;      // a non-zero LongnameFound value and
30029                            <1>       ;      // if LFN_CheckSum value is equal to
30030                            <1>       ;      // the next shortname checksum,
30031                            <1>       ;      // long name is valid.
30032                            <1>       ;      // If a longname is longer than 65 bytes,
30033                            <1>       ;      // it is invalid for trdos. (>45h)
30034                            <1>
30035 0000A192 57                <1>       push   edi
30036 0000A193 56                <1>       push   esi
30037                            <1>       ;push  ebx
30038                            <1>       ;push  ecx
30039                            <1>       ;push  edx
30040 0000A194 50                <1>       push   eax
30041                            <1>
30042 0000A195 29C9              <1>       sub    ecx, ecx
30043                            <1>       ;sub   eax, eax
30044 0000A197 B11A              <1>       mov    cl, 26
30045                            <1>
30046 0000A199 0FB607            <1>       movzx  eax, byte [edi] ; LDIR_Order
30047 0000A19C 3C41              <1>       cmp    al, 41h  ; 40h (last long entry sign) + 1
30048 0000A19E 722B              <1>       jb     short pass_pslnsc_last_long_entry
30049                            <1>
30050 0000A1A0 88C4              <1>       mov    ah, al
30051 0000A1A2 80EC40            <1>       sub    ah, 40h
30052 0000A1A5 8825[8E570100]    <1>       mov    [LFN_EntryLength], ah
30053                            <1>
30054 0000A1AB 3C45              <1>       cmp    al, 45h  ; 40h (last long entry sign) + 5
30055                            <1>              ; Max 130 byte length is usable in TRDOS
30056                            <1> ; 26*5 = 130
30057 0000A1AD 7753              <1>       ja     short loc_pslnsc_retn
30058                            <1>
30059 0000A1AF 2407              <1>       and    al, 07h ; 0Fh
30060 0000A1B1 A2[8D570100]      <1>       mov    [LongNameFound], al
30061                            <1>
30062 0000A1B6 FEC8              <1>       dec    al
30063                            <1>       ;mov   cl, 26
30064 0000A1B8 F6E1              <1>       mul    cl
30065                            <1>
30066 0000A1BA 89C6              <1>       mov    esi, eax
30067 0000A1BC 01CE              <1>       add    esi, ecx
30068                            <1>              ; to make is an ASCIZZ string
30069                            <1>              ; with ax+26 bytes length
30070 0000A1BE 81C6[90570100]    <1>       add    esi, LongFileName
30071 0000A1C4 66C7060000        <1>       mov    word [esi], 0
30072 0000A1C9 EB16              <1>       jmp    short loc_pslsc_move_ldir_name2
30073                            <1>
30074                            <1> pass_pslnsc_last_long_entry:
30075 0000A1CB 3C04              <1>       cmp    al, 04h
30076 0000A1CD 7733              <1>       ja     short loc_pslnsc_retn
30077 0000A1CF FE0D[8D570100]    <1>       dec    byte [LongNameFound]
30078 0000A1D5 3A05[8D570100]    <1>       cmp    al, [LongNameFound]
30079 0000A1DB 7525              <1>       jne    short loc_pslnsc_retn
30080                            <1>
30081                            <1> loc_pslsc_move_ldir_name1:
30082 0000A1DD FEC8              <1>       dec    al
30083                            <1>       ;mov   cl, 26
30084 0000A1DF F6E1              <1>       mul    cl
```

```
30085                                  <1>
30086                                  <1> loc_pslsc_move_ldir_name2:
30087 0000A1E1 8A4F0D                  <1>       mov   cl, [edi+0Dh] ; long name checksum
30088 0000A1E4 880D[8F570100]          <1>       mov   [LFN_CheckSum], cl
30089 0000A1EA 89FE                    <1>       mov   esi, edi ; LDIR_Order
30090 0000A1EC BF[90570100]            <1>       mov   edi, LongFileName
30091 0000A1F1 01C7                    <1>       add   edi, eax
30092 0000A1F3 46                      <1>       inc   esi
30093 0000A1F4 B105                    <1>       mov   cl, 5 ; chars 1 to 5
30094 0000A1F6 F366A5                  <1>       rep   movsw
30095 0000A1F9 83C603                  <1>       add   esi, 3
30096 0000A1FC A5                      <1>       movsd ; char 6 & 7
30097 0000A1FD A5                      <1>       movsd ; char 8 & 9
30098 0000A1FE A5                      <1>       movsd ; char 10 & 11
30099 0000A1FF 46                      <1>       inc   esi
30100 0000A200 46                      <1>       inc   esi
30101 0000A201 A5                      <1>       movsd   ; char 12 & 13
30102                                  <1>
30103                                  <1> loc_pslnsc_retn:
30104 0000A202 58                      <1>       pop   eax
30105                                  <1>       ;pop  edx
30106                                  <1>       ;pop  ecx
30107                                  <1>       ;pop  ebx
30108 0000A203 5E                      <1>       pop   esi
30109 0000A204 5F                      <1>       pop   edi
30110                                  <1>
30111 0000A205 C3                      <1>       retn
30112                                  <1>
30113                                  <1> parse_path_name:
30114                                  <1>       ; 10/02/2016
30115                                  <1>       ; 08/02/2016 (TRDOS 386 = TRDOS v2.0)
30116                                  <1>       ; 10/009/2011 ('proc_parse_pathname')
30117                                  <1>       ; 27/11/2009
30118                                  <1>       ; 05/12/2004
30119                                  <1>       ;
30120                                  <1>       ; INPUT ->
30121                                  <1>       ;     ESI = Beginning of ASCIIZ pathname string
30122                                  <1>       ;      EDI = Destination Address
30123                                  <1>       ;           (which is TR-DOS FindFile data buffer)
30124                                  <1>       ; OUTPUT ->
30125                                  <1>       ;     CF = 1 -> Error
30126                                  <1>       ;          EAX = Error Code (AL)
30127                                  <1>       ;
30128                                  <1>       ; (Modified registers: eax, ecx, esi, edi)
30129                                  <1>
30130                                  <1>       ; Clear the pathname bytes in TR-DOS Findfile data buffer
30131 0000A206 57                      <1>       push  edi
30132 0000A207 B914000000              <1>       mov   ecx, 20  ; 80 bytes
30133 0000A20C 31C0                    <1>       xor   eax, eax
30134 0000A20E F3AB                    <1>       rep   stosd
30135 0000A210 5F                      <1>       pop   edi
30136                                  <1>
30137 0000A211 668B06                  <1>       mov   ax, [esi]
30138 0000A214 80FC3A                  <1>       cmp   ah, ':'
30139 0000A217 741C                    <1>       je    short loc_ppn_change_drive
30140 0000A219 A0[8E4E0100]            <1>       mov   al, [Current_Drv]
30141 0000A21E EB33                    <1>       jmp   short pass_ppn_change_drive
30142                                  <1>
30143                                  <1> pass_ppn_cdir:
30144 0000A220 8B35[B2580100]          <1>       mov   esi, [First_Path_Pos]
30145 0000A226 AC                      <1>       lodsb
30146                                  <1> loc_ppn_get_filename:
30147 0000A227 83C741                  <1>       add   edi, 65 ; FindFile_Name location
30148                                  <1>       ; TRDOS Filename length must not be more than 12 bytes
30149                                  <1>       ;mov  ecx, 12
30150 0000A22A B10C                    <1>       mov   cl, 12
30151                                  <1> loc_ppn_get_fnchar_next:
30152 0000A22C AA                      <1>       stosb
30153 0000A22D AC                      <1>       lodsb
30154 0000A22E 3C21                    <1>       cmp   al, 21h
30155 0000A230 7274                    <1>       jb    short loc_ppn_clc_return
30156 0000A232 E2F8                    <1>         loop    loc_ppn_get_fnchar_next
30157                                  <1> loc_ppn_return:
30158 0000A234 C3                      <1>       retn
30159                                  <1>
30160                                  <1> loc_ppn_change_drive:
30161 0000A235 24DF                    <1>       and   al, 0DFh
30162 0000A237 2C41                    <1>       sub   al, 'A'; A:
30163 0000A239 726F                    <1>       jc    short loc_ppn_invalid_drive
30164 0000A23B 3805[97020100]          <1>       cmp   [Last_DOS_DiskNo], al
30165 0000A241 7267                    <1>       jb    short loc_ppn_invalid_drive
30166                                  <1>
30167 0000A243 46                      <1>       inc   esi
30168 0000A244 46                      <1>       inc   esi
30169 0000A245 8A26                    <1>       mov   ah, [esi]
30170 0000A247 80FC21                  <1>       cmp   ah, 21h
30171 0000A24A 7307                    <1>       jnb   short pass_ppn_change_drive
30172                                  <1>
30173                                  <1> loc_ppn_cmd_failed:
30174                                  <1>       ; File or directory name is not existing
30175 0000A24C 8807                    <1>       mov   [edi], al ; Drv
30176 0000A24E 66B80100                <1>       mov   ax, 1 ; eax = 1
30177                                  <1>       ; TR-DOS Error Code 01h = Bad Command Argument
30178                                  <1>       ; MS-DOS Error Code 01h : Invalid Function Number
30179                                  <1>       ;stc
30180                                  <1>       ; (MainProg ErrMsg: "Bad command or file name!")
30181 0000A252 C3                      <1>       retn
30182                                  <1>
30183                                  <1> pass_ppn_change_drive:
30184 0000A253 8935[B2580100]          <1>       mov   [First_Path_Pos], esi
30185 0000A259 C705[B6580100]0000-     <1>       mov   dword [Last_Slash_Pos], 0
30186 0000A261 0000                    <1>
```

```
30187 0000A263 AA                <1>       stosb
30188 0000A264 8A06              <1>       mov    al, [esi]
30189                            <1> loc_scan_ppn_dslash:
30190 0000A266 3C2F              <1>       cmp    al, '/'
30191 0000A268 7506              <1>       jne    short loc_scan_next_slash_pos
30192 0000A26A 8935[B6580100]    <1>       mov    [Last_Slash_Pos], esi
30193                            <1> loc_scan_next_slash_pos:
30194 0000A270 46                <1>       inc    esi
30195 0000A271 8A06              <1>       mov    al, [esi]
30196 0000A273 3C20              <1>       cmp    al, 20h
30197 0000A275 77EF              <1>       ja     short loc_scan_ppn_dslash
30198 0000A277 833D[B6580100]00  <1>       cmp    dword [Last_Slash_Pos], 0
30199 0000A27E 76A0              <1>       jna    short pass_ppn_cdir
30200                            <1>
30201 0000A280 8B0D[B6580100]    <1>       mov    ecx, [Last_Slash_Pos]
30202 0000A286 8B35[B2580100]    <1>       mov    esi, [First_Path_Pos]
30203 0000A28C 29F1              <1>       sub    ecx, esi
30204 0000A28E 41                <1>       inc    ecx
30205                            <1>       ;cmp   ecx, 64
30206 0000A28F 80F940            <1>       cmp    cl, 64
30207 0000A292 7715              <1>       ja     short loc_ppn_invalid_drive_stc
30208                            <1>
30209 0000A294 89F8              <1>       mov    eax, edi ; Dest Dir String Location (65 byte)
30210 0000A296 F3A4              <1>       rep    movsb
30211                            <1>       ;mov   [edi], cl ; 0, End of Dir String
30212 0000A298 8B35[B6580100]    <1>       mov    esi, [Last_Slash_Pos]
30213 0000A29E 46                <1>       inc    esi
30214 0000A29F 89C7              <1>       mov    edi, eax
30215 0000A2A1 AC                <1>       lodsb
30216 0000A2A2 3C21              <1>       cmp    al, 21h
30217 0000A2A4 7381              <1>       jnb    short loc_ppn_get_filename
30218                            <1> loc_ppn_clc_return:
30219                            <1>       ;clc
30220 0000A2A6 31C0              <1>       xor    eax, eax
30221 0000A2A8 C3                <1>       retn
30222                            <1>
30223                            <1> loc_ppn_invalid_drive_stc:
30224 0000A2A9 F5                <1>       cmc    ; stc
30225                            <1> loc_ppn_invalid_drive:
30226                            <1>       ; cf = 1
30227                            <1>       ; The Drive Letter/Char < "A" or > "Z"
30228 0000A2AA 66B80F00          <1>       mov    ax, 0Fh
30229                            <1>       ; MS-DOS Error Code 0Fh = Disk Drive Invalid
30230                            <1>       ; (MainProg ErrMsg: "Drive not ready or read error!")
30231 0000A2AE C3                <1>       retn
30232                            <1>
30233                            <1> find_longname:
30234                            <1>       ; 13/02/2016 (TRDOS 386 = TRDOS v2.0)
30235                            <1>       ; 24/01/2010 (DIR.ASM, 'proc_find_longname')
30236                            <1>       ; 17/10/2009
30237                            <1>
30238                            <1>       ; INPUT ->
30239                            <1>       ;     ESI = DOS short file name address
30240                            <1>       ;     for example: "filename.ext"
30241                            <1>       ;
30242                            <1>       ; OUTPUT ->
30243                            <1>       ;     ESI = ASCIIZ longname address (cf = 0)
30244                            <1>       ;     cf = 1 -> error number returns in EAX (AL)
30245                            <1>       ;     AL = 0 & CF=1 -> longname not found
30246                            <1>       ;          the file/directory has no longname
30247                            <1>       ;     cf = 0 -> AL = FAT Type
30248                            <1>
30249                            <1>       ; 17/10/2009
30250                            <1>       ; ASCIIZ string will be returned
30251                            <1>       ; as LongFileName
30252                            <1>       ; clearing/reset is not needed
30253                            <1>       ;mov   ecx, 33
30254                            <1>       ;mov   edi, LongFileName
30255                            <1>       ;sub   ax, ax ; 0
30256                            <1>       ;rep   stosw
30257                            <1>
30258                            <1>       ;mov   byte [LongNameFound], 0
30259                            <1>
30260                            <1>       ; ESI = ASCIIZ file/directory name address
30261                            <1>       ;   AL = Attributes AND mask
30262                            <1>       ;      (Result of AND must be equal to AL)
30263                            <1>       ;   AH = Negative attributes mask
30264                            <1>       ;      (Result of AND must be ZERO)
30265 0000A2AF 66B80008          <1>       mov    ax, 0800h
30266                            <1>              ; it must not be volume name or longname
30267 0000A2B3 E837DDFFFF        <1>       call   find_first_file
30268 0000A2B8 7216              <1>       jc     short loc_fln_retn
30269                            <1>
30270                            <1> loc_fln_check_FAT_Type:
30271 0000A2BA 803D[8D4E0100]01  <1>       cmp    byte [Current_FATType], 1
30272 0000A2C1 7306              <1>       jnb    short loc_fln_check_longname_yes_sign
30273                            <1>
30274 0000A2C3 E839000000        <1>       call   get_fs_longname
30275 0000A2C8 C3                <1>       retn
30276                            <1>
30277                            <1> loc_fln_check_longname_yes_sign:
30278 0000A2C9 08FF              <1>       or     bh, bh
30279 0000A2CB 7504              <1>       jnz    short loc_fln_check_longnamefound_number
30280                            <1> loc_fln_longname_not_found_retn:
30281 0000A2CD 31C0              <1>       xor    eax, eax
30282                            <1>       ; cf = 1 & al = 0 -> longname not found
30283 0000A2CF F9                <1>       stc
30284                            <1> loc_fln_retn:
30285 0000A2D0 C3                <1>       retn
30286                            <1>
30287                            <1> loc_fln_check_longnamefound_number:
30288                            <1>       ; 'LongNameFound' is set by
```

```
30289                                  <1>         ; by 'save_longname_sub_component'
30290                                  <1>       ; which is called from
30291                                  <1>       ; 'find_directory_entry'
30292                                  <1>       ; which is called from
30293                                  <1>       ; 'find_first_file'
30294                                  <1>       ; It must 1 if the longname is valid
30295 0000A2D1 803D[8D570100]01       <1>         cmp     byte [LongNameFound], 1
30296 0000A2D8 75F3                   <1>       jne    short loc_fln_longname_not_found_retn
30297                                  <1>
30298                                  <1> loc_fln_calculate_checksum:
30299 0000A2DA E813000000             <1>       call   calculate_checksum
30300                                  <1>       ; AL = shortname checksum
30301                                  <1>
30302                                  <1> loc_fln_longname_validation:
30303                                  <1>       ; 'LFN_CheckSum' has been set already
30304                                  <1>       ; by 'save_longname_sub_component'
30305                                  <1>       ; which is called from
30306                                  <1>       ; 'find_directory_entry'
30307                                  <1>       ; which is called from
30308                                  <1>       ; 'find_first_file'
30309 0000A2DF 3805[8F570100]         <1>       cmp    [LFN_CheckSum], al
30310 0000A2E5 75E6                   <1>       jne    short loc_fln_longname_not_found_retn
30311                                  <1>
30312 0000A2E7 BE[90570100]           <1>       mov    esi, LongFileName
30313 0000A2EC A0[8D4E0100]           <1>       mov    al, [Current_FATType]
30314 0000A2F1 C3                     <1>       retn
30315                                  <1>
30316                                  <1> calculate_checksum:
30317                                  <1>       ; 13/02/2016 (TRDOS 386 = TRDOS v2.0)
30318                                  <1>       ; 17/10/2009 (DIR.ASM, 'proc_calculate_checksum')
30319                                  <1>         ;
30320                                  <1>       ; INPUT ->
30321                                  <1>       ;      ESI = 11 byte DOS File Name location
30322                                  <1>       ;      (in DOS Directory Entry Format)
30323                                  <1>       ; OUTPUT ->
30324                                  <1>       ;       AL = 8 bit checksum (CRC) value
30325                                  <1>       ;
30326                                  <1>       ; (Modified registers: EAX, ECX, ESI)
30327                                  <1>
30328                                  <1>       ; Erdogan Tan [ 17-10-2009 ]
30329                                  <1>       ;  'ror al, 1' instruction
30330                                  <1>
30331                                  <1>       ; Erdogan Tan [ 20-06-2004 ]
30332                                  <1>       ; This 8086 assembly code is an original code
30333                                  <1>       ; which is adapted from C code in
30334                                  <1>       ; Microsoft FAT32 File System Specification
30335                                  <1>       ; Version 1.03, December 6, 2000
30336                                  <1>       ; Page 28
30337                                  <1>
30338 0000A2F2 30C0                   <1>       xor    al, al
30339 0000A2F4 B90B000000             <1>       mov    ecx, 11
30340                                  <1> loc_next_sum:
30341                                  <1>       ;xor   ah, ah
30342                                  <1>       ;test  al, 1
30343                                  <1>       ;jz    short pass_ah_80h
30344                                  <1>       ;mov   ah, 80h
30345                                  <1> ;pass_ah_80h:
30346                                  <1>       ;shr   al, 1
30347 0000A2F9 D0C8                   <1>       ror    al, 1 ; 17/10/2009
30348 0000A2FB 0206                   <1>       add    al, [esi]
30349 0000A2FD 46                     <1>       inc    esi
30350                                  <1>       ;add   al, ah
30351 0000A2FE E2F9                   <1>       loop   loc_next_sum
30352 0000A300 C3                     <1>       retn
30353                                  <1>
30354                                  <1> get_fs_longname:
30355                                  <1>       ; temporary (13/02/2016)
30356 0000A301 31C0                   <1>       xor eax, eax
30357 0000A303 F9                     <1>       stc
30358 0000A304 C3                     <1>       retn
30359                                  <1>
30360                                  <1> make_sub_directory:
30361                                  <1>       ; 16/10/2016
30362                                  <1>       ; 02/03/2016, 03/03/2016
30363                                  <1>       ; 26/02/2016, 27/02/2016
30364                                  <1>       ; 21/02/2016 (TRDOS 386 = TRDOS v2.0)
30365                                  <1>       ; 01/08/2011 (DIR.ASM, 'proc_make_directory')
30366                                  <1>       ; 10/07/2010
30367                                  <1>       ; INPUT ->
30368                                  <1>       ;      ESI = ASCIIZ Directory Name
30369                                  <1>       ;      CL = Directory Attributes
30370                                  <1>       ; OUTPUT ->
30371                                  <1>       ;      EAX = New sub dir's first cluster
30372                                  <1>       ;      ESI = Logical Dos Drv Descr. Table Addr.
30373                                  <1>       ;      CF = 1 -> error code in AL (EAX)
30374                                  <1>
30375                                  <1>       ;test  cl, 10h  ; directory
30376                                  <1>       ;jz    short loc_make_directory_access_denied
30377                                  <1>       ;test  cl, 08h ; volume name
30378                                  <1>       ;jnz   short loc_make_directory_access_denied
30379                                  <1>
30380 0000A305 80E107                 <1>       and    cl, 07h
30381 0000A308 880D[0C590100]         <1>       mov    byte [mkdir_attrib], cl
30382                                  <1>
30383 0000A30E 56                     <1>       push   esi
30384 0000A30F 31DB                   <1>       xor    ebx, ebx
30385 0000A311 8A3D[8E4E0100]         <1>       mov    bh, [Current_Drv]
30386 0000A317 BE00010900             <1>       mov    esi, Logical_DOSDisks
30387 0000A31C 01DE                   <1>       add    esi, ebx
30388 0000A31E 5B                     <1>       pop    ebx
30389                                  <1>
30390                                  <1>       ; 10/07/2010 -> 1st writable disk check for trdos
```

```
30391                                  <1>       ; LD_DiskType = 0 for write protection (read only)
30392 0000A31F 807E0101                <1>       cmp    byte [esi+LD_DiskType], 1 ; 0 = Invalid
30393 0000A323 730B                    <1>       jnb    short loc_mkdir_check_file_sytem
30394                                  <1>       ; 16/10/2016 (13h -> 30)
30395 0000A325 B81E000000              <1>       mov    eax, 30 ; 'Disk write-protected' error
30396 0000A32A BA00000000              <1>       mov    edx, 0
30397                                  <1>       ; err retn: EDX = 0, EBX = Dir name offset
30398                                  <1>       ;ESI = Logical DOS drive description table address
30399 0000A32F C3                      <1>       retn
30400                                  <1>
30401                                  <1> ;loc_make_directory_access_denied:
30402                                  <1>       ;mov   ax, 05h ; access denied (invalid attributes input)
30403                                  <1>       ;stc
30404                                  <1>       ;retn
30405                                  <1>
30406                                  <1> loc_mkdir_check_file_sytem:
30407 0000A330 807E0301                <1>       cmp    byte [esi+LD_FATType], 1
30408 0000A334 730B                    <1>       jnb    short loc_mkdir_check_free_sectors
30409                                  <1>
30410                                  <1> loc_make_fs_directory:
30411 0000A336 A1[884E0100]            <1>       mov    eax, [Current_Dir_FCluster]
30412                                  <1>       ; EAX = Parent directory DDT Address
30413                                  <1>       ; ESI = Logical DOS Drive DT Address
30414                                  <1>       ; EBX = Directory name offset (as ASCIIZ name)
30415 0000A33B E8D8150000              <1>       call   make_fs_directory
30416 0000A340 C3                      <1>       retn
30417                                  <1>
30418                                  <1> loc_mkdir_check_free_sectors:
30419 0000A341 0FB64613                <1>         movzx   eax, byte [esi+LD_BPB+SecPerClust]
30420 0000A345 8B4E74                  <1>       mov    ecx, [esi+LD_FreeSectors]
30421 0000A348 39C1                    <1>       cmp    ecx, eax
30422 0000A34A 7255                    <1>       jb     short loc_mkdir_insufficient_disk_space
30423                                  <1>
30424                                  <1> loc_make_fat_directory:
30425 0000A34C 891D[FC580100]          <1>       mov    [mkdir_DirName_Offset], ebx
30426 0000A352 890D[08590100]          <1>       mov    [mkdir_FreeSectors], ecx
30427                                  <1>
30428                                  <1>       ;mov   al, [esi+LD_BPB+SecPerClust]
30429 0000A358 A2[0E590100]            <1>       mov    byte [mkdir_SecPerClust], al
30430                                  <1>
30431                                  <1> loc_mkdir_gffc_1:
30432 0000A35D E812180000              <1>       call   get_first_free_cluster
30433 0000A362 722A                    <1>       jc     short loc_mkdir_gffc_retn
30434                                  <1>
30435                                  <1> ;loc_mkdir_gffc_1_cont:
30436                                  <1>       ;cmp   eax, 2
30437                                  <1>       ;jb    short loc_mkdir_gffc_insufficient_disk_space
30438                                  <1>
30439                                  <1> ;loc_mkdir_gffc_1_save_fcluster:
30440 0000A364 A3[00590100]            <1>       mov    [mkdir_FFCluster], eax
30441                                  <1>
30442                                  <1> loc_mkdir_locate_ffe:
30443                                  <1>       ; Current directory fcluster <> Directory buffer cluster
30444                                  <1>       ; Current directory will be reloaded by
30445                                  <1>       ; 'locate_current_dir_file' procedure
30446                                  <1>       ;
30447                                  <1>       ; ESI = Logical DOS Drive Description Table Address
30448                                  <1>       ;push  esi ; 27/02/2016
30449 0000A369 31C0                    <1>       xor    eax, eax
30450 0000A36B 89C1                    <1>         mov  ecx, eax
30451 0000A36D 6649                    <1>       dec    cx ; FFFFh
30452                                  <1>       ; CX = FFFFh -> find first deleted or free entry
30453                                  <1>       ; ESI would be ASCIIZ filename address if the call
30454                                  <1>       ; would not be for first free or deleted dir entry
30455 0000A36F E8D2FAFFFF              <1>       call   locate_current_dir_file
30456 0000A374 734C                    <1>       jnc    short loc_mkdir_set_ff_dir_entry_1
30457                                  <1>       ;pop   esi
30458                                  <1>       ; ESI = Logical DOS Drive Description Table Address
30459 0000A376 83F802                  <1>       cmp    eax, 2  ; cmp al, 2 ; File/Dir not found !
30460 0000A379 752B                    <1>       jne    short loc_mkdir_stc_return
30461                                  <1>
30462                                  <1> loc_mkdir_add_new_cluster:
30463 0000A37B 3805[8D4E0100]          <1>       cmp    byte [Current_FATType], al ; 2
30464                                  <1>       ;cmp   byte ptr [esi+LD_FATType], 2
30465 0000A381 770C                    <1>       ja     short loc_mkdir_add_new_cluster_check_fsc
30466 0000A383 803D[8C4E0100]01        <1>       cmp    byte [Current_Dir_Level], 1
30467                                  <1>       ;cmp   byte [esi+LD_CDirLevel], 1
30468 0000A38A 7303                    <1>       jnb    short loc_mkdir_add_new_cluster_check_fsc
30469                                  <1>
30470 0000A38C B00C                    <1>       mov    al, 12 ; No more files
30471                                  <1> loc_mkdir_gffc_retn:
30472 0000A38E C3                      <1>       retn
30473                                  <1>
30474                                  <1> loc_mkdir_add_new_cluster_check_fsc:
30475 0000A38F 8B0D[08590100]          <1>       mov    ecx, [mkdir_FreeSectors]
30476                                  <1>       ;movzx eax, byte [mkdir_SecPerClust]
30477 0000A395 A0[0E590100]            <1>       mov    al, [mkdir_SecPerClust]
30478 0000A39A 66D1E0                  <1>       shl    ax, 1 ; AX = 2 * AX
30479 0000A39D 39C1                    <1>       cmp    ecx, eax
30480 0000A39F 7350                    <1>       jnb    short loc_mkdir_add_new_subdir_cluster
30481                                  <1>
30482                                  <1> loc_mkdir_insufficient_disk_space:
30483                                  <1>       ;mov   edx, ecx
30484                                  <1> ;loc_mkdir_gffc_insufficient_disk_space:
30485 0000A3A1 66B82700                <1>       mov    ax, 27h ; MSDOS err => insufficient disk space
30486                                  <1>       ; err retn: EDX = Free sectors, EBX = Dir name offset
30487                                  <1>         ; ESI -> Dos drive description table address
30488                                  <1>       ;; ecx = edx
30489                                  <1>       ;
30490 0000A3A5 C3                      <1>       retn
30491                                  <1>
30492                                  <1> loc_mkdir_stc_return:
```

```
30493 0000A3A6 F9                <1>       stc
30494 0000A3A7 C3                <1>       retn
30495                            <1>
30496                            <1> loc_mkdir_gffc_2:
30497 0000A3A8 E8C7170000        <1>       call   get_first_free_cluster
30498 0000A3AD 72DF              <1>       jc     short loc_mkdir_gffc_retn
30499                            <1>
30500                            <1> ;loc_mkdir_gffc_1_cont:
30501                            <1>       ;cmp   eax, 2
30502                            <1>       ;jb    short loc_mkdir_gffc_insufficient_disk_space
30503                            <1>
30504                            <1> ;loc_mkdir_gffc_2_save_fcluster:
30505 0000A3AF A3[00590100]      <1>       mov    [mkdir_FFCluster], eax
30506                            <1>
30507 0000A3B4 A1[04590100]      <1>       mov    eax, [mkdir_LastDirCluster]
30508                            <1>
30509 0000A3B9 E845170000        <1>       call   load_FAT_sub_directory
30510 0000A3BE 72CE              <1>       jc     short loc_mkdir_gffc_retn
30511                            <1>
30512 0000A3C0 31FF              <1>       xor    edi, edi
30513                            <1> loc_mkdir_set_ff_dir_entry_1:
30514                            <1>       ; 27/02/2016
30515 0000A3C2 56                <1>       push   esi ; Logical DOS Drv Desc. Tbl. address
30516                            <1>       ; EDI = Directory Entry Address
30517 0000A3C3 8B35[FC580100]    <1>       mov    esi, [mkdir_DirName_Offset]
30518 0000A3C9 A1[00590100]      <1>       mov    eax, [mkdir_FFCluster]
30519                            <1>
30520 0000A3CE 66B91000          <1>       mov    cx, 10h      ; CL = Directory attribute
30521                            <1>                          ; CH = 0 -> File size is 0
30522 0000A3D2 0A0D[0C590100]    <1>       or     cl, [mkdir_attrib] ; S, H, R
30523 0000A3D8 E8B0010000        <1>       call   make_directory_entry
30524                            <1>
30525 0000A3DD 5E                <1>       pop    esi
30526                            <1>
30527 0000A3DE C605[B8560100]02  <1>       mov    byte [DirBuff_ValidData], 2
30528 0000A3E5 E880020000        <1>       call   save_directory_buffer
30529 0000A3EA 0F83DA000000      <1>         jnc     loc_mkdir_set_ff_dir_entry_2
30530                            <1>
30531                            <1> loc_mkdir_return:
30532 0000A3F0 C3                <1>       retn
30533                            <1>
30534                            <1> loc_mkdir_add_new_subdir_cluster:
30535 0000A3F1 8B15[BD560100]    <1>       mov    edx, [DirBuff_Cluster]
30536 0000A3F7 8915[04590100]    <1>       mov    [mkdir_LastDirCluster], edx
30537                            <1>
30538 0000A3FD A1[00590100]      <1>       mov    eax, [mkdir_FFCluster]
30539 0000A402 E8FC160000        <1>       call   load_FAT_sub_directory
30540 0000A407 72E7              <1>       jc     short loc_mkdir_return
30541                            <1>       ; eax = 0
30542                            <1>       ; ecx =  directory buffer sector count (<= 128)
30543                            <1>
30544                            <1> pass_mkdir_add_new_subdir_cluster:
30545 0000A409 29FF              <1>       sub    edi, edi ; 0
30546                            <1>       ;mov   al, 128 ; double word
30547                            <1>       ;mul   ecx ; ecx =  directory buffer sector count
30548                            <1>       ;mov   ecx, eax
30549                            <1>       ;shl   cx, 7 ; 128 * sector count
30550 0000A40B 668B4611          <1>       mov    ax, [esi+LD_BPB+BytesPerSec] ; 512
30551 0000A40F 66C1E802          <1>       shr    ax, 2 ; 'byte count / 4' for 'stosd'
30552 0000A413 66F7E1            <1>       mul    cx ; max = 128*(512/4) -> 16384 (stosd)
30553 0000A416 6689C1            <1>       mov    cx, ax
30554 0000A419 6629C0            <1>       sub    ax, ax ; 0
30555 0000A41C F3AB              <1>       rep    stosd ; clear directory buffer
30556                            <1>
30557 0000A41E C605[B8560100]02  <1>       mov    byte [DirBuff_ValidData], 2
30558 0000A425 E840020000        <1>       call   save_directory_buffer
30559 0000A42A 72C4              <1>       jc     short loc_mkdir_return
30560                            <1>
30561                            <1> loc_mkdir_save_added_cluster:
30562 0000A42C A1[04590100]      <1>       mov    eax, [mkdir_LastDirCluster]
30563 0000A431 8B0D[00590100]    <1>       mov    ecx, [mkdir_FFCluster]
30564                            <1>       ; 01/03/2016
30565 0000A437 31D2              <1>       xor    edx, edx
30566 0000A439 8915[AE560100]    <1>       mov    [FAT_ClusterCounter], edx ; 0 ; reset
30567 0000A43F E803180000        <1>       call   update_cluster
30568 0000A444 7304              <1>       jnc    short loc_mkdir_save_fat_buffer_0
30569 0000A446 09C0              <1>       or     eax, eax ; EAX = 0 -> cluster value is 0 or eocc
30570 0000A448 7518              <1>       jnz    short loc_mkdir_save_fat_buffer_stc_retn
30571                            <1>
30572                            <1> loc_mkdir_save_fat_buffer_0:
30573 0000A44A A1[00590100]      <1>       mov    eax, [mkdir_FFCluster]
30574 0000A44F A3[04590100]      <1>       mov    [mkdir_LastDirCluster], eax
30575                            <1>
30576 0000A454 31C9              <1>       xor    ecx, ecx
30577 0000A456 49                <1>       dec    ecx ; FFFFFFFFh
30578                            <1>       ; ESI = Logical DOS Drive Description Table address
30579 0000A457 E8EB170000        <1>       call   update_cluster
30580 0000A45C 731A              <1>       jnc    short loc_mkdir_save_fat_buffer_1
30581 0000A45E 09C0              <1>       or     eax, eax
30582 0000A460 7416              <1>       jz     short loc_mkdir_save_fat_buffer_1
30583                            <1>
30584                            <1> loc_mkdir_save_fat_buffer_stc_retn:
30585                            <1>       ; 01/03/2016
30586 0000A462 803D[AE560100]01  <1>       cmp    byte [FAT_ClusterCounter], 1
30587 0000A469 720C              <1>       jb     short loc_mkdir_save_fat_buffer_retn
30588                            <1>
30589 0000A46B 66BB00FF          <1>       mov    bx, 0FF00h ; recalculate free space (BL = 0)
30590                            <1>                         ; (BH = FFh -> Use ESI as Drv Param. Tbl.)
30591 0000A46F 50                <1>       push   eax
30592 0000A470 E8241B0000        <1>       call   calculate_fat_freespace
30593 0000A475 58                <1>       pop    eax
30594 0000A476 F9                <1>       stc
```

```
30595                                  <1> loc_mkdir_save_fat_buffer_retn:
30596 0000A477 C3                      <1>       retn
30597                                  <1>
30598                                  <1> loc_mkdir_save_fat_buffer_1:
30599                                  <1>       ; byte [FAT_BuffValidData] = 2
30600 0000A478 E8871A0000              <1>       call   save_fat_buffer
30601 0000A47D 72E3                    <1>       jc     short loc_mkdir_save_fat_buffer_stc_retn
30602                                  <1>
30603                                  <1>       ; 01/03/2016
30604 0000A47F 803D[AE560100]01        <1>       cmp    byte [FAT_ClusterCounter], 1
30605 0000A486 721B                    <1>       jb     short loc_mkdir_save_fat_buffer_2
30606                                  <1>
30607                                  <1>       ; ESI = Logical DOS Drive Description Table address
30608 0000A488 A1[AE560100]            <1>       mov    eax, [FAT_ClusterCounter]
30609 0000A48D 66BB01FF                <1>       mov    bx, 0FF01h ; add free clusters
30610 0000A491 E8031B0000              <1>       call   calculate_fat_freespace
30611                                  <1>
30612                                  <1>       ;inc   eax ; 0FFFFFFFFh -> 0 ; recalculation is needed!
30613                                  <1>       ;jnz   short loc_mkdir_save_fat_buffer_2
30614                                  <1>
30615                                  <1>       ; ecx > 0 -> Recalculation is needed
30616 0000A496 09C9                    <1>       or     ecx, ecx
30617 0000A498 7409                    <1>       jz     short loc_mkdir_save_fat_buffer_2
30618                                  <1>
30619 0000A49A 66BB00FF                <1>       mov    bx, 0FF00h ; ; recalculate free space
30620 0000A49E E8F61A0000              <1>       call   calculate_fat_freespace
30621                                  <1>
30622                                  <1> loc_mkdir_save_fat_buffer_2:
30623 0000A4A3 C605[0F590100]01        <1>       mov    byte [mkdir_add_new_cluster], 1
30624 0000A4AA E9C4000000              <1>       jmp    loc_mkdir_upd_parent_dir_lmdt
30625                                  <1>
30626                                  <1> loc_mkdir_update_sub_dir_cluster:
30627 0000A4AF A1[00590100]            <1>       mov    eax, [mkdir_FFCluster]
30628 0000A4B4 29C9                    <1>       sub    ecx, ecx ; 0
30629                                  <1>       ; 01/03/2016
30630 0000A4B6 890D[AE560100]          <1>       mov    [FAT_ClusterCounter], ecx ; 0 ; Reset
30631 0000A4BC 49                      <1>       dec    ecx ; 0FFFFFFFFh
30632                                  <1>
30633                                  <1>       ; ESI = Logical DOS Drive Descisption Table address
30634 0000A4BD E885170000              <1>       call   update_cluster
30635 0000A4C2 7379                    <1>       jnc    short loc_mkdir_save_fat_buffer_3
30636 0000A4C4 09C0                    <1>       or     eax, eax ; EAX = 0 -> cluster value is 0 or eocc
30637 0000A4C6 7475                    <1>       jz     short loc_mkdir_save_fat_buffer_3
30638                                  <1>       ; 01/03/2016
30639 0000A4C8 EB98                    <1>       jmp    short loc_mkdir_save_fat_buffer_stc_retn
30640                                  <1>
30641                                  <1> loc_mkdir_set_ff_dir_entry_2:
30642                                  <1>       ; ESI = Logical DOS Drive Description Table address
30643 0000A4CA A1[00590100]            <1>       mov    eax, [mkdir_FFCluster]
30644                                  <1>       ; Load disk sectors as a directory cluster
30645 0000A4CF E82F160000              <1>       call   load_FAT_sub_directory
30646 0000A4D4 7266                    <1>       jc     short retn_make_fat_directory
30647                                  <1>
30648                                  <1>       ; eax = 0
30649                                  <1>       ; ecx =  directory buffer sector count (<= 128)
30650                                  <1>
30651 0000A4D6 BF40000800              <1>       mov    edi, Directory_Buffer + 64 ; 26/02/2016
30652                                  <1>
30653                                  <1>       ; 02/03/2016
30654 0000A4DB 668B4611                <1>       mov    ax, [esi+LD_BPB+BytesPerSec] ; 512
30655 0000A4DF 66C1E802                <1>       shr    ax, 2 ; 'byte count / 4' for 'stosd'
30656 0000A4E3 F7E1                    <1>       mul    ecx
30657 0000A4E5 89C1                    <1>       mov    ecx, eax
30658 0000A4E7 6629C0                  <1>       sub    ax, ax
30659 0000A4EA F3AB                    <1>       rep    stosd
30660                                  <1>
30661                                  <1>       ;;mov  al, 128 ; double word
30662                                  <1>       ;;mul  ecx ; ecx =  directory buffer sector count
30663                                  <1>       ;;mov  ecx, eax
30664                                  <1>       ;shl   cx, 7 ; 128 * sector count
30665                                  <1>       ;;sub  eax, eax
30666                                  <1>       ;;sub  al, al ; 0
30667                                  <1>       ;rep   stosd ; clear directory buffer
30668                                  <1>
30669 0000A4EC BF00000800              <1>       mov    edi, Directory_Buffer ; 26/02/2016
30670                                  <1>
30671 0000A4F1 56                      <1>       push   esi
30672                                  <1>
30673 0000A4F2 BE[10590100]            <1>       mov    esi, mkdir_Name
30674 0000A4F7 66C7062E00              <1>       mov    word [esi], 2Eh ; db '.', '0'
30675                                  <1>
30676 0000A4FC A1[00590100]            <1>       mov    eax, [mkdir_FFCluster]
30677 0000A501 66B91000                <1>       mov    cx, 10h ; CL = Directory attribute
30678                                  <1>                      ; CH = 0 -> File size is 0
30679 0000A505 E883000000              <1>       call   make_directory_entry
30680                                  <1>
30681 0000A50A BF20000800              <1>       mov    edi, Directory_Buffer + 32 ; 26/02/2016
30682                                  <1>
30683                                  <1>       ; 03/03/2016
30684                                  <1>       ; Following modification has been done according to
30685                                  <1>       ; 'Microsoft Extensible Firmware Initiative
30686                                  <1>       ; FAT32 File System Specification' document,
30687                                  <1>       ; 'FAT: General Overview of On-Disk Format–Page 25'.
30688                                  <1>       ; "Finally, you set DIR_FstClusLO and DIR_FstClusHI
30689                                  <1>       ; for the dotdot entry (the second entry) to the
30690                                  <1>       ; first cluster number of the directory in which you
30691                                  <1>       ; just created the directory (value is 0 if this directory
30692                                  <1>       ; is the root directory even for FAT32 volumes)."
30693                                  <1>       ; (Correctness of this modification has been verified
30694                                  <1>       ;  by using Windows 98 'scandisk.exe'.)
30695                                  <1>
30696 0000A50F 29C0                    <1>       sub    eax, eax
```

```
30697 0000A511 3805[8C4E0100]      <1>       cmp    byte [Current_Dir_Level], al ; 0
30698 0000A517 7605                <1>       jna    short loc_mkdir_set_ff_dir_entry_3
30699 0000A519 A1[884E0100]        <1>       mov    eax, [Current_Dir_FCluster] ; parent dir
30700                              <1> loc_mkdir_set_ff_dir_entry_3:
30701 0000A51E 66C746012E00        <1>       mov    word [esi+1], 2Eh ; db '.', '0'
30702                              <1>
30703                              <1>       ;mov   cx, 10h
30704 0000A524 E864000000          <1>       call   make_directory_entry
30705                              <1>
30706 0000A529 5E                  <1>       pop    esi
30707                              <1>
30708 0000A52A C605[B8560100]02    <1>       mov    byte [DirBuff_ValidData], 2
30709 0000A531 E834010000          <1>       call   save_directory_buffer
30710 0000A536 0F8373FFFFFF        <1>         jnc     loc_mkdir_update_sub_dir_cluster
30711                              <1>
30712                              <1> retn_make_fat_directory:
30713 0000A53C C3                  <1>       retn
30714                              <1>
30715                              <1> loc_mkdir_save_fat_buffer_3:
30716                              <1>       ; 01/03/2016
30717                              <1>       ; byte [FAT_BuffValidData] = 2
30718 0000A53D E8C2190000          <1>       call   save_fat_buffer
30719 0000A542 0F821AFFFFFF        <1>         jc      loc_mkdir_save_fat_buffer_stc_retn
30720                              <1>
30721 0000A548 803D[AE560100]01    <1>       cmp    byte [FAT_ClusterCounter], 1
30722 0000A54F 721B                <1>       jb     short loc_mkdir_save_fat_buffer_4
30723                              <1>
30724                              <1>       ; ESI = Logical DOS Drive Description Table address
30725 0000A551 A1[AE560100]        <1>       mov    eax, [FAT_ClusterCounter]
30726 0000A556 66BB01FF            <1>       mov    bx, 0FF01h ; add free clusters
30727 0000A55A E83A1A0000          <1>       call   calculate_fat_freespace
30728                              <1>
30729                              <1>       ;inc   eax ; 0FFFFFFFFh -> 0 ; recalculation is needed!
30730                              <1>         ;jnz    short loc_mkdir_save_fat_buffer_4
30731                              <1>
30732                              <1>       ; ecx > 0 -> Recalculation is needed
30733 0000A55F 09C9                <1>       or     ecx, ecx
30734 0000A561 7409                <1>         jz      short loc_mkdir_save_fat_buffer_4
30735                              <1>
30736 0000A563 66BB00FF            <1>       mov    bx, 0FF00h ; recalculate free space
30737 0000A567 E82D1A0000          <1>       call   calculate_fat_freespace
30738                              <1>
30739                              <1> loc_mkdir_save_fat_buffer_4:
30740 0000A56C C605[0F590100]00    <1>       mov    byte [mkdir_add_new_cluster], 0
30741                              <1>
30742                              <1> loc_mkdir_upd_parent_dir_lmdt:
30743 0000A573 E88D010000          <1>       call   update_parent_dir_lmdt
30744                              <1>
30745                              <1>       ; 01/03/2016
30746 0000A578 803D[0F590100]00    <1>       cmp    byte [mkdir_add_new_cluster], 0
30747 0000A57F 0F8723FEFFFF        <1>         ja      loc_mkdir_gffc_2
30748                              <1>
30749                              <1> loc_mkdir_retn_new_dir_cluster:
30750 0000A585 A1[00590100]        <1>       mov    eax, [mkdir_FFCluster]
30751 0000A58A 31D2                <1>       xor    edx, edx
30752                              <1> loc_mkdir_retn:
30753 0000A58C C3                  <1>       retn
30754                              <1>
30755                              <1> make_directory_entry:
30756                              <1>       ; 02/03/2016
30757                              <1>       ; 21/02/2016 (TRDOS 386 = TRDOS v2.0)
30758                              <1>       ; 09/08/2010 (DIR.ASM, 'proc_make_directory_entry')
30759                              <1>       ; 17/07/2010
30760                              <1>       ; INPUT ->
30761                              <1>       ;     EDI = Directory Entry Address
30762                              <1>       ;     ESI = Dot File Name Location
30763                              <1>       ;     EAX = First Cluster
30764                              <1>       ;     File Size = 0 (Must be set later)
30765                              <1>       ;     CL = Attributes
30766                              <1>       ;     CH = 0 (File size = 0)
30767                              <1>       ;     (If CH>0, File size is in dword [EBX]) (*)
30768                              <1>       ; OUTPUT ->
30769                              <1>       ;     EDI = Directory Entry Address
30770                              <1>       ;     ESI = Dot File Name Location (Capitalized)
30771                              <1>       ;     If CH input = 0, File Size = 0
30772                              <1>       ;     Otherwise file size is as dword [EBX] (*)
30773                              <1>       ;     DX = Date, AX = Time in DOS Dir Entry format
30774                              <1>       ;     EBX = same
30775                              <1>       ;     ECX = same
30776                              <1>
30777 0000A58D 51                  <1>       push   ecx
30778                              <1>
30779 0000A58E 884F0B              <1>       mov    [edi+11], cl ; Attributes
30780 0000A591 6689471A            <1>       mov    [edi+26], ax ; FClusterLw, 26
30781 0000A595 C1E810              <1>       shr    eax, 16
30782 0000A598 66894714            <1>       mov    [edi+20], ax ; FClusterHw, 20
30783 0000A59C 6631C0              <1>       xor    ax, ax
30784 0000A59F 6689470C            <1>       mov    [edi+12], ax ; NTReserved, 12
30785                              <1>                           ; CrtTimeTenth, 13
30786 0000A5A3 08ED                <1>       or     ch, ch
30787 0000A5A5 7402                <1>       jz     short loc_make_direntry_set_filesize
30788                              <1>
30789 0000A5A7 8B03                <1>       mov    eax, [ebx]
30790                              <1>
30791                              <1> loc_make_direntry_set_filesize:
30792 0000A5A9 89471C              <1>       mov    [edi+28], eax ; FileSize, 28
30793                              <1>
30794 0000A5AC E88AFBFFFF          <1>       call   convert_file_name
30795                              <1>       ;EDI = Dir Entry Format File Name Location
30796                              <1>       ;ESI = Dot File Name Location (capitalized)
30797                              <1>
30798 0000A5B1 E816000000          <1>       call   convert_current_date_time
```

```
30799                                  <1>       ; OUTPUT -> DX = Date in dos dir entry format
30800                                  <1>         ;       AX = Time in dos dir entry format
30801 0000A5B6 6689470E                <1>       mov   [edi+14], ax ; CrtTime, 14
30802 0000A5BA 66895710                <1>       mov   [edi+16], dx ; CrtDate, 16
30803 0000A5BE 66895712                <1>       mov   [edi+18], dx ; LastAccDate, 18
30804 0000A5C2 66894716                <1>       mov   [edi+22], ax ; WrtTime, 14
30805 0000A5C6 66895718                <1>       mov   [edi+24], dx ; WrtDate, 16
30806 0000A5CA 59                      <1>       pop   ecx
30807                                  <1>
30808 0000A5CB C3                      <1>       retn
30809                                  <1>
30810                                  <1> convert_current_date_time:
30811                                  <1>       ; 21/02/2016 (TRDOS 386 = TRDOS v2.0)
30812                                  <1>       ; 13/06/2010 (DIR.ASM, 'proc_convert_current_date_time')
30813                                  <1>       ; converts date&time to dos dir entry format
30814                                  <1>       ; INPUT -> none
30815                                  <1>       ; OUTPUT -> DX = Date in dos dir entry format
30816                                  <1>       ;           AX = Time in dos dir entry format
30817                                  <1>
30818 0000A5CC B404                    <1>       mov   ah, 04h ; Return Current Date
30819 0000A5CE E8A6B3FFFF              <1>       call  int1Ah
30820                                  <1>
30821 0000A5D3 88E8                    <1>       mov   al, ch ; <- century BCD
30822 0000A5D5 240F                    <1>       and   al, 0Fh
30823 0000A5D7 88EC                    <1>       mov   ah, ch
30824 0000A5D9 C0EC04                  <1>       shr   ah, 4
30825 0000A5DC D50A                    <1>       aad
30826 0000A5DE 88C5                    <1>       mov   ch, al ; -> century
30827                                  <1>
30828 0000A5E0 88C8                    <1>       mov   al, cl ; <- year BCD
30829 0000A5E2 240F                    <1>       and   al, 0Fh
30830 0000A5E4 88CC                    <1>       mov   ah, cl
30831 0000A5E6 C0EC04                  <1>       shr   ah, 4
30832 0000A5E9 D50A                    <1>       aad
30833 0000A5EB 88C1                    <1>       mov   cl, al ; -> year
30834                                  <1>
30835 0000A5ED 88E8                    <1>       mov   al, ch
30836 0000A5EF B464                    <1>       mov   ah, 100
30837 0000A5F1 F6E4                    <1>       mul   ah
30838 0000A5F3 30ED                    <1>       xor   ch, ch
30839 0000A5F5 6601C8                  <1>       add   ax, cx
30840 0000A5F8 662DBC07                <1>       sub   ax, 1980 ; ms-dos epoch
30841 0000A5FC 6689C1                  <1>       mov   cx, ax
30842                                  <1>
30843 0000A5FF 88F0                    <1>       mov   al, dh ; <- month in bcd
30844 0000A601 240F                    <1>       and   al, 0Fh
30845 0000A603 88F4                    <1>       mov   ah, dh
30846 0000A605 C0EC04                  <1>       shr   ah, 4
30847 0000A608 D50A                    <1>       aad
30848 0000A60A 88C6                    <1>       mov   dh, al ; -> month
30849                                  <1>
30850 0000A60C 88D0                    <1>       mov   al, dl ; <- day BCD
30851 0000A60E 240F                    <1>       and   al, 0Fh
30852 0000A610 88D4                    <1>       mov   ah, dl
30853 0000A612 C0EC04                  <1>       shr   ah, 4
30854 0000A615 D50A                    <1>       aad
30855 0000A617 88C2                    <1>       mov   dl, al ; -> day
30856                                  <1>
30857 0000A619 88C8                    <1>       mov   al, cl ; count of years from 1980
30858 0000A61B 66C1E004                <1>       shl   ax, 4
30859 0000A61F 08F0                    <1>       or    al, dh ; month of year, 1 to 12
30860 0000A621 66C1E005                <1>       shl   ax, 5
30861 0000A625 08D0                    <1>       or    al, dl ; day of year, 1 to 31
30862                                  <1>
30863 0000A627 6650                    <1>       push  ax ; push date
30864                                  <1>
30865 0000A629 B402                    <1>       mov   ah, 02h ; Return Current Time
30866 0000A62B E849B3FFFF              <1>       call  int1Ah
30867                                  <1>
30868 0000A630 88E8                    <1>       mov   al, ch ; <- hours BCD
30869 0000A632 240F                    <1>       and   al, 0Fh
30870 0000A634 88EC                    <1>       mov   ah, ch
30871 0000A636 C0EC04                  <1>       shr   ah, 4
30872 0000A639 D50A                    <1>       aad
30873 0000A63B 88C5                    <1>       mov   ch, al ; -> hours
30874                                  <1>
30875 0000A63D 88C8                    <1>       mov   al, cl ; <- minutes BCD
30876 0000A63F 240F                    <1>       and   al, 0Fh
30877 0000A641 88CC                    <1>       mov   ah, cl
30878 0000A643 C0EC04                  <1>       shr   ah, 4
30879 0000A646 D50A                    <1>       aad
30880 0000A648 88C1                    <1>       mov   cl, al ; -> minutes
30881                                  <1>
30882 0000A64A 88F0                    <1>       mov   al, dh ; <- seconds BCD
30883 0000A64C 240F                    <1>       and   al, 0Fh
30884 0000A64E 88F4                    <1>       mov   ah, dh
30885 0000A650 C0EC04                  <1>       shr   ah, 4
30886 0000A653 D50A                    <1>       aad
30887 0000A655 88C6                    <1>       mov   dh, al ; -> seconds
30888                                  <1>
30889 0000A657 88E8                    <1>       mov   al, ch ; hours
30890 0000A659 66C1E006                <1>       shl   ax, 6
30891 0000A65D 08C8                    <1>       or    al, cl ; minutes
30892 0000A65F 66C1E005                <1>       shl   ax, 5
30893 0000A663 D0EE                    <1>       shr   dh, 1 ; 2 seconds
30894                                  <1>       ; There is a bug in TRDOS v1 here !
30895                                  <1>       ; it was 'or al, dl' !
30896 0000A665 08F0                    <1>       or    al, dh ; seconds
30897                                  <1>
30898 0000A667 665A                    <1>       pop   dx ; pop date
30899                                  <1>
30900 0000A669 C3                      <1>       retn
```

```
30901                                  <1>
30902                                  <1> save_directory_buffer:
30903                                  <1>       ; 15/10/2016
30904                                  <1>       ; 23/03/2016
30905                                  <1>       ; 26/02/2016
30906                                  <1>       ; 22/02/2016 (TRDOS 386 = TRDOS v2.0)
30907                                  <1>       ; 01/08/2011
30908                                  <1>       ; 14/03/2010
30909                                  <1>       ; INPUT ->
30910                                  <1>       ;       none
30911                                  <1>       ; OUTPUT ->
30912                                  <1>       ;  cf = 0 -> write OK...
30913                                  <1>       ;  cf = 1 -> error code in AL (EAX)
30914                                  <1>       ;  cf = 1 & AL = 0Dh => CH & CL = FS & FAT type
30915                                  <1>       ;  EBX = Directory Buffer Address
30916                                  <1>       ;
30917                                  <1>       ;  (EAX, ECX, EDX will be modified)
30918                                  <1>
30919 0000A66A BB00000800              <1>       mov    ebx, Directory_Buffer
30920 0000A66F 803D[B8560100]02        <1>       cmp    byte [DirBuff_ValidData], 2
30921 0000A676 7403                    <1>       je     short loc_save_dir_buffer
30922 0000A678 31C0                    <1>       xor    eax, eax
30923 0000A67A C3                      <1>       retn
30924                                  <1>
30925                                  <1> loc_save_dir_buffer:
30926 0000A67B 56                      <1>       push   esi
30927 0000A67C 31DB                    <1>       xor    ebx, ebx
30928 0000A67E 8A3D[B6560100]          <1>         mov     bh, [DirBuff_DRV]
30929 0000A684 80EF41                  <1>       sub    bh, 'A'
30930 0000A687 BE00010900              <1>         mov     esi, Logical_DOSDisks
30931 0000A68C 01DE                    <1>       add    esi, ebx
30932 0000A68E 668B4E03                <1>         mov     cx, [esi+LD_FATType]
30933                                  <1>       ; CH = FS Type (A1h for FS)
30934                                  <1>       ; CL = FAT Type (0 for FS)
30935 0000A692 08C9                    <1>       or     cl, cl
30936 0000A694 7433                    <1>       jz     short loc_save_dir_buff_stc_retn
30937                                  <1>
30938                                  <1> loc_save_dir_buffer_check_cluster_no:
30939 0000A696 A1[BD560100]            <1>       mov    eax, [DirBuff_Cluster]
30940 0000A69B 28FF                    <1>       sub    bh, bh ; ebx = 0
30941 0000A69D 09C0                    <1>       or     eax, eax
30942 0000A69F 7540                    <1>       jnz    short loc_save_sub_dir_buffer
30943 0000A6A1 8A25[B7560100]          <1>       mov    ah, [DirBuff_FATType]
30944 0000A6A7 FEC3                    <1>       inc    bl ;  bl = 1
30945 0000A6A9 38DC                    <1>       cmp    ah, bl
30946 0000A6AB 721D                    <1>       jb     short loc_save_dir_buff_inv_data_retn
30947 0000A6AD FEC3                    <1>       inc    bl ; bl = 2
30948 0000A6AF 38E3                    <1>       cmp    bl, ah
30949 0000A6B1 7217                    <1>       jb     short loc_save_dir_buff_inv_data_retn
30950                                  <1>
30951                                  <1> loc_save_root_dir_buffer:
30952 0000A6B3 668B5E17                <1>       mov    bx, [esi+LD_BPB+RootDirEnts]
30953 0000A6B7 6683C30F                <1>       add    bx, 15
30954 0000A6BB 66C1EB04                <1>       shr    bx, 4 ; 16 dir entries per sector
30955 0000A6BF 6609DB                  <1>       or     bx, bx
30956 0000A6C2 7405                    <1>       jz     short loc_save_dir_buff_stc_retn
30957                                  <1>       ;mov   ecx, ebx
30958 0000A6C4 8B4664                  <1>       mov    eax, [esi+LD_ROOTBegin] ; 26/02/2016
30959 0000A6C7 EB23                    <1>       jmp    short loc_write_directory_to_disk
30960                                  <1>
30961                                  <1> loc_save_dir_buff_stc_retn:
30962 0000A6C9 F9                      <1>       stc
30963                                  <1> loc_save_dir_buff_inv_data_retn:
30964                                  <1>       ; 15/10/2016 (0Dh -> 29)
30965 0000A6CA B01D                    <1>       mov    al, 29 ; Invalid data !
30966 0000A6CC C605[B8560100]00        <1>       mov    byte [DirBuff_ValidData], 0
30967 0000A6D3 EB05                    <1>       jmp    short loc_save_dir_buff_retn
30968                                  <1>
30969                                  <1> loc_write_directory_to_disk_err:
30970                                  <1>       ; 15/10/2016 (disk write error code, 1Dh -> 18)
30971 0000A6D5 B812000000              <1>       mov    eax, 18 ; Drive not ready or write error
30972                                  <1>
30973                                  <1> loc_save_dir_buff_retn:
30974 0000A6DA BB00000800              <1>       mov    ebx, Directory_Buffer
30975 0000A6DF 5E                      <1>       pop    esi
30976 0000A6E0 C3                      <1>       retn
30977                                  <1>
30978                                  <1> loc_save_sub_dir_buffer:
30979                                  <1>       ; ebx  = 0
30980 0000A6E1 83E802                  <1>       sub    eax, 2
30981 0000A6E4 8A5E13                  <1>       mov    bl, [esi+LD_BPB+SecPerClust]
30982 0000A6E7 F7E3                    <1>       mul    ebx
30983 0000A6E9 034668                  <1>         add     eax, [esi+LD_DATABegin]
30984                                  <1>       ;mov   ecx, ebx
30985                                  <1>
30986                                  <1> loc_write_directory_to_disk:
30987 0000A6EC 89D9                    <1>       mov    ecx, ebx
30988 0000A6EE BB00000800              <1>       mov    ebx, Directory_Buffer
30989 0000A6F3 E8974A0000              <1>       call   disk_write
30990 0000A6F8 72DB                    <1>       jc     short loc_write_directory_to_disk_err
30991                                  <1>
30992                                  <1> loc_save_dir_buff_validate_retn:
30993 0000A6FA C605[B8560100]01        <1>       mov    byte [DirBuff_ValidData], 1
30994 0000A701 31C0                    <1>       xor    eax, eax
30995                                  <1>       ; 26/02/2016
30996 0000A703 EBD5                    <1>       jmp    short loc_save_dir_buff_retn
30997                                  <1>
30998                                  <1> update_parent_dir_lmdt:
30999                                  <1>       ; 22/02/2016 (TRDOS 386 = TRDOS v2.0)
31000                                  <1>       ; 01/08/2011
31001                                  <1>       ; 16/10/2010
31002                                  <1>       ;
```

```
31003                                  <1>       ; INPUT ->
31004                                  <1>       ;     none
31005                                  <1>       ; OUTPUT ->
31006                                  <1>       ;     (last modification date & time of the parent dir
31007                                  <1>       ;     will be changed/updated)
31008                                  <1>       ;
31009                                  <1>       ; (EAX, EBX, ECX, EDX, EDI will be changed)
31010                                  <1>
31011 0000A705 29C0                    <1>       sub    eax, eax
31012 0000A707 8A25[8C4E0100]          <1>       mov    ah, [Current_Dir_Level]
31013 0000A70D A0[8D4E0100]            <1>       mov    al, [Current_FATType]
31014 0000A712 3C01                    <1>       cmp    al, 1
31015 0000A714 723A                    <1>       jb     short loc_UPDLMDT_proc_retn
31016                                  <1>
31017                                  <1> loc_update_parent_dir_lm_date_time:
31018 0000A716 08E4                    <1>       or     ah, ah
31019 0000A718 7436                    <1>       jz     short loc_UPDLMDT_proc_retn
31020                                  <1>
31021 0000A71A 56                      <1>       push   esi ; *
31022 0000A71B 8825[30590100]          <1>       mov    [UPDLMDT_CDirLevel], ah
31023 0000A721 8B15[884E0100]          <1>       mov    edx, [Current_Dir_FCluster]
31024 0000A727 8915[31590100]          <1>       mov    [UPDLMDT_CDirFCluster], edx
31025                                  <1>
31026 0000A72D FECC                    <1>       dec    ah
31027 0000A72F B90C000000              <1>       mov    ecx, 12
31028 0000A734 BE[EF560100]            <1>         mov     esi, PATH_Array
31029                                  <1>
31030 0000A739 8825[8C4E0100]          <1>       mov    [Current_Dir_Level], ah
31031 0000A73F 08E4                    <1>       or     ah, ah
31032 0000A741 750E                    <1>       jnz    short loc_update_parent_dir_lmdt_load_sub_dir_1
31033 0000A743 803D[8D4E0100]02        <1>       cmp    byte [Current_FATType], 2
31034 0000A74A 770B                    <1>       ja     short loc_update_parent_dir_lmdt_load_sub_dir_2
31035 0000A74C 28C0                    <1>       sub    al, al ; eax = 0
31036 0000A74E EB0A                    <1>       jmp    short loc_update_parent_dir_lmdt_load_sub_dir_3
31037                                  <1>
31038                                  <1> loc_UPDLMDT_proc_retn:
31039 0000A750 C3                      <1>       retn
31040                                  <1>
31041                                  <1> loc_update_parent_dir_lmdt_load_sub_dir_1:
31042 0000A751 B010                    <1>       mov    al, 16
31043 0000A753 F6E4                    <1>       mul    ah
31044 0000A755 01C6                    <1>       add    esi, eax
31045                                  <1>
31046                                  <1> loc_update_parent_dir_lmdt_load_sub_dir_2:
31047 0000A757 8B460C                  <1>       mov    eax, [esi+12] ; Parent Dir First Cluster
31048                                  <1>
31049                                  <1> loc_update_parent_dir_lmdt_load_sub_dir_3:
31050 0000A75A A3[884E0100]            <1>       mov    [Current_Dir_FCluster], eax
31051                                  <1>
31052 0000A75F 83C610                  <1>       add    esi, 16
31053 0000A762 66BF[1658]              <1>       mov    di, Dir_File_Name
31054 0000A766 F3A4                    <1>       rep    movsb
31055                                  <1>
31056 0000A768 BE00010900              <1>       mov    esi, Logical_DOSDisks
31057 0000A76D 29DB                    <1>       sub    ebx, ebx
31058 0000A76F 8A3D[8E4E0100]          <1>       mov    bh, [Current_Drv]
31059 0000A775 01DE                    <1>       add    esi, ebx
31060 0000A777 E88FF7FFFF              <1>       call   reload_current_directory
31061 0000A77C 7232                    <1>       jc     short loc_update_parent_dir_lmdt_restore_cdirlevel
31062                                  <1>
31063                                  <1> loc_update_parent_dir_lmdt_locate_dir:
31064 0000A77E BE[16580100]            <1>       mov    esi, Dir_File_Name
31065 0000A783 6631C9                  <1>       xor    cx, cx
31066 0000A786 66B81008                <1>       mov    ax, 0810h ; Only directories
31067 0000A78A E8B7F6FFFF              <1>         call    locate_current_dir_file
31068                                  <1>       ; EDI = DirBuff Directory Entry Address
31069 0000A78F 721F                    <1>       jc short loc_update_parent_dir_lmdt_restore_cdirlevel
31070                                  <1>
31071 0000A791 E836FEFFFF              <1>       call   convert_current_date_time
31072 0000A796 66895712                <1>       mov    [edi+18], dx ; Last Access Date
31073 0000A79A 66895718                <1>       mov    [edi+24], dx ; Last Write Date
31074 0000A79E 66894716                <1>       mov    [edi+22], ax ; Last Write Time
31075                                  <1>
31076 0000A7A2 C605[B8560100]02        <1>       mov    byte [DirBuff_ValidData], 2
31077 0000A7A9 E8BCFEFFFF              <1>       call   save_directory_buffer
31078 0000A7AE 7200                    <1>       jc     short loc_update_parent_dir_lmdt_restore_cdirlevel
31079                                  <1>       ;xor   al, al
31080                                  <1> loc_update_parent_dir_lmdt_restore_cdirlevel:
31081                                  <1>       ;current directory level restoration
31082 0000A7B0 8A25[30590100]          <1>       mov    ah, [UPDLMDT_CDirLevel]
31083 0000A7B6 8825[8C4E0100]          <1>       mov    [Current_Dir_Level], ah
31084 0000A7BC 8B15[31590100]          <1>         mov     edx, [UPDLMDT_CDirFCluster]
31085 0000A7C2 8915[884E0100]          <1>       mov    [Current_Dir_FCluster], edx
31086                                  <1>
31087 0000A7C8 5E                      <1>       pop    esi ; *
31088 0000A7C9 C3                      <1>       retn
31089                                  <1>
31090                                  <1> delete_longname:
31091                                  <1>       ; 27/02/2016 (TRDOS 386 = TRDOS v2.0)
31092                                  <1>       ; 01/08/2011 (DIR.ASM, 'proc_delete_longname')
31093                                  <1>       ; 14/03/2010
31094                                  <1>       ; INPUT ->
31095                                  <1>       ;     EAX = Directory Entry (Index) Number (< 65536)
31096                                  <1>       ; OUTPUT ->
31097                                  <1>       ;     cf = 0 -> OK  (EAX = 0)
31098                                  <1>       ;     cf = 1 -> error code in EAX (AL)
31099                                  <1>       ;
31100                                  <1>       ; (Modified registers: EAX, EDX, ECX, EBX, EDI)
31101                                  <1>
31102 0000A7CA 66A3[60590100]          <1>       mov    [DLN_EntryNumber], ax
31103 0000A7D0 C605[62590100]40        <1>         mov     byte [DLN_40h], 40h
31104                                  <1>
```

```
31105 0000A7D7 E858000000          <1>       call   locate_current_dir_entry
31106 0000A7DC 7308                <1>       jnc    short loc_dln_check_attributes
31107 0000A7DE C3                  <1>       retn
31108                              <1>
31109                              <1> loc_dln_longname_not_found:
31110 0000A7DF B802000000          <1>       mov    eax, 2
31111 0000A7E4 F9                  <1>       stc
31112 0000A7E5 C3                  <1>       retn
31113                              <1>
31114                              <1> loc_dln_check_attributes:
31115 0000A7E6 B00F                <1>       mov    al, 0Fh  ; long name
31116 0000A7E8 8A670B              <1>       mov    ah, [edi+0Bh] ; dir entry attributes
31117 0000A7EB 38C4                <1>       cmp    ah, al
31118 0000A7ED 75F0                <1>       jne    short loc_dln_longname_not_found
31119 0000A7EF 8A27                <1>       mov    ah, [edi]
31120 0000A7F1 2A25[62590100]      <1>       sub    ah, [DLN_40h]
31121 0000A7F7 76E6                <1>       jna    short loc_dln_longname_not_found
31122 0000A7F9 80FC14              <1>       cmp    ah, 14h ; 84-64=20 -> 20*13=260 bytes
31123 0000A7FC 77E1                <1>       ja     short loc_dln_longname_not_found
31124                              <1>
31125 0000A7FE C607E5              <1>       mov    byte [edi], 0E5h  ; deleted sign
31126 0000A801 C605[B8560100]02    <1>       mov    byte [DirBuff_ValidData], 2 ; changed/write sign
31127 0000A808 C605[62590100]00    <1>       mov    byte [DLN_40h], 0 ; 40h -> 0
31128                              <1>
31129                              <1> loc_dln_delete_next_ln_entry:
31130 0000A80F 80FC01              <1>       cmp    ah, 1
31131 0000A812 7616                <1>       jna    short loc_dln_longname_retn
31132                              <1> loc_dln_delete_next_ln_entry_0:
31133 0000A814 66FF05[60590100]    <1>       inc    word [DLN_EntryNumber]
31134 0000A81B 0FB705[60590100]    <1>       movzx  eax, word [DLN_EntryNumber]
31135 0000A822 E80D000000          <1>       call   locate_current_dir_entry
31136 0000A827 73BD                <1>       jnc    short loc_dln_check_attributes
31137                              <1>
31138                              <1> loc_dln_longname_stc_retn:
31139 0000A829 C3                  <1>       retn
31140                              <1>
31141                              <1> loc_dln_longname_retn:
31142                              <1>       ;cmp   byte [DirBuff_ValidData], 2
31143                              <1>       ;jne   short loc_dln_longname_retn_xor_eax
31144 0000A82A E83BFEFFFF          <1>       call   save_directory_buffer
31145 0000A82F 72F8                <1>       jc     short loc_dln_longname_stc_retn
31146                              <1>
31147                              <1> loc_dln_longname_retn_xor_eax:
31148 0000A831 31C0                <1>       xor    eax, eax
31149 0000A833 C3                  <1>       retn
31150                              <1>
31151                              <1> locate_current_dir_entry:
31152                              <1>       ; 16/10/2016
31153                              <1>       ; 15/10/2016
31154                              <1>       ; 23/03/2016
31155                              <1>       ; 27/02/2016 (TRDOS 386 = TRDOS v2.0)
31156                              <1>       ; 01/08/2011 (DIR.ASM, 'proc_locate_current_dir_entry')
31157                              <1>       ; 07/03/2010
31158                              <1>       ; INPUT ->
31159                              <1>       ;     EAX = Directory Entry (Index) Number (< 65536)
31160                              <1>       ; OUTPUT ->
31161                              <1>       ;     EDI = Directory Entry Address
31162                              <1>       ;     EAX = Cluster Number of Directory Buffer
31163                              <1>       ;     EBX = Directory Buffer Entry Offset
31164                              <1>       ;     ECX = DirBuff Valid Data identifier (CL)
31165                              <1>       ;     If CF = 0 and CL = 2 then
31166                              <1>       ;        directory buffer modified and
31167                              <1>       ;        must be written to disk.
31168                              <1>       ;     If CF = 0  and CL = 1 then
31169                              <1>       ;        dir buffer has been written to disk, already.
31170                              <1>       ;     CF = 1 -> Error code in EAX (AL)
31171                              <1>       ;
31172                              <1>       ; (Modified registers: EAX, EDX, ECX, EBX, EDI)
31173                              <1>
31174                              <1> loc_locate_current_dir_entry:
31175 0000A834 56                  <1>       push   esi
31176 0000A835 89C1                <1>       mov    ecx, eax
31177 0000A837 BA20000000          <1>       mov    edx, 32
31178 0000A83C F7E2                <1>       mul    edx
31179 0000A83E A3[6C590100]        <1>       mov    [LCDE_ByteOffset], eax
31180 0000A843 31DB                <1>       xor    ebx, ebx
31181 0000A845 8A3D[8E4E0100]      <1>       mov    bh, [Current_Drv]
31182 0000A84B A0[B6560100]        <1>         mov     al, [DirBuff_DRV]
31183 0000A850 2C41                <1>       sub    al, 'A'
31184 0000A852 BE00010900          <1>         mov     esi, Logical_DOSDisks
31185 0000A857 01DE                <1>       add    esi, ebx
31186 0000A859 38C7                <1>       cmp    bh, al
31187 0000A85B 0F8592000000        <1>         jne     loc_lcde_reload_current_directory
31188                              <1> loc_lcde_cdl_check:
31189 0000A861 803D[8C4E0100]00    <1>       cmp    byte [Current_Dir_Level], 0
31190 0000A868 772A                <1>       ja     short loc_lcde_calc_dirbuff_cluster_offset
31191                              <1>       ; 27/02/2016
31192                              <1>       ; TRDOS v1 has bug here for FAT32 fs !
31193                              <1>       ; (Root Directory Entries for FAT32 = 0)
31194 0000A86A 807E0303            <1>       cmp    byte [esi+LD_FATType], 3  ; FAT32
31195 0000A86E 7324                <1>       jnb    short loc_lcde_calc_dirbuff_cluster_offset
31196                              <1>
31197                              <1> loc_lcde_cdl_check_FAT12_16:
31198 0000A870 668B4617            <1>       mov    ax, [esi+LD_BPB+RootDirEnts]
31199 0000A874 6648                <1>       dec    ax
31200                              <1>       ;xor   dx, dx
31201 0000A876 6639C8              <1>       cmp    ax, cx ; cx = Directory Entry (Index) Number
31202 0000A879 720E                <1>       jb     short loc_lcde_stc_12h_retn
31203 0000A87B 66890D[64590100]    <1>       mov    [LCDE_EntryIndex], cx
31204 0000A882 31C0                <1>       xor    eax, eax
31205 0000A884 E993000000          <1>         jmp     loc_lcde_check_dir_buffer_cluster
31206                              <1>
```

```
                                <1> loc_lcde_stc_12h_retn:
0000A889 5E                     <1>       pop    esi
0000A88A 89CB                   <1>       mov    ebx, ecx
0000A88C 89D1                   <1>       mov    ecx, edx
                                <1>       ; 16/10/2016 (12h -> 12)
0000A88E B80C000000             <1>       mov    eax, 12 ; No more files
0000A893 C3                     <1>       retn
                                <1>
                                <1> loc_lcde_calc_dirbuff_cluster_offset:
0000A894 8A5E13                 <1>       mov    bl, [esi+LD_BPB+SecPerClust]
0000A897 30FF                   <1>       xor    bh, bh
0000A899 668B4611               <1>       mov    ax, [esi+LD_BPB+BytesPerSec]
0000A89D 66F7E3                 <1>       mul    bx
0000A8A0 6609D2                 <1>       or     dx, dx ; If bytes per cluster > 32KB it is invalid
0000A8A3 755D                   <1>       jnz    short loc_lcde_invalid_format
                                <1>       ;mov   ecx, eax
0000A8A5 6689C1                 <1>       mov    cx, ax ; BYTES PER CLUSTER
0000A8A8 A1[6C590100]           <1>       mov    eax, [LCDE_ByteOffset]
                                <1>       ;sub   edx, edx
0000A8AD F7F1                   <1>       div    ecx
0000A8AF 3DFFFF0000             <1>       cmp    eax, 65535
0000A8B4 774C                   <1>       ja     short loc_lcde_invalid_format
                                <1>
                                <1>       ; cluster sequence number of directory (< 65536)
0000A8B6 66A3[66590100]         <1>       mov    [LCDE_ClusterSN], ax
                                <1>
0000A8BC 6689D0                 <1>       mov    ax, dx ; byte offset in cluster (directory buffer)
0000A8BF 66BB2000               <1>       mov    bx, 32 ; ; 1 dir entry = 32 bytes
0000A8C3 6629D2                 <1>         sub     dx, dx  ; 0
0000A8C6 66F7F3                 <1>       div    bx
0000A8C9 66A3[64590100]         <1>       mov    [LCDE_EntryIndex], ax ; dir entry index/sequence number
                                <1>                                    ; (in directory buffer/cluster)
                                <1> loc_lcde_get_current_sub_dir_fcluster:
0000A8CF A1[884E0100]           <1>       mov    eax, [Current_Dir_FCluster]
                                <1>
                                <1> loc_lcde_get_next_cluster:
0000A8D4 66833D[66590100]00     <1>       cmp    word [LCDE_ClusterSN], 0
0000A8DC 763E                   <1>       jna    short loc_lcde_check_dir_buffer_cluster
0000A8DE A3[68590100]           <1>       mov    [LCDE_Cluster], eax
0000A8E3 E835100000             <1>       call   get_next_cluster
0000A8E8 7220                   <1>       jc     short loc_lcde_check_gnc_error
0000A8EA 66FF0D[66590100]       <1>       dec    word [LCDE_ClusterSN]
0000A8F1 EBE1                   <1>       jmp    short loc_lcde_get_next_cluster
                                <1>
                                <1> loc_lcde_reload_current_directory:
0000A8F3 51                     <1>       push   ecx
0000A8F4 E812F6FFFF             <1>       call   reload_current_directory
0000A8F9 59                     <1>       pop    ecx
0000A8FA 0F8361FFFFFF           <1>         jnc     loc_lcde_cdl_check
0000A900 5E                     <1>       pop    esi
0000A901 C3                     <1>       retn
                                <1>
                                <1> loc_lcde_invalid_format:
                                <1>       ; 15/10/2016 (0Bh -> 28)
0000A902 B81C000000             <1>       mov    eax, 28 ; Invalid Format !
                                <1> loc_lcde_drive_not_ready_read_err:
0000A907 F9                     <1>       stc
0000A908 5E                     <1>       pop    esi
0000A909 C3                     <1>       retn
                                <1>
                                <1> loc_lcde_check_gnc_error:
0000A90A 09C0                   <1>       or     eax, eax
0000A90C 75F9                   <1>       jnz    short loc_lcde_drive_not_ready_read_err
0000A90E 66FF0D[66590100]       <1>       dec    word [LCDE_ClusterSN]
0000A915 75EB                   <1>       jnz    short loc_lcde_invalid_format
0000A917 A1[68590100]           <1>       mov    eax, [LCDE_Cluster]
                                <1>
                                <1> loc_lcde_check_dir_buffer_cluster:
0000A91C 3B05[BD560100]         <1>       cmp    eax, [DirBuff_Cluster]
0000A922 755C                   <1>       jne    short loc_lcde_load_dir_cluster
0000A924 803D[B8560100]00       <1>       cmp    byte [DirBuff_ValidData], 0
0000A92B 7727                   <1>       ja     short lcde_check_dir_buffer_cluster_next
0000A92D 803D[8C4E0100]00       <1>       cmp    byte [Current_Dir_Level], 0
0000A934 775F                   <1>       ja     short loc_lcde_load_dir_cluster_0
                                <1>       ; 27/02/2016
                                <1>       ; TRDOS v1 has bug here for FAT32 fs !
0000A936 807E0303               <1>       cmp    byte [esi+LD_FATType], 3  ; FAT32
0000A93A 7359                   <1>       jnb    short loc_lcde_load_dir_cluster_0
                                <1>       ;
0000A93C 0FB74E17               <1>       movzx  ecx, word [esi+LD_BPB+RootDirEnts]
0000A940 6683C10F               <1>       add    cx, 15 ; round up (16 entries per sector)
0000A944 66C1E904               <1>       shr    cx, 4 ; 1 sector contains 16 dir entries
                                <1>
0000A948 8B4664                 <1>         mov     eax, [esi+LD_ROOTBegin]
0000A94B EB54                   <1>       jmp    short loc_lcde_load_dir_cluster_1
                                <1>
                                <1> loc_lcde_validate_dirBuff:
0000A94D C605[B8560100]01       <1>       mov    byte [DirBuff_ValidData], 1
                                <1>
                                <1> lcde_check_dir_buffer_cluster_next:
0000A954 0FB71D[64590100]       <1>       movzx  ebx, word [LCDE_EntryIndex]
0000A95B 663B1D[BB560100]       <1>       cmp    bx, [DirBuff_LastEntry]
0000A962 779E                   <1>       ja     short loc_lcde_invalid_format
0000A964 B820000000             <1>       mov    eax, 32
0000A969 F7E3                   <1>       mul    ebx
                                <1>       ;or    edx, edx
                                <1>       ;jnz   short loc_lcde_invalid_format
                                <1>
0000A96B BF00000800             <1>       mov    edi, Directory_Buffer
0000A970 01C7                   <1>       add    edi, eax ; add entry offset to buffer address
                                <1>
                                <1> loc_lcde_dir_buffer_last_check:
```

```
31309 0000A972 A1[BD560100]        <1>       mov    eax, [DirBuff_Cluster]
31310 0000A977 0FB60D[B8560100]    <1>       movzx  ecx, byte [DirBuff_ValidData]
31311                              <1>
31312                              <1> loc_lcde_retn:
31313 0000A97E 5E                  <1>       pop    esi
31314 0000A97F C3                  <1>       retn
31315                              <1>
31316                              <1> loc_lcde_load_dir_cluster:
31317                              <1>       ;cmp   byte [DirBuff_ValidData], 2
31318                              <1>       ;jne   short loc_lcde_load_dir_cluster_n2
31319 0000A980 50                  <1>       push   eax
31320 0000A981 E8E4FCFFFF          <1>       call   save_directory_buffer
31321 0000A986 58                  <1>       pop    eax
31322 0000A987 72F5                <1>       jc     short loc_lcde_retn
31323                              <1>
31324                              <1> loc_lcde_load_dir_cluster_n2:
31325 0000A989 C605[B8560100]00    <1>       mov    byte [DirBuff_ValidData], 0
31326 0000A990 A3[BD560100]        <1>       mov    [DirBuff_Cluster], eax
31327                              <1>
31328                              <1> loc_lcde_load_dir_cluster_0:
31329 0000A995 83E802              <1>       sub    eax, 2
31330 0000A998 0FB64E13            <1>       movzx  ecx, byte [esi+LD_BPB+SecPerClust]
31331 0000A99C F7E1                <1>       mul    ecx
31332 0000A99E 034668              <1>         add     eax, [esi+LD_DATABegin]
31333                              <1>
31334                              <1> loc_lcde_load_dir_cluster_1:
31335 0000A9A1 BB00000800          <1>       mov    ebx, Directory_Buffer
31336                              <1>       ; ecx = sector count
31337 0000A9A6 E8F3470000          <1>       call   disk_read
31338 0000A9AB 73A0                <1>       jnc    short loc_lcde_validate_dirBuff
31339                              <1>
31340                              <1>       ; 15/10/2016
31341                              <1>       ; (Disk read error instead of drv not ready err)
31342 0000A9AD B811000000          <1>       mov    eax, 17 ; Drive not ready or read error !
31343 0000A9B2 EBCA                <1>       jmp    short loc_lcde_retn
31344                              <1>
31345                              <1>
31346                              <1> remove_file:
31347                              <1>       ; 15/10/2016
31348                              <1>       ; 28/02/2016 (TRDOS 386 = TRDOS v2.0)
31349                              <1>       ; 10/04/2011 (FILE.ASM, 'proc_delete_file')
31350                              <1>       ; 09/08/2010
31351                              <1>       ; INPUT ->
31352                              <1>       ;      EDI = Directory Buffer Entry Address
31353                              <1>       ;       CX = Directory Buffer Entry Counter/Index
31354                              <1>       ;       BL = Longname Entry Length
31355                              <1>       ;       BH = Logical DOS Drive Number
31356                              <1>
31357 0000A9B4 29C0                <1>       sub    eax, eax
31358 0000A9B6 88FC                <1>       mov    ah, bh
31359 0000A9B8 BE00010900          <1>       mov    esi, Logical_DOSDisks
31360 0000A9BD 01C6                <1>       add    esi, eax
31361                              <1>
31362 0000A9BF 807E0301            <1>       cmp    byte [esi+LD_FATType], 1
31363 0000A9C3 7312                <1>       jnb    short loc_del_fat_file
31364                              <1>
31365 0000A9C5 807E04A1            <1>       cmp    byte [esi+LD_FSType], 0A1h
31366 0000A9C9 7406                <1>       je     short loc_del_fs_file
31367                              <1>
31368                              <1> loc_del_file_invalid_format:
31369 0000A9CB 30E4                <1>       xor    ah, ah
31370                              <1>       ; 15/10/2016 (0Bh -> 28)
31371 0000A9CD B01C                <1>       mov    al, 28  ; Invalid Format
31372 0000A9CF F9                  <1>       stc
31373 0000A9D0 C3                  <1>       retn
31374                              <1>
31375                              <1> loc_del_fs_file:
31376 0000A9D1 E8400F0000          <1>       call   delete_fs_file
31377 0000A9D6 C3                  <1>       retn
31378                              <1>
31379                              <1> loc_del_fat_file:
31380 0000A9D7 E808000000          <1>       call   delete_directory_entry
31381 0000A9DC 7205                <1>       jc     short loc_del_file_err_retn
31382                              <1>
31383                              <1> loc_delfile_unlink_cluster_chain:
31384 0000A9DE E864170000          <1>       call   truncate_cluster_chain
31385                              <1>       ;jc    short loc_del_file_err_retn
31386                              <1>
31387                              <1> loc_delfile_return:
31388                              <1> loc_del_file_err_retn:
31389 0000A9E3 C3                  <1>       retn
31390                              <1>
31391                              <1> delete_directory_entry:
31392                              <1>       ; 15/10/2016
31393                              <1>       ; 28/02/2016 (TRDOS 386 = TRDOS v2.0)
31394                              <1>       ; 01/08/2011 (DIR.ASM, 'proc_delete_directory_entry')
31395                              <1>       ; 10/04/2011
31396                              <1>       ; INPUT ->
31397                              <1>       ;      ESI = Logical Dos Drive Descripton Table Address
31398                              <1>       ;      EDI = Directory Buffer Entry Address
31399                              <1>       ;       CX = Directory Buffer Entry Counter/Index
31400                              <1>       ;       BL = Longname Entry Length
31401                              <1>       ; OUTPUT ->
31402                              <1>       ;      ESI = Logical dos drive descripton table address
31403                              <1>       ;      EAX = First cluster to be truncated/unlinked
31404                              <1>       ;        CF = 1 -> Error code in EAX (AL)
31405                              <1>       ;        CF = 0 & BH <> 0 -> LMDT write error  (BH = 1)
31406                              <1>       ;        CF = 0 & BL <> 0 -> Long name delete error (BL = FFh)
31407                              <1>       ;
31408                              <1>       ;  (EDI, EBX, ECX register contents will be changed)
31409                              <1>
31410 0000A9E4 881D[FA580100]      <1>       mov    [DelFile_LNEL], bl
```

```
31411 0000A9EA 66890D[F8580100]    <1>       mov    [DelFile_EntryCounter], cx
31412                              <1>
31413 0000A9F1 668B4714            <1>       mov    ax, [edi+20] ; First Cluster High Word
31414 0000A9F5 C1E010              <1>       shl    eax, 16
31415 0000A9F8 668B471A            <1>       mov    ax, [edi+26] ; First Cluster Low Word
31416                              <1>
31417 0000A9FC A3[F4580100]        <1>       mov    [DelFile_FCluster], eax
31418                              <1>
31419                              <1> loc_del_short_name:
31420 0000AA01 C607E5              <1>       mov    byte [edi], 0E5h  ; Deleted sign
31421                              <1>
31422 0000AA04 C605[B8560100]02    <1>       mov    byte [DirBuff_ValidData], 2
31423 0000AA0B E85AFCFFFF          <1>       call   save_directory_buffer
31424 0000AA10 723D                <1>       jc     short loc_delete_direntry_err_return
31425                              <1>
31426                              <1> loc_del_long_name:
31427 0000AA12 0FB615[FA580100]    <1>       movzx  edx, byte [DelFile_LNEL]
31428 0000AA19 08D2                <1>       or     dl, dl
31429 0000AA1B 7416                <1>       jz     short loc_del_dir_entry_update_parent_dir_lm_date
31430                              <1>
31431 0000AA1D 8835[FA580100]      <1>       mov    byte [DelFile_LNEL], dh ; 0
31432                              <1>
31433 0000AA23 0FB705[F8580100]    <1>       movzx  eax,  word [DelFile_EntryCounter]
31434 0000AA2A 29D0                <1>       sub    eax, edx
31435                              <1>       ;jnc   short loc_del_long_name_continue
31436 0000AA2C 7205                <1>       jc     short loc_del_dir_entry_update_parent_dir_lm_date
31437                              <1>
31438                              <1> ;loc_del_direntry_inv_data_return: ; 15/10/2016 (0Dh -> 29)
31439                              <1> ;     mov    eax, 29 ; 0Dh (TRDOS 8086) ; Invalid data
31440                              <1> ;     retn
31441                              <1>
31442                              <1> loc_del_long_name_continue:
31443                              <1>       ; AX = Directory Entry Number of the long name last entry
31444 0000AA2E E897FDFFFF          <1>       call   delete_longname
31445                              <1>       ;jc    short loc_delete_direntry_err_return
31446                              <1>
31447                              <1> loc_del_dir_entry_update_parent_dir_lm_date:
31448 0000AA33 801D[FA580100]00    <1>       sbb    byte [DelFile_LNEL], 0 ; 0FFh if cf = 1
31449                              <1>
31450 0000AA3A E8C6FCFFFF          <1>       call   update_parent_dir_lmdt
31451 0000AA3F B700                <1>       mov    bh, 0
31452 0000AA41 80D700              <1>       adc    bh, 0
31453                              <1>
31454 0000AA44 8A1D[FA580100]      <1>       mov    bl, byte [DelFile_LNEL]
31455                              <1>
31456                              <1> loc_delete_direntry_return:
31457 0000AA4A A1[F4580100]        <1>       mov    eax, [DelFile_FCluster]
31458                              <1> loc_delete_direntry_err_return:
31459 0000AA4F C3                  <1>       retn
31460                              <1>
31461                              <1> rename_directory_entry:
31462                              <1>       ; 15/10/2016
31463                              <1>       ; 06/03/2016 (TRDOS 386 = TRDOS v2.0)
31464                              <1>       ; 01/08/2011 (DIR.ASM, 'proc_rename_directory_entry')
31465                              <1>       ; 19/11/2010
31466                              <1>       ; INPUT -> (Current Directory)
31467                              <1>       ;     CX = Directory Entry Number
31468                              <1>       ;      EAX = First Cluster number of file or directory
31469                              <1>       ;      EBX = Longname Length (dir entry count) (< 256)
31470                              <1>       ;      ESI = New file (or directory) name (no path).
31471                              <1>       ;           (ASCIIZ string)
31472                              <1>       ; OUTPUT ->
31473                              <1>       ;      CF = 0 -> successfull
31474                              <1>       ;      CF = 1 -> error code in EAX (AL)
31475                              <1>       ;
31476                              <1>       ; (EAX, EBX, ECX, EDX, ESI, EDI will be changed)
31477                              <1>
31478 0000AA50 803D[8D4E0100]00    <1>       cmp    byte [Current_FATType], 0
31479 0000AA57 7706                <1>       ja     short loc_rename_directory_entry
31480                              <1>
31481 0000AA59 E8B90E0000          <1>       call   rename_fs_file_or_directory
31482 0000AA5E C3                  <1>       retn
31483                              <1>
31484                              <1> loc_rename_directory_entry:
31485 0000AA5F 881D[FA580100]      <1>       mov    [DelFile_LNEL], bl
31486 0000AA65 66890D[F8580100]    <1>       mov    [DelFile_EntryCounter], cx
31487 0000AA6C A3[F4580100]        <1>       mov    [DelFile_FCluster], eax
31488                              <1>
31489 0000AA71 0FB7C1              <1>       movzx  eax, cx
31490 0000AA74 E8BBFDFFFF          <1>       call   locate_current_dir_entry
31491 0000AA79 7308                <1>       jnc    short loc_rename_direntry_check_fcluster
31492                              <1>
31493                              <1> loc_rename_direntry_pop_retn:
31494 0000AA7B C3                  <1>       retn
31495                              <1>
31496                              <1> loc_rename_direntry_pop_invd_retn:
31497 0000AA7C F9                  <1>       stc
31498                              <1> loc_rename_direntry_invd_retn:
31499                              <1>       ; 15/10/2016 (0Dh -> 29)
31500 0000AA7D B81D000000          <1>       mov    eax, 29 ; Invalid data
31501                              <1> loc_rename_retn:
31502 0000AA82 C3                  <1>       retn
31503                              <1>
31504                              <1> loc_rename_direntry_check_fcluster:
31505 0000AA83 668B5714            <1>       mov    dx, [edi+20] ; First Cluster HW
31506 0000AA87 66C1E210            <1>       shl    dx, 16
31507 0000AA8B 668B571A            <1>       mov    dx, [edi+26] ; First Cluster LW
31508 0000AA8F 3B15[F4580100]      <1>       cmp    edx, [DelFile_FCluster]
31509 0000AA95 75E5                <1>       jne    short loc_rename_direntry_pop_invd_retn
31510                              <1>       ; ESI = New file (or directory) name. (ASCIIZ string)
31511                              <1>       ; 06/03/2016
31512                              <1>       ; TRDOS v2 - NOTE: 'convert_file_name' procedure
```

```
31513                                  <1>       ; has been modified for eliminating following situation.
31514                                  <1>       ;
31515                                  <1>       ; TRDOS v1 - NOTE: If file/dir name is more than 11 bytes
31516                                  <1>       ; without a dot, attributes (edi+11) byte will be overwritten !
31517                                  <1>       ; (Dot file name input must be proper for 11 byte dir entry
31518                                  <1>       ;  type file name output.)
31519 0000AA97 E89FF6FFFF             <1>       call   convert_file_name
31520                                  <1>
31521 0000AA9C C605[B8560100]02        <1>         mov     byte [DirBuff_ValidData], 2
31522 0000AAA3 E8C2FBFFFF             <1>       call   save_directory_buffer
31523 0000AAA8 72D8                   <1>       jc     short loc_rename_retn
31524                                  <1>
31525                                  <1> loc_rename_direntry_del_ln:
31526 0000AAAA 0FB615[FA580100]        <1>       movzx  edx, byte [DelFile_LNEL]
31527 0000AAB1 08D2                   <1>       or     dl, dl
31528 0000AAB3 7410                   <1>       jz     short loc_rename_direntry_update_parent_dir_lm_date
31529                                  <1>
31530 0000AAB5 0FB705[F8580100]        <1>       movzx  eax, word [DelFile_EntryCounter]
31531 0000AABC 29D0                   <1>       sub    eax, edx
31532 0000AABE 72BD                   <1>       jc     short loc_rename_direntry_invd_retn
31533                                  <1>
31534                                  <1> loc_rename_direntry_del_ln_continue:
31535                                  <1>       ; EAX = Directory Entry Number of the long name last entry
31536 0000AAC0 E805FDFFFF             <1>       call   delete_longname
31537                                  <1>
31538                                  <1> loc_rename_direntry_update_parent_dir_lm_date:
31539 0000AAC5 E83BFCFFFF             <1>       call   update_parent_dir_lmdt
31540 0000AACA 31C0                   <1>       xor    eax, eax
31541 0000AACC C3                     <1>       retn
31542                                  <1>
31543                                  <1> move_source_file_to_destination_file:
31544                                  <1>       ; 15/10/2016
31545                                  <1>       ; 11/03/2016
31546                                  <1>       ; 10/03/2016 (TRDOS 386 = TRDOS v2.0)
31547                                  <1>       ; 01/08/2011 (FILE.ASM)
31548                                  <1>       ; 04/08/2010
31549                                  <1>       ;
31550                                  <1>       ;   Phase 1 -> Check destination file,
31551                                  <1>       ;              'not found' is required
31552                                  <1>       ;   Phase 2 -> Check source file
31553                                  <1>       ;              'found' and proper attributes is required
31554                                  <1>       ;   Phase 3 -> Make destination directory entry,
31555                                  <1>       ;            add new dir cluster or section if it is required
31556                                  <1>       ;   Phase 4 -> Delete source directory entry.
31557                                  <1>       ;       cf = 1 causes to return before the phase 4.
31558                                  <1>       ;    (source file protection against any possible errors)
31559                                  <1>       ;
31560                                  <1>       ; 08/05/2011 major modification
31561                                  <1>       ;            -> destination file deleting is removed
31562                                  <1>       ;            for msdos move/rename compatibility.
31563                                  <1>       ;            (Access denied error will return if
31564                                  <1>       ;            the destination file is found...)
31565                                  <1>       ; INPUT ->
31566                                  <1>       ;      ESI = Source File Pathname (Asciiz)
31567                                  <1>       ;       EDI = Destination File Pathname (Asciiz)
31568                                  <1>       ;       AL = 0 --> Interrupt (System call)
31569                                  <1>       ;       AL > 0 --> Command Interpreter (Question)
31570                                  <1>       ;       AL = 1 --> Question Phase
31571                                  <1>       ;       AL = 2 --> Progress Phase
31572                                  <1>       ; OUTPUT ->
31573                                  <1>       ;      cf = 0 -> OK
31574                                  <1>       ;       EAX = Destination directory first cluster
31575                                  <1>       ;       ESI = Logical DOS drive description table
31576                                  <1>       ;       EBX = Destination file structure offset
31577                                  <1>       ;       CX = 0 (CX > 0 --> calculate free space error)
31578                                  <1>       ;       cf = 1 -> Error code in EAX (AL)
31579                                  <1>       ;
31580                                  <1>       ;  (EDX, ECX, EBX, ESI, EDI will be changed)
31581                                  <1>
31582 0000AACD 3C02                   <1>       cmp    al, 2
31583 0000AACF 0F847F010000           <1>       je     msftdf_df2_check_directory
31584 0000AAD5 A2[7A5A0100]           <1>       mov    [move_cmd_phase], al
31585                                  <1>
31586                                  <1> msftdf_parse_sf_path:
31587                                  <1>       ; ESI = ASCIIZ pathname (Source)
31588 0000AADA 57                     <1>       push   edi
31589 0000AADB BF[78590100]           <1>       mov    edi, SourceFile_Drv
31590 0000AAE0 E821F7FFFF             <1>       call   parse_path_name
31591 0000AAE5 5E                     <1>       pop    esi
31592 0000AAE6 7211                   <1>       jc     short msftdf_psf_retn
31593                                  <1>
31594                                  <1> msftdf_parse_df_path:
31595                                  <1>       ; ESI = ASCIIZ pathname   (Destination)
31596 0000AAE8 BF[F8590100]           <1>       mov    edi, DestinationFile_Drv
31597 0000AAED E814F7FFFF             <1>       call   parse_path_name
31598 0000AAF2 7306                   <1>       jnc    short msftdf_check_sf_drv
31599                                  <1>
31600 0000AAF4 3C01                   <1>       cmp    al, 1 ; File or directory name is not existing
31601 0000AAF6 7602                   <1>       jna    short msftdf_check_sf_drv
31602                                  <1>
31603                                  <1> msftdf_stc_retn:
31604 0000AAF8 F9                     <1>       stc
31605                                  <1> msftdf_psf_retn:
31606 0000AAF9 C3                     <1>       retn
31607                                  <1>
31608                                  <1> msftdf_check_sf_drv:
31609 0000AAFA A0[78590100]           <1>       mov    al, [SourceFile_Drv]
31610                                  <1>
31611                                  <1> msftdf_check_df_drv:
31612 0000AAFF 8A15[F8590100]         <1>       mov    dl, [DestinationFile_Drv]
31613                                  <1>
31614                                  <1> msftdf_compare_sf_df_drv:
```

```
31615 0000AB05 29DB                <1>       sub    ebx, ebx
31616 0000AB07 8A3D[8E4E0100]      <1>       mov    bh, [Current_Drv]
31617 0000AB0D 38C2                <1>       cmp    dl, al
31618 0000AB0F 7409                <1>       je     short msftdf_check_sf_df_drv_ok
31619                              <1>
31620                              <1> msftdf_not_same_drv:
31621                              <1>         ; DL = source file's drive number
31622 0000AB11 88C6                <1>       mov    dh, al ; destination file's drive number
31623                              <1>       ; 15/10/2016 (11h -> 21)
31624 0000AB13 B815000000          <1>       mov    eax, 21 ; Not the same drive
31625 0000AB18 F9                  <1>       stc
31626 0000AB19 C3                  <1>       retn
31627                              <1>
31628                              <1> msftdf_check_sf_df_drv_ok:
31629 0000AB1A 8815[7B5A0100]      <1>       mov    [msftdf_sf_df_drv], dl
31630                              <1>
31631 0000AB20 29C0                <1>         sub  eax, eax
31632 0000AB22 88D4                <1>       mov    ah, dl
31633 0000AB24 0500010900          <1>       add    eax, Logical_DOSDisks
31634 0000AB29 A3[7C5A0100]        <1>       mov    [msftdf_drv_offset], eax
31635                              <1>
31636 0000AB2E 38FA                <1>       cmp    dl, bh ; byte [Current_Drv]
31637 0000AB30 7407                <1>       je     short msftdf_df_check_directory
31638                              <1>
31639                              <1> msftdf_change_drv:
31640 0000AB32 E826C1FFFF          <1>       call   change_current_drive
31641 0000AB37 726D                <1>       jc     short msftdf_df_error_retn
31642                              <1>
31643                              <1> msftdf_check_destination_file:
31644                              <1> msftdf_df_check_directory:
31645 0000AB39 BE[F9590100]        <1>       mov    esi, DestinationFile_Directory
31646 0000AB3E 803E20              <1>       cmp    byte [esi], 20h
31647 0000AB41 760F                <1>       jna    short msftdf_df_find_1
31648                              <1>
31649                              <1> msftdf_df_change_directory:
31650 0000AB43 FE05[98020100]      <1>       inc    byte [Restore_CDIR]
31651 0000AB49 30E4                <1>       xor    ah, ah ; CD_COMMAND sign -> 0
31652 0000AB4B E8A2F0FFFF          <1>       call   change_current_directory
31653 0000AB50 7254                <1>       jc     short msftdf_df_error_retn
31654                              <1>
31655                              <1> ;msftdf_df_change_prompt_dir_string:
31656                              <1> ;     call   change_prompt_dir_string
31657                              <1>
31658                              <1> msftdf_df_find_1:
31659 0000AB52 BE[3A5A0100]        <1>         mov     esi, DestinationFile_Name
31660 0000AB57 803E20              <1>       cmp    byte [esi], 20h
31661 0000AB5A 7631                <1>       jna    short msftdf_df_copy_sf_name
31662                              <1>
31663                              <1> msftdf_df_find_2:
31664 0000AB5C 6631C0              <1>       xor    ax, ax ; DestinationFile_AttributesMask -> any/zero
31665 0000AB5F E88BD4FFFF          <1>       call   find_first_file
31666 0000AB64 0F838D000000        <1>       jnc    msftdf_permission_denied_retn
31667                              <1>
31668                              <1> msftdf_df_check_error_code:
31669                              <1>       ;cmp   eax, 2 ; File not found error
31670 0000AB6A 3C02                <1>       cmp    al, 2
31671 0000AB6C 7537                <1>       jne    short msftdf_df_stc_retn
31672                              <1>
31673                              <1> msftdf_df_check_fname:
31674                              <1>       ; 15/10/2016
31675 0000AB6E BE[3A5A0100]        <1>       mov    esi, DestinationFile_Name ; *
31676 0000AB73 E837D8FFFF          <1>       call   check_filename
31677 0000AB78 7307                <1>       jnc    short msftdf_convert_df_direntry_name
31678                              <1>       ; invalid file name chars !
31679 0000AB7A B81A000000          <1>       mov    eax, ERR_INV_FILE_NAME  ; 26
31680 0000AB7F EB24                <1>       jmp    short msftdf_df_stc_retn
31681                              <1>
31682                              <1> msftdf_convert_df_direntry_name:
31683                              <1>       ; mov  esi, DestinationFile_Name ; *
31684 0000AB81 BF[4A5A0100]        <1>       mov    edi, DestinationFile_DirEntry
31685 0000AB86 E8B0F5FFFF          <1>       call   convert_file_name
31686 0000AB8B EB1A                <1>       jmp    short msftdf_restore_current_dir_1
31687                              <1>
31688                              <1> msftdf_df_copy_sf_name:
31689 0000AB8D 89F7                <1>       mov    edi, esi
31690 0000AB8F 57                  <1>       push   edi
31691 0000AB90 BE[BA590100]        <1>         mov     esi, SourceFile_Name
31692 0000AB95 B90C000000          <1>       mov    ecx, 12
31693                              <1> msftdf_df_copy_sf_name_loop:
31694 0000AB9A AC                  <1>       lodsb
31695 0000AB9B AA                  <1>         stosb
31696 0000AB9C 08C0                <1>       or     al, al
31697 0000AB9E 7402                <1>       jz     short msftdf_df_copy_sf_name_ok
31698 0000ABA0 E2F8                <1>         loop    msftdf_df_copy_sf_name_loop
31699                              <1> msftdf_df_copy_sf_name_ok:
31700 0000ABA2 5E                  <1>       pop    esi
31701 0000ABA3 EBB7                <1>       jmp    short msftdf_df_find_2
31702                              <1>
31703                              <1> msftdf_df_stc_retn:
31704 0000ABA5 F9                  <1>       stc
31705                              <1> msftdf_restore_cdir_failed:
31706                              <1> msftdf_df_error_retn:
31707 0000ABA6 C3                  <1>       retn
31708                              <1>
31709                              <1> msftdf_restore_current_dir_1:
31710 0000ABA7 803D[98020100]00    <1>       cmp    byte [Restore_CDIR], 0
31711 0000ABAE 760D                <1>       jna    short msftdf_sf_check_directory
31712 0000ABB0 8B35[7C5A0100]      <1>       mov    esi, [msftdf_drv_offset]
31713 0000ABB6 E859C1FFFF          <1>       call   restore_current_directory
31714 0000ABBB 72E9                <1>       jc     short msftdf_restore_cdir_failed
31715                              <1>
31716                              <1> msftdf_sf_check_directory:
```

```
31717 0000ABBD BE[79590100]        <1>       mov    esi, SourceFile_Directory
31718 0000ABC2 803E20              <1>       cmp    byte [esi], 20h
31719 0000ABC5 760F                <1>       jna    short msftdf_sf_find
31720                              <1> msftdf_sf_change_directory:
31721 0000ABC7 FE05[98020100]      <1>       inc    byte [Restore_CDIR]
31722 0000ABCD 30E4                <1>       xor    ah, ah ; CD_COMMAND sign -> 0
31723 0000ABCF E81EF0FFFF          <1>       call   change_current_directory
31724 0000ABD4 7227                <1>       jc     short msftdf_return
31725                              <1>
31726                              <1> ;msftdf_sf_change_prompt_dir_string:
31727                              <1> ;     call   change_prompt_dir_string
31728                              <1>
31729                              <1> msftdf_sf_find:
31730 0000ABD6 BE[BA590100]        <1>         mov     esi, SourceFile_Name  ; Offset 66
31731 0000ABDB 66B80018            <1>       mov    ax, 1800h ; Only files
31732 0000ABDF E80BD4FFFF          <1>       call   find_first_file
31733 0000ABE4 7217                <1>       jc     short msftdf_return
31734                              <1>
31735                              <1> msftdf_sf_ambgfn_check:
31736 0000ABE6 6609D2              <1>       or     dx, dx ; Ambiguous filename chars used sign (DX>0)
31737 0000ABE9 7407                <1>       jz     short msftdf_sf_found
31738                              <1>
31739                              <1> msftdf_ambiguous_file_name_error:
31740 0000ABEB B802000000          <1>       mov    eax, 2 ; File not found error
31741 0000ABF0 F9                  <1>       stc
31742 0000ABF1 C3                  <1>       retn
31743                              <1>
31744                              <1> msftdf_sf_found:
31745 0000ABF2 80E31F              <1>       and    bl, 1Fh ; Attributes, D-V-S-H-R
31746 0000ABF5 7416                <1>       jz     short msftdf_save_sf_structure
31747                              <1>
31748                              <1> msftdf_permission_denied_retn:
31749 0000ABF7 B805000000          <1>       mov    eax, 05h ; Access (Permission) denied !
31750 0000ABFC F9                  <1>       stc
31751                              <1> msftdf_rest_cdir_err_retn:
31752                              <1> msftdf_return:
31753 0000ABFD C3                  <1>       retn
31754                              <1>
31755                              <1> msftdf_phase_1_return:
31756 0000ABFE 31C0                <1>       xor    eax, eax
31757 0000AC00 A2[7A5A0100]        <1>       mov    [move_cmd_phase], al ; 0
31758 0000AC05 FEC0                <1>       inc    al ; mov al, 1
31759 0000AC07 BB[54AC0000]        <1>       mov    ebx, msftdf_df2_check_directory
31760                              <1>       ;mov   edx, 0FFFFFFFFh
31761 0000AC0C C3                  <1>       retn
31762                              <1>
31763                              <1> msftdf_save_sf_structure:
31764 0000AC0D BE[84580100]        <1>       mov    esi, FindFile_DirEntry
31765 0000AC12 BF[CA590100]        <1>       mov    edi, SourceFile_DirEntry
31766 0000AC17 B908000000          <1>       mov    ecx, 8
31767 0000AC1C F3A5                <1>       rep    movsd
31768                              <1>
31769                              <1> msftdf_df_copy_sf_parameters:
31770 0000AC1E BE0B000000          <1>       mov    esi, 11
31771 0000AC23 89F7                <1>       mov    edi, esi
31772 0000AC25 81C6[CA590100]      <1>       add    esi, SourceFile_DirEntry
31773 0000AC2B 81C7[4A5A0100]      <1>       add    edi, DestinationFile_DirEntry
31774                              <1>       ;mov   ecx, 21
31775 0000AC31 B115                <1>       mov    cl, 21
31776 0000AC33 F3A4                <1>       rep    movsb
31777                              <1>
31778                              <1> msftdf_restore_current_dir_2:
31779 0000AC35 803D[98020100]00    <1>       cmp    byte [Restore_CDIR], 0
31780 0000AC3C 760D                <1>       jna    short msftdf_df2_check_move_cmd_phase
31781 0000AC3E 8B35[7C5A0100]      <1>       mov    esi, [msftdf_drv_offset]
31782 0000AC44 E8CBC0FFFF          <1>       call   restore_current_directory
31783 0000AC49 72B2                <1>       jc     short msftdf_rest_cdir_err_retn
31784                              <1>
31785                              <1> msftdf_df2_check_move_cmd_phase:
31786 0000AC4B 803D[7A5A0100]01    <1>       cmp    byte [move_cmd_phase], 1
31787 0000AC52 74AA                <1>       je     short msftdf_phase_1_return
31788                              <1>
31789                              <1> msftdf_df2_check_directory:
31790 0000AC54 BE[F9590100]        <1>       mov    esi, DestinationFile_Directory
31791 0000AC59 803E20              <1>       cmp    byte [esi], 20h
31792 0000AC5C 760F                <1>       jna    short msftdf_make_dfde_locate_ffe_on_directory
31793                              <1> msftdf_df2_change_directory:
31794 0000AC5E FE05[98020100]      <1>       inc    byte [Restore_CDIR]
31795 0000AC64 30E4                <1>       xor    ah, ah ; CD_COMMAND sign -> 0
31796 0000AC66 E887EFFFFF          <1>       call   change_current_directory
31797 0000AC6B 7290                <1>       jc     short msftdf_return
31798                              <1>
31799                              <1> ;msftdf_df2_change_prompt_dir_string:
31800                              <1> ;     call   change_prompt_dir_string
31801                              <1>
31802                              <1> msftdf_make_dfde_locate_ffe_on_directory:
31803                              <1>       ; Current directory fcluster <> Directory buffer cluster
31804                              <1>       ; Current directory will be reloaded by
31805                              <1>       ; 'locate_current_dir_file' procedure
31806                              <1>       ;
31807                              <1>       ;xor   ax, ax
31808 0000AC6D 31C0                <1>       xor    eax, eax
31809 0000AC6F 89C1                <1>       mov    ecx, eax
31810 0000AC71 6649                <1>       dec    cx ; FFFFh
31811                              <1>              ; CX = FFFFh -> find first deleted or free entry
31812                              <1>              ; ESI would be ASCIIZ filename address if the call
31813                              <1>              ; would not be for first free or deleted dir entry
31814 0000AC73 E8CEF1FFFF          <1>       call   locate_current_dir_file
31815 0000AC78 733F                <1>       jnc    msftdf_make_dfde_set_ff_dir_entry
31816                              <1>
31817                              <1>       ;cmp   eax, 2
31818 0000AC7A 3C02                <1>         cmp  al, 2
```

```
31819 0000AC7C 7537               <1>       jne    short msftdf_error_retn
31820                             <1>
31821                             <1> msftdf_add_new_dir_entry_check_fs:
31822 0000AC7E 8B35[7C5A0100]     <1>       mov    esi, [msftdf_drv_offset]
31823 0000AC84 A1[BD560100]       <1>       mov    eax, [DirBuff_Cluster]
31824 0000AC89 807E0300           <1>       cmp    byte [esi+LD_FATType], 0
31825 0000AC8D 7711               <1>       ja     short msftdf_add_new_subdir_cluster
31826                             <1>
31827                             <1> msftdf_add_new_fs_subdir_section:
31828                             <1>       ;CL=0, CH=E5h --> deleted entry, CH=0 --> free entry
31829                             <1>         ;xor cx, cx
31830 0000AC8F 30ED               <1>       xor    ch, ch ; cx = 0 --> add a new subdir section
31831 0000AC91 E8830C0000         <1>       call   add_new_fs_section
31832 0000AC96 721E               <1>         jc   short msftdf_dsfde_error_retn
31833                             <1>       ;mov   [createfile_LastDirCluster], eax
31834                             <1>
31835 0000AC98 E8A30E0000         <1>       call   load_FS_sub_directory
31836                             <1>       ;mov   ebx, Directory_Buffer
31837 0000AC9D 7318               <1>       jnc    short msftdf_add_new_fs_subdir_section_ok
31838 0000AC9F C3                 <1>       retn
31839                             <1>
31840                             <1> msftdf_add_new_subdir_cluster:
31841 0000ACA0 E881150000         <1>       call   add_new_cluster
31842 0000ACA5 720F               <1>       jc     short msftdf_dsfde_error_retn
31843                             <1>
31844                             <1>       ;mov   [createfile_LastDirCluster], eax
31845                             <1>
31846 0000ACA7 E8570E0000         <1>       call   load_FAT_sub_directory
31847 0000ACAC 7309               <1>       jnc    short msftdf_add_new_subdir_cluster_ok
31848                             <1>       ; EBX = Directory buffer address
31849                             <1>
31850                             <1> msftdf_ansdc_update_parent_dir_lmdt:
31851                             <1> msftdf_make_dfde_err_upd_pdir_lmdt:
31852 0000ACAE 50                 <1>       push   eax
31853 0000ACAF E851FAFFFF         <1>       call   update_parent_dir_lmdt
31854 0000ACB4 58                 <1>       pop    eax
31855                             <1>
31856                             <1> msftdf_error_retn:
31857 0000ACB5 F9                 <1>       stc
31858                             <1> msftdf_dsfde_restore_cdir_failed:
31859                             <1> msftdf_dsfde_error_retn:
31860 0000ACB6 C3                 <1>       retn
31861                             <1>
31862                             <1> msftdf_add_new_fs_subdir_section_ok:
31863                             <1> msftdf_add_new_subdir_cluster_ok:
31864 0000ACB7 89DF               <1>       mov    edi, ebx ; Directory buffer address
31865                             <1>
31866                             <1> msftdf_make_dfde_set_ff_dir_entry:
31867 0000ACB9 8B15[884E0100]     <1>       mov    edx, [Current_Dir_FCluster]
31868 0000ACBF 8915[E05A0100]     <1>       mov    [createfile_FFCluster], edx
31869                             <1>       ; EDI = Directory entry offset
31870 0000ACC5 BE[4A5A0100]       <1>       mov    esi, DestinationFile_DirEntry
31871 0000ACCA B908000000         <1>       mov    ecx, 8
31872 0000ACCF F3A5               <1>       rep    movsd
31873                             <1>
31874 0000ACD1 C605[B8560100]02   <1>       mov    byte [DirBuff_ValidData], 2
31875 0000ACD8 E88DF9FFFF         <1>       call   save_directory_buffer
31876 0000ACDD 72CF               <1>       jc     short msftdf_make_dfde_err_upd_pdir_lmdt
31877                             <1>
31878                             <1> msftdf_make_dfde_update_pdir_lmdt:
31879 0000ACDF E821FAFFFF         <1>       call   update_parent_dir_lmdt
31880                             <1>
31881                             <1> msftdf_dsfde_restore_current_dir_1:
31882 0000ACE4 803D[98020100]00   <1>       cmp    byte [Restore_CDIR], 0
31883 0000ACEB 760D               <1>       jna    short msftdf_dsfde_check_directory
31884 0000ACED 8B35[7C5A0100]     <1>       mov    esi, [msftdf_drv_offset]
31885 0000ACF3 E81CC0FFFF         <1>       call   restore_current_directory
31886 0000ACF8 72BC               <1>       jc     short msftdf_dsfde_restore_cdir_failed
31887                             <1>
31888                             <1> msftdf_dsfde_check_directory:
31889 0000ACFA BE[79590100]       <1>       mov    esi, SourceFile_Directory
31890 0000ACFF 803E20             <1>       cmp    byte [esi], 20h
31891 0000AD02 760F               <1>       jna    short msftdf_dsfde_find_file
31892                             <1>
31893                             <1> msftdf_dsfde_change_directory:
31894 0000AD04 FE05[98020100]     <1>       inc    byte [Restore_CDIR]
31895 0000AD0A 28E4               <1>       sub    ah, ah ; CD_COMMAND sign -> 0
31896 0000AD0C E8E1EEFFFF         <1>       call   change_current_directory
31897 0000AD11 72A3               <1>       jc     short msftdf_dsfde_error_retn
31898                             <1>
31899                             <1> ;msftdf_dsfde_sf_change_prompt_dir_string:
31900                             <1> ;     call   change_prompt_dir_string
31901                             <1>
31902                             <1> msftdf_dsfde_find_file:
31903 0000AD13 BE[BA590100]       <1>       mov    esi, SourceFile_Name  ; Offset 66
31904 0000AD18 668B460E           <1>       mov    ax, [esi+14] ; 80 -> SourceFile_AttributesMask
31905 0000AD1C E8CED2FFFF         <1>       call   find_first_file
31906 0000AD21 7293               <1>       jc     short msftdf_dsfde_error_retn
31907                             <1>
31908                             <1> msftdf_dsfde_delete_direntry:
31909 0000AD23 8B35[7C5A0100]     <1>       mov    esi, [msftdf_drv_offset]
31910                             <1>
31911 0000AD29 807E0300           <1>       cmp    byte [esi+LD_FATType], 0
31912 0000AD2D 770A               <1>       ja     short msftdf_delete_FAT_direntry
31913                             <1>
31914 0000AD2F 30DB               <1>       xor    bl, bl
31915                             <1>       ; BL = 0 -> File
31916                             <1>       ; EDI -> Directory buffer entry offset/address
31917 0000AD31 E8E40B0000         <1>       call   delete_fs_directory_entry
31918 0000AD36 7315               <1>       jnc    short msftdf_dsfde_restore_current_dir_2
31919 0000AD38 C3                 <1>       retn
31920                             <1>
```

```
31921                                  <1> msftdf_delete_FAT_direntry:
31922 0000AD39 8A1D[81580100]          <1>       mov    bl, [FindFile_LongNameEntryLength]
31923 0000AD3F 668B0D[AC580100]        <1>       mov    cx, [FindFile_DirEntryNumber]
31924                                  <1>       ; ESI = Logical DOS drive description table address
31925                                  <1>       ; EDI = Directory buffer entry offset/address
31926 0000AD46 E899FCFFFF              <1>       call   delete_directory_entry
31927 0000AD4B 721C                    <1>       jc     short msftdf_retn
31928                                  <1>
31929                                  <1> msftdf_dsfde_restore_current_dir_2:
31930 0000AD4D 803D[98020100]00        <1>       cmp    byte [Restore_CDIR], 0
31931 0000AD54 7607                    <1>       jna    short msftdf_new_dir_fcluster_retn
31932                                  <1>       ;mov   esi, [msftdf_drv_offset]
31933 0000AD56 E8B9BFFFFF              <1>       call   restore_current_directory
31934 0000AD5B 720C                    <1>       jc     short msftdf_retn
31935                                  <1>
31936                                  <1> msftdf_new_dir_fcluster_retn:
31937 0000AD5D 31C9                    <1>       xor    ecx, ecx
31938 0000AD5F A1[E05A0100]            <1>       mov    eax, [createfile_FFCluster]
31939 0000AD64 BB[F8590100]            <1>       mov    ebx, DestinationFile_Drv
31940                                  <1>
31941                                  <1> msftdf_retn:
31942 0000AD69 C3                      <1>       retn
31943                                  <1>
31944                                  <1>
31945                                  <1> copy_source_file_to_destination_file:
31946                                  <1>       ; 17/10/2016
31947                                  <1>       ; 16/10/2016
31948                                  <1>       ; 15/10/2016
31949                                  <1>       ; 30/03/2016, 31/03/2016
31950                                  <1>       ; 24/03/2016, 25/03/2016, 28/03/2016
31951                                  <1>       ; 21/03/2016, 22/03/2016, 23/03/2016
31952                                  <1>       ; 16/03/2016, 17/03/2016, 18/03/2016
31953                                  <1>       ; 15/03/2016 (TRDOS 386 = TRDOS v2.0)
31954                                  <1>       ; 02/09/2011 (FILE.ASM 'copy_source_file_to_destination_file')
31955                                  <1>       ; 01/08/2010 - 18/05/2011
31956                                  <1>       ;
31957                                  <1>       ;   Command Interpreter phase 1 enter ->
31958                                  <1>       ;           AL = 1 -> Caller is command interpreter
31959                                  <1>       ;           AL = 2 -> The second call, re-enter/continue
31960                                  <1>       ;   Phase 1 -> Check source file
31961                                  <1>       ;              'found' is required
31962                                  <1>       ;   Phase 2 -> Check destination file,
31963                                  <1>       ;              save 'found' or 'not found' status
31964                                  <1>       ;              'permission denied' error will be return
31965                                  <1>       ;              if attributes have not for ordinary file
31966                                  <1>       ;              without readonly attribute
31967                                  <1>       ;   Command Interpreter phase 1 return ->
31968                                  <1>       ;              DH = Source file attributes
31969                                  <1>       ;              DL = Destination file found status
31970                                  <1>       ;              EAX = 0
31971                                  <1>       ;   Command Interpreter phase 2 enter ->
31972                                  <1>       ;              AL = 2 -> Continue from the last position
31973                                  <1>       ;              AH =
31974                                  <1>       ;   Phase 3 -> Load source file or use read/write cluster method
31975                                  <1>       ;   Phase 4 -> Create destination file if it is not found
31976                                  <1>       ;   Phase 5 -> Open destination file
31977                                  <1>       ;   Phase 6 -> Read from source and write to destination
31978                                  <1>       ;   Phase 7 -> Unload source file, if it is loaded at memory
31979                                  <1>       ;       cf = 1 causes to return before the phase 7
31980                                  <1>       ;              but loaded file will be unloaded
31981                                  <1>       ;             (allocated memory block will be deallocated)
31982                                  <1>       ;
31983                                  <1>       ; INPUT ->
31984                                  <1>       ;      ESI = Source File Pathname (Asciiz)
31985                                  <1>       ;       EDI = Destination File Pathname (Asciiz)
31986                                  <1>       ;       AL = 0 --> Interrupt (System call)
31987                                  <1>       ;       AL > 0 --> Command Interpreter (Question)
31988                                  <1>       ;       AL = 1 --> Question Phase
31989                                  <1>       ;       AL = 2 --> Progress Phase
31990                                  <1>       ;
31991                                  <1>       ; OUTPUT ->
31992                                  <1>       ;      cf = 0 -> OK
31993                                  <1>       ;      EAX = Destination file first cluster
31994                                  <1>       ;
31995                                  <1>       ;        CL > 0 if there is file reading error before EOF
31996                                  <1>       ;             (incomplete copy)
31997                                  <1>       ;        CH > 0 if file is (full) loaded at memory
31998                                  <1>       ;
31999                                  <1>       ;      cf = 1 -> Error code in AL (EAX)
32000                                  <1>       ;
32001                                  <1>       ; (EBX, ECX, ESI, EDI register contents will be changed)
32002                                  <1>
32003                                  <1>
32004 0000AD6A 3C02                    <1>       cmp    al, 2
32005 0000AD6C 0F845A020000            <1>       je     csftdf2_check_cdrv
32006                                  <1>
32007                                  <1> ; Phase 1
32008                                  <1>
32009 0000AD72 A2[A05A0100]            <1>       mov    byte [copy_cmd_phase], al
32010                                  <1>
32011 0000AD77 57                      <1>       push   edi ; *
32012                                  <1>
32013                                  <1> csftdf_parse_sf_path:
32014 0000AD78 BF[78590100]            <1>       mov    edi, SourceFile_Drv
32015 0000AD7D E884F4FFFF              <1>       call   parse_path_name
32016 0000AD82 721C                    <1>       jc     short csftdf_parse_sf_path_failed
32017                                  <1>
32018                                  <1> csftdf_parse_df_path:
32019 0000AD84 5E                      <1>       pop    esi ; * (pushed edi)
32020                                  <1>
32021                                  <1> csftdf_sf_check_filename_exists:
32022 0000AD85 803D[BA590100]21        <1>       cmp    byte [SourceFile_Name], 21h
```

```
32023 0000AD8C 7215              <1>       jb    short csftdf_sf_file_not_found_error
32024                            <1>
32025 0000AD8E BF[F8590100]      <1>       mov   edi, DestinationFile_Drv
32026 0000AD93 E86EF4FFFF        <1>       call  parse_path_name
32027 0000AD98 7310              <1>       jnc   short csftdf_check_sf_cdrv
32028                            <1>
32029 0000AD9A 3C01              <1>       cmp   al, 1 ; File or directory name is not existing
32030 0000AD9C 760C              <1>       jna   short csftdf_check_sf_cdrv
32031                            <1>
32032                            <1> csftdf_parse_df_path_failed:
32033 0000AD9E F9                <1>       stc
32034                            <1> csftdf_sf_error_retn:
32035 0000AD9F C3                <1>       retn
32036                            <1>
32037                            <1> csftdf_parse_sf_path_failed:
32038 0000ADA0 5F                <1>       pop   edi ; *
32039 0000ADA1 EBFC              <1>       jmp   short csftdf_sf_error_retn
32040                            <1>
32041                            <1> csftdf_sf_file_not_found_error:
32042 0000ADA3 B802000000        <1>       mov   eax, 2 ; File not found
32043 0000ADA8 EBF5              <1>       jmp   short csftdf_sf_error_retn
32044                            <1>
32045                            <1> csftdf_check_sf_cdrv:
32046 0000ADAA 8A3D[8E4E0100]    <1>       mov   bh, [Current_Drv]
32047                            <1>
32048 0000ADB0 883D[A35A0100]    <1>       mov   [csftdf_cdrv], bh ; 23/03/2016
32049                            <1>
32050 0000ADB6 8A15[78590100]    <1>       mov   dl, [SourceFile_Drv]
32051 0000ADBC 38FA              <1>       cmp   dl, bh ; byte [Current_Drv]
32052 0000ADBE 7407              <1>       je    short csftdf_sf_check_directory
32053                            <1>
32054 0000ADC0 E898BEFFFF        <1>       call  change_current_drive
32055 0000ADC5 72D8              <1>       jc    short csftdf_sf_error_retn
32056                            <1>
32057                            <1> csftdf_sf_check_directory:
32058 0000ADC7 BE[79590100]      <1>       mov   esi, SourceFile_Directory
32059 0000ADCC 803E20            <1>       cmp   byte [esi], 20h
32060 0000ADCF 760F              <1>       jna   short csftdf_find_sf
32061                            <1>
32062                            <1> csftdf_sf_change_directory:
32063 0000ADD1 FE05[98020100]    <1>       inc   byte [Restore_CDIR]
32064 0000ADD7 30E4              <1>       xor   ah, ah ; CD_COMMAND sign -> 0
32065 0000ADD9 E814EEFFFF        <1>       call  change_current_directory
32066 0000ADDE 72BF              <1>       jc    short csftdf_sf_error_retn
32067                            <1>
32068                            <1> ;csftdf_sf_change_prompt_dir_string:
32069                            <1> ;     call  change_prompt_dir_string
32070                            <1>
32071                            <1> csftdf_find_sf:
32072 0000ADE0 BE[BA590100]      <1>       mov   esi, SourceFile_Name
32073 0000ADE5 66B80018          <1>       mov   ax, 1800h ; Except volume label and dirs
32074 0000ADE9 E801D2FFFF        <1>       call  find_first_file
32075 0000ADEE 72AF              <1>       jc    short csftdf_sf_error_retn
32076                            <1>
32077                            <1> csftdf_sf_ambgfn_check:
32078 0000ADF0 6621D2            <1>       and   dx, dx ; Ambiguous filename chars used sign (DX>0)
32079 0000ADF3 7407              <1>       jz    short csftdf_sf_found
32080                            <1>
32081                            <1> csftdf_ambiguous_file_name_error:
32082 0000ADF5 B802000000        <1>       mov   eax, 2 ; File not found error
32083 0000ADFA F9                <1>       stc
32084 0000ADFB C3                <1>       retn
32085                            <1>
32086                            <1> csftdf_sf_found:
32087 0000ADFC A3[A45A0100]      <1>       mov   [csftdf_filesize], eax
32088                            <1>
32089 0000AE01 09C0              <1>       or    eax, eax
32090 0000AE03 7507              <1>       jnz   short csftdf_set_source_file_direntry
32091                            <1>
32092                            <1> csftdf_sf_file_size_zero:
32093 0000AE05 B814000000        <1>       mov   eax, 20 ; TRDOS zero length (file size) error
32094 0000AE0A F9                <1>       stc
32095 0000AE0B C3                <1>       retn
32096                            <1>
32097                            <1> csftdf_set_source_file_direntry:
32098 0000AE0C BE[84580100]      <1>       mov   esi, FindFile_DirEntry
32099 0000AE11 BF[CA590100]      <1>       mov   edi, SourceFile_DirEntry
32100 0000AE16 B908000000        <1>       mov   ecx, 8
32101 0000AE1B F3A5              <1>       rep   movsd
32102                            <1>
32103                            <1> csftdf_sf_restore_cdrv:
32104                            <1>       ; 22/03/2016
32105 0000AE1D 8A15[A35A0100]    <1>       mov   dl, [csftdf_cdrv]
32106 0000AE23 3A15[8E4E0100]    <1>       cmp   dl, [Current_Drv]
32107 0000AE29 7407              <1>       je    short csftdf_sf_restore_cdir
32108 0000AE2B E82DBEFFFF        <1>       call  change_current_drive
32109 0000AE30 724F              <1>       jc    short csftdf_df_error_retn ; 30/03/2016
32110                            <1>
32111                            <1> csftdf_sf_restore_cdir:
32112 0000AE32 803D[98020100]00  <1>       cmp   byte [Restore_CDIR], 0
32113 0000AE39 7612              <1>       jna   short csftdf_df_check_filename_exists
32114 0000AE3B 29C0              <1>       sub   eax, eax
32115 0000AE3D BE00010900        <1>       mov   esi, Logical_DOSDisks
32116 0000AE42 88D4              <1>       mov   ah, dl ; byte [csftdf_cdrv]
32117 0000AE44 01C6              <1>       add   esi, eax
32118 0000AE46 E8C9BEFFFF        <1>       call  restore_current_directory
32119 0000AE4B 7234              <1>       jc    short csftdf_df_error_retn
32120                            <1>
32121                            <1> csftdf_df_check_filename_exists:
32122 0000AE4D 803D[3A5A0100]20  <1>       cmp   byte [DestinationFile_Name], 20h
32123 0000AE54 7716              <1>       ja    short csftdf_check_df_cdrv
32124                            <1>
```

```
32125                                  <1> csftdf_copy_sf_name:
32126 0000AE56 BF[3A5A0100]            <1>       mov    edi, DestinationFile_Name
32127 0000AE5B BE[BA590100]            <1>       mov    esi, SourceFile_Name
32128 0000AE60 B10C                    <1>       mov    cl, 12
32129                                  <1>
32130                                  <1> csftdf_df_copy_sf_name_loop:
32131 0000AE62 AC                      <1>       lodsb
32132 0000AE63 AA                      <1>       stosb
32133 0000AE64 08C0                    <1>       or     al, al
32134 0000AE66 7404                    <1>       jz     short csftdf_check_df_cdrv
32135 0000AE68 FEC9                    <1>       dec    cl
32136 0000AE6A 75F6                    <1>       jnz    csftdf_df_copy_sf_name_loop
32137                                  <1>
32138                                  <1> csftdf_check_df_cdrv:
32139 0000AE6C 8A15[F8590100]          <1>       mov    dl, [DestinationFile_Drv]
32140 0000AE72 3A15[8E4E0100]          <1>       cmp    dl, [Current_Drv]
32141 0000AE78 7408                    <1>       je     short csftdf_df_check_directory
32142                                  <1>
32143 0000AE7A E8DEBDFFFF              <1>       call   change_current_drive
32144 0000AE7F 7301                    <1>       jnc    short csftdf_df_check_directory
32145                                  <1>
32146                                  <1> csftdf_df_error_retn:
32147 0000AE81 C3                      <1>       retn
32148                                  <1>
32149                                  <1> csftdf_df_check_directory:
32150 0000AE82 BE[F9590100]            <1>       mov    esi, DestinationFile_Directory
32151 0000AE87 803E20                  <1>         cmp     byte [esi], 20h
32152 0000AE8A 760F                    <1>       jna    short csftdf_find_df
32153                                  <1>
32154                                  <1> csftdf_df_change_directory:
32155 0000AE8C FE05[98020100]          <1>       inc    byte [Restore_CDIR]
32156 0000AE92 28E4                    <1>       sub    ah, ah ; CD_COMMAND sign -> 0
32157 0000AE94 E859EDFFFF              <1>       call   change_current_directory
32158 0000AE99 72E6                    <1>       jc     short csftdf_df_error_retn
32159                                  <1>
32160                                  <1> ;csftdf_df_change_prompt_dir_string:
32161                                  <1> ;     call   change_prompt_dir_string
32162                                  <1>
32163                                  <1> csftdf_find_df:
32164                                  <1>       ; 23/03/2016
32165 0000AE9B 29DB                    <1>       sub    ebx, ebx
32166 0000AE9D 8A3D[F8590100]          <1>       mov    bh,  [DestinationFile_Drv]
32167 0000AEA3 81C300010900            <1>       add    ebx, Logical_DOSDisks
32168 0000AEA9 891D[D05A0100]          <1>       mov    [csftdf_df_drv_dt], ebx
32169                                  <1>
32170 0000AEAF BE[3A5A0100]            <1>       mov    esi, DestinationFile_Name
32171 0000AEB4 6631C0                  <1>       xor    ax, ax
32172                                  <1>              ; DestinationFile_AttributesMask -> any/zero
32173 0000AEB7 E833D1FFFF              <1>       call   find_first_file
32174 0000AEBC 7218                    <1>       jc     short csftdf_df_check_error_code
32175                                  <1>
32176                                  <1> csftdf_df_ambgfn_check:
32177 0000AEBE 6609D2                  <1>       or     dx, dx ; Ambiguous filename chars used sign (DX>0)
32178 0000AEC1 752A                    <1>       jnz    short csftdf_df_error_inv_fname
32179                                  <1>
32180                                  <1> csftdf_df_found:
32181 0000AEC3 C605[A25A0100]01        <1>       mov    byte [DestinationFileFound], 1
32182                                  <1>       ; 17/10/2016 (cl -> bl)
32183 0000AECA 80E31F                  <1>       and    bl, 1Fh ; Attributes, D-V-S-H-R
32184 0000AECD 745F                    <1>       jz     short csftdf_df_save_first_cluster
32185                                  <1>
32186                                  <1> csftdf_df_permission_denied_retn:
32187 0000AECF B805000000              <1>       mov    eax, 05h ; Access/Permisson denied.
32188                                  <1> csftdf_df_error_stc_retn:
32189 0000AED4 F9                      <1>       stc
32190 0000AED5 C3                      <1>       retn
32191                                  <1>
32192                                  <1> csftdf_df_check_error_code:
32193                                  <1>       ;cmp   eax, 2
32194 0000AED6 3C02                    <1>       cmp    al, 2
32195 0000AED8 75FA                    <1>       jne    short csftdf_df_error_stc_retn
32196                                  <1>
32197 0000AEDA C605[A25A0100]00        <1>       mov    byte [DestinationFileFound], 0
32198                                  <1>
32199                                  <1>       ; 15/10/2016
32200 0000AEE1 BE[74580100]            <1>       mov    esi, FindFile_Name ; *
32201 0000AEE6 E8C4D4FFFF              <1>       call   check_filename
32202 0000AEEB 7307                    <1>       jnc    short csftdf_df_valid_fname
32203                                  <1> csftdf_df_error_inv_fname: ; 'invalid file name !'
32204 0000AEED B81A000000              <1>       mov    eax, ERR_INV_FILE_NAME  ; 26
32205 0000AEF2 F9                      <1>       stc
32206 0000AEF3 C3                      <1>       retn
32207                                  <1>
32208                                  <1> csftdf_df_valid_fname:
32209                                  <1>       ; 21/03/2016
32210                                  <1>       ; (Capitalized file name)
32211                                  <1>       ;mov   esi, FindFile_Name ; * ; 15/10/2016
32212 0000AEF4 BF[3A5A0100]            <1>       mov    edi, DestinationFile_Name
32213 0000AEF9 A5                      <1>       movsd
32214 0000AEFA A5                      <1>       movsd
32215 0000AEFB A5                      <1>       movsd
32216                                  <1>       ;movsb
32217                                  <1>
32218                                  <1> csftdf_check_disk_free_size_0:
32219 0000AEFC A1[E6590100]            <1>       mov    eax, [SourceFile_DirEntry+DirEntry_FileSize]
32220                                  <1>
32221                                  <1> csftdf_check_disk_free_size_1:
32222                                  <1>       ;sub   ebx, ebx
32223                                  <1>       ;mov   esi, Logical_DOSDisks
32224                                  <1>       ;mov   bh,  [DestinationFile_Drv]
32225                                  <1>       ;add   esi, ebx
32226                                  <1>
```

```
32227 0000AF01 8B35[D05A0100]      <1>       mov    esi, [csftdf_df_drv_dt] ; 23/03/2016
32228                              <1>
32229 0000AF07 0FB74E11            <1>       movzx  ecx, word [esi+LD_BPB+BytesPerSec] ; 17, LD_BPB + 0Bh
32230 0000AF0B 01C8                <1>       add    eax, ecx
32231 0000AF0D 48                  <1>       dec    eax  ; file size (additional bytes) + 511 (round up)
32232                              <1> csftdf_check_disk_free_size_3: ; 16/03/2016
32233 0000AF0E 29D2                <1>       sub    edx, edx
32234 0000AF10 F7F1                <1>       div    ecx ; bytes per sector
32235                              <1>
32236                              <1> csftdf_check_disk_free_size:
32237 0000AF12 3B4674              <1>       cmp    eax, [esi+LD_FreeSectors]
32238 0000AF15 0F8294000000        <1>         jb     csftdf_check_disk_free_size_ok
32239 0000AF1B 770A                <1>       ja     short csftdf_df_insufficient_disk_space
32240                              <1>
32241 0000AF1D 807E0300            <1>       cmp    byte [esi+LD_FATType], 0 ; FS needs FDT sector also.
32242 0000AF21 0F8788000000        <1>         ja     csftdf_check_disk_free_size_ok
32243                              <1>
32244                              <1> csftdf_df_insufficient_disk_space:
32245 0000AF27 B827000000          <1>       mov    eax, 27h ; insufficient disk space
32246 0000AF2C EBA6                <1>       jmp    short csftdf_df_error_stc_retn
32247                              <1>
32248                              <1> csftdf_df_save_first_cluster:
32249                              <1>       ; ESI = FindFile_DirEntry (for the old destination file)
32250                              <1>       ; EAX = Old destination file size
32251                              <1>       ; 24/03/2016
32252                              <1>       ; EDI = Directory entry address (within Dir Buffer boundaries)
32253 0000AF2E 81EF00000800        <1>       sub    edi, Directory_Buffer  ; (<65536)
32254 0000AF34 66C1EF05            <1>       shr    di, 5 ; Convert entry offset to entry index/number
32255 0000AF38 66893D[725A0100]    <1>       mov    [DestinationFile_DirEntryNumber], di ; (<2048)
32256                              <1>
32257                              <1> csftdf_df_check_sf_df_fcluster:
32258 0000AF3F 668B5614            <1>       mov    dx, [esi+DirEntry_FstClusHI]
32259 0000AF43 C1E210              <1>       shl    edx, 16
32260 0000AF46 668B561A            <1>       mov    dx, [esi+DirEntry_FstClusLO]
32261 0000AF4A 8915[B45A0100]      <1>       mov    [csftdf_df_cluster], edx
32262                              <1> csftdf_df_check_sf_df_fcluster_1:
32263 0000AF50 668B15[DE590100]    <1>       mov    dx, [SourceFile_DirEntry+DirEntry_FstClusHI]
32264 0000AF57 C1E210              <1>       shl    edx, 16
32265 0000AF5A 668B15[E4590100]    <1>       mov    dx, [SourceFile_DirEntry+DirEntry_FstClusLO]
32266 0000AF61 3B15[B45A0100]      <1>       cmp    edx, [csftdf_df_cluster]
32267 0000AF67 7512                <1>       jne    short csftdf_df_check_sf_df_fcluster_ok
32268                              <1> csftdf_df_check_sf_df_drv:
32269 0000AF69 8A15[78590100]      <1>       mov    dl, [SourceFile_Drv]
32270 0000AF6F 3A15[F8590100]      <1>       cmp    dl, [DestinationFile_Drv]
32271 0000AF75 7504                <1>       jne    short csftdf_df_check_sf_df_fcluster_ok
32272                              <1>
32273                              <1>       ; source and destination files are same !
32274                              <1>       ; (they have same first cluster value on same logical disk)
32275                              <1>
32276 0000AF77 31C0                <1>       xor    eax, eax ; mov eax, 0 -> Bad command or file name !
32277 0000AF79 F9                  <1>       stc
32278 0000AF7A C3                  <1>       retn
32279                              <1>
32280                              <1> csftdf_df_check_sf_df_fcluster_ok:
32281                              <1> csftdf_df_move_findfile_struct:
32282                              <1>       ; mov  esi, FindFile_DirEntry
32283 0000AF7B BF[4A5A0100]        <1>       mov    edi, DestinationFile_DirEntry
32284 0000AF80 B908000000          <1>       mov    ecx, 8
32285 0000AF85 F3A5                <1>       rep    movsd
32286                              <1>
32287                              <1> csftdf_check_disk_free_size_2:
32288 0000AF87 89C2                <1>       mov    edx, eax ; Old destination file size
32289                              <1>
32290                              <1>       ;mov   eax, [SourceFile_DirEntry+DirEntry_FileSize]
32291 0000AF89 A1[A45A0100]        <1>       mov    eax, [csftdf_filesize] ; 23/03/2016
32292                              <1>
32293                              <1>       ;;sub  ecx, ecx ; 0
32294                              <1>       ;mov   esi, Logical_DOSDisks
32295                              <1>       ;mov   ch,  [DestinationFile_Drv]
32296                              <1>       ;add   esi, ecx
32297                              <1>       ;
32298                              <1>       ;mov   [csftdf_df_drv_dt], esi
32299                              <1>
32300 0000AF8E 8B35[D05A0100]      <1>       mov    esi, [csftdf_df_drv_dt] ; 23/03/2016
32301                              <1>
32302 0000AF94 668B4E11            <1>       mov    cx, [esi+LD_BPB+BytesPerSec] ; 17, LD_BPB + 0Bh
32303 0000AF98 01CA                <1>       add    edx, ecx ; + 512
32304 0000AF9A 01C8                <1>       add    eax, ecx ; + 512
32305 0000AF9C 4A                  <1>       dec    edx ; old file size + 511 (round up)
32306 0000AF9D 48                  <1>       dec    eax ; new file size + 511 (round up)
32307 0000AF9E F7D9                <1>       neg    ecx ; -512 ; 0FFFFFE00h
32308 0000AFA0 21CA                <1>       and    edx, ecx ; = old sector count * 512
32309 0000AFA2 21C8                <1>       and    eax, ecx ; = new sector count * 512
32310                              <1>
32311 0000AFA4 29D0                <1>       sub    eax, edx ; new file size - old file size (on disk)
32312 0000AFA6 7607                <1>       jna    short csftdf_check_disk_free_size_ok
32313                              <1>
32314 0000AFA8 F7D9                <1>       neg    ecx ; 512 (bytes per sector) ; 200h
32315                              <1>       ; check free space for additional sectors
32316                              <1>       ; eax = number of additional sectors * bytes per sector
32317                              <1>       ; esi = Logical DOS drive number (of destination disk)
32318 0000AFAA E95FFFFFFF          <1>         jmp    csftdf_check_disk_free_size_3
32319                              <1>
32320                              <1> csftdf_check_disk_free_size_ok:
32321                              <1>       ; 18/03/2016
32322                              <1> csftdf_df_check_copy_cmd_phase:
32323 0000AFAF A0[A05A0100]        <1>       mov    al, [copy_cmd_phase]
32324 0000AFB4 3C01                <1>       cmp    al, 1
32325 0000AFB6 7514                <1>       jne    short csftdf2_check_cdrv
32326                              <1>
32327 0000AFB8 31C0                <1>       xor    eax, eax
32328 0000AFBA A2[A05A0100]        <1>       mov    [copy_cmd_phase], al ; 0
```

```
32329                                  <1>
32330 0000AFBF 8A15[A25A0100]          <1>       mov    dl, [DestinationFileFound]
32331 0000AFC5 8A35[D5590100]          <1>       mov    dh, [SourceFile_DirEntry+11] ; Attributes
32332                                  <1>
32333                                  <1> csftdf_return:
32334 0000AFCB C3                      <1>       retn
32335                                  <1>
32336                                  <1> ; Phase 2
32337                                  <1>
32338                                  <1> csftdf2_check_cdrv:
32339                                  <1>       ; 18/03/2016
32340                                  <1>       ; Here, destination drive and directory are ready !
32341                                  <1>       ; (checking/restoring is not needed)
32342                                  <1>       ; (Since at the end of the phase 1)
32343                                  <1>
32344                                  <1> ;     mov    dl, [DestinationFile_Drv]
32345                                  <1> ;     cmp    dl, [Current_Drv]
32346                                  <1> ;     je     short csftdf2_df_check_directory
32347                                  <1> ;
32348                                  <1> ;     call   change_current_drive
32349                                  <1> ;     jc     short csftdf2_read_error
32350                                  <1> ;
32351                                  <1> ;csftdf2_df_check_directory:
32352                                  <1> ;     mov    esi, DestinationFile_Directory
32353                                  <1> ;     cmp    byte [esi], 20h
32354                                  <1> ;     jna    short csftdf2_df_check_found_or_not
32355                                  <1> ;
32356                                  <1> ;csftdf2_df_change_directory:
32357                                  <1> ;     inc    byte [Restore_CDIR]
32358                                  <1> ;     xor    ah, ah ; CD_COMMAND sign -> 0
32359                                  <1> ;     call   change_current_directory
32360                                  <1> ;     jc     short csftdf2_stc_return
32361                                  <1> ;
32362                                  <1> ;;csftdf2_df_change_prompt_dir_string:
32363                                  <1> ;;    call   change_prompt_dir_string
32364                                  <1>
32365                                  <1> csftdf2_df_check_found_or_not:
32366                                  <1>       ; 21/03/2016
32367 0000AFCC 803D[A25A0100]00        <1>       cmp    byte [DestinationFileFound], 0
32368 0000AFD3 7739                    <1>       ja     short csftdf2_set_sf_percentage
32369                                  <1>
32370                                  <1> csftdf2_create_file:
32371 0000AFD5 BE[3A5A0100]            <1>       mov    esi, DestinationFile_Name
32372 0000AFDA A1[A45A0100]            <1>       mov    eax, [csftdf_filesize]
32373 0000AFDF 30C9                    <1>       xor    cl, cl ; 0
32374                                  <1>
32375 0000AFE1 31DB                    <1>       xor    ebx, ebx ; 0
32376 0000AFE3 4B                      <1>       dec    ebx ; 0FFFFFFFFh
32377                                  <1>
32378                                  <1>       ; INPUT ->
32379                                  <1>       ;      EAX -> File Size
32380                                  <1>       ;      ESI = ASCIIZ File name
32381                                  <1>       ;       CL = File attributes
32382                                  <1>       ;      EBX = FFFFFFFFh -> empty file sign for FAT fs
32383                                  <1>       ;      EBX <> FFFFFFFFh -> use file size for FAT fs
32384                                  <1>       ;
32385                                  <1>       ; OUTPUT ->
32386                                  <1>       ;      EAX = New file's first cluster
32387                                  <1>       ;      ESI = Logical Dos Drv Descr. Table Addr.
32388                                  <1>       ;      EBX = CreateFile_Size address
32389                                  <1>       ;      ECX = Sectors per cluster (<256)
32390                                  <1>       ;      EDX = Directory Entry Index/Number (<65536)
32391                                  <1>       ;
32392                                  <1>       ;      cf = 1 -> error code in AL (EAX)
32393                                  <1>
32394 0000AFE4 E8EC050000              <1>       call   create_file
32395                                  <1>       ;pop   esi
32396 0000AFE9 0F82A3050000            <1>         jc     csftdf2_rw_error
32397                                  <1>
32398                                  <1> csftdf2_create_file_OK:
32399 0000AFEF A3[B45A0100]            <1>       mov    [csftdf_df_cluster], eax
32400                                  <1>
32401                                  <1>       ; 24/03/2016
32402 0000AFF4 668915[725A0100]        <1>       mov    [DestinationFile_DirEntryNumber], dx
32403                                  <1>
32404                                  <1>       ; 21/03/2016
32405 0000AFFB BE00000800              <1>       mov    esi, Directory_Buffer
32406 0000B000 C1E205                  <1>       shl    edx, 5 ; 32 * index number
32407 0000B003 01D6                    <1>       add    esi, edx
32408 0000B005 BF[4A5A0100]            <1>       mov    edi, DestinationFile_DirEntry
32409 0000B00A B108                    <1>       mov    cl, 8 ; 32 bytes
32410 0000B00C F3A5                    <1>       rep    movsd
32411                                  <1>
32412                                  <1> csftdf2_set_sf_percentage:
32413                                  <1>       ; 17/03/2016
32414 0000B00E 31C0                    <1>       xor    eax, eax
32415 0000B010 A2[C85A0100]            <1>       mov    [csftdf_percentage], al ; 0, reset
32416                                  <1>
32417 0000B015 A3[C05A0100]            <1>       mov    [csftdf_sf_rbytes], eax ; 0, reset
32418 0000B01A A3[C45A0100]            <1>       mov    [csftdf_df_wbytes], eax ; 0, reset
32419                                  <1>
32420 0000B01F 8A25[78590100]          <1>       mov    ah, [SourceFile_Drv]
32421 0000B025 BE00010900              <1>       mov    esi, Logical_DOSDisks
32422 0000B02A 01C6                    <1>       add    esi, eax
32423                                  <1>
32424 0000B02C 8935[CC5A0100]          <1>       mov    [csftdf_sf_drv_dt], esi ; 23/03/2016
32425                                  <1>
32426 0000B032 668B15[DE590100]        <1>       mov    dx, [SourceFile_DirEntry+DirEntry_FstClusHI]
32427 0000B039 C1E210                  <1>       shl    edx, 16
32428 0000B03C 668B15[E4590100]        <1>       mov    dx, [SourceFile_DirEntry+DirEntry_FstClusLO]
32429 0000B043 8915[B05A0100]          <1>       mov    [csftdf_sf_cluster], edx
32430                                  <1>
```

```
32431                                  <1>       ; 16/03/2016
32432                                  <1>       ; Note: Singlix FS boot sector parameters (for cluster
32433                                  <1>       ;     related calculations) has same offset
32434                                  <1>       ;     values from LD_BPB as in FAT file system.
32435                                  <1>       ;     [esi+LD_BPB+SecPerClust] is 1 for Singlix FS.
32436                                  <1>       ;
32437 0000B049 0FB64E13               <1>       movzx  ecx, byte [esi+LD_BPB+SecPerClust]
32438 0000B04D 880D[F6590100]         <1>       mov    [SourceFile_SecPerClust], cl
32439                                  <1>
32440                                  <1>       ; 17/03/2016
32441 0000B053 386E03                 <1>       cmp    [esi+LD_FATType], ch ; 0
32442 0000B056 7707                   <1>       ja     short csftdf2_set_sf_percent_rsize1
32443                                  <1>
32444 0000B058 B800000100             <1>       mov    eax, 65536 ; read/write buffer size for Singlix FS
32445 0000B05D EB06                   <1>       jmp    short csftdf2_set_sf_percent_rsize2
32446                                  <1>
32447                                  <1> csftdf2_set_sf_percent_rsize1:
32448 0000B05F 668B4611               <1>       mov    ax, [esi+LD_BPB+BytesPerSec]
32449 0000B063 F7E1                   <1>       mul    ecx
32450                                  <1>       ;sub   edx, edx
32451                                  <1> csftdf2_set_sf_percent_rsize2:
32452 0000B065 A3[B85A0100]           <1>       mov    [csftdf_r_size], eax
32453                                  <1>
32454                                  <1> csftdf2_set_df_percentage:
32455                                  <1>       ;sub   eax, eax
32456                                  <1>       ;mov   ah, [DestinationFile_Drv]
32457                                  <1>       ;mov   edi, Logical_DOSDisks
32458                                  <1>       ;add   edi, eax
32459                                  <1>       ;mov   [csftdf_df_drv_dt], edi ; 17/03/2016
32460                                  <1>
32461 0000B06A 8B3D[D05A0100]         <1>       mov    edi, [csftdf_df_drv_dt] ; 23/03/2016
32462                                  <1>
32463                                  <1>       ; 16/03/2016
32464                                  <1>       ; Note: Singlix FS boot sector parameters (for cluster
32465                                  <1>       ;     related calculations) has same offset
32466                                  <1>       ;     values from LD_BPB as in FAT file system.
32467                                  <1>       ;     [edi+LD_BPB+SecPerClust] is 1 for Singlix FS.
32468                                  <1>       ;
32469                                  <1>       ;movzx ecx, byte [edi+LD_BPB+SecPerClust]
32470 0000B070 8A4F13                 <1>       mov    cl, [edi+LD_BPB+SecPerClust]
32471 0000B073 880D[765A0100]         <1>       mov    [DestinationFile_SecPerClust], cl
32472                                  <1>
32473                                  <1>       ; 17/03/2016
32474 0000B079 386F03                 <1>       cmp    [edi+LD_FATType], ch ; 0
32475 0000B07C 7707                   <1>       ja     short csftdf2_set_df_percent_wsize1
32476                                  <1>
32477 0000B07E B800000100             <1>       mov    eax, 65536 ; read/write buffer size for Singlix FS
32478 0000B083 EB06                   <1>       jmp    short csftdf2_set_df_percent_wsize2
32479                                  <1>
32480                                  <1> csftdf2_set_df_percent_wsize1:
32481 0000B085 0FB74711               <1>       movzx  eax, word [edi+LD_BPB+BytesPerSec]
32482 0000B089 F7E1                   <1>       mul    ecx
32483                                  <1>       ;sub   edx, edx
32484                                  <1> csftdf2_set_df_percent_wsize2:
32485 0000B08B A3[BC5A0100]           <1>       mov    [csftdf_w_size], eax
32486                                  <1>
32487 0000B090 A1[A45A0100]           <1>       mov    eax, [csftdf_filesize]
32488                                  <1>
32489 0000B095 3D00000100             <1>       cmp    eax, 65536 ; 64KB   ; small file
32490 0000B09A 721F                   <1>       jb     short csftdf2_load_file ; do not display percentage
32491                                  <1>
32492                                  <1> csftdf2_reset_wf_percent_ptr_chk_64k:
32493 0000B09C B201                   <1>       mov    dl, 1 ; 25/03/2016
32494                                  <1>
32495 0000B09E 3D00000400             <1>       cmp    eax, 65536*4 ; 256KB
32496 0000B0A3 7310                   <1>       jnb    short csftdf2_enable_percentage_display ; big file
32497                                  <1>
32498                                  <1>       ; 64-128KB file size for floppy disks
32499 0000B0A5 3815[78590100]         <1>       cmp    byte [SourceFile_Drv], dl ; 1 ; read from floppy disk ?
32500 0000B0AB 7608                   <1>       jna    short csftdf2_enable_percentage_display
32501                                  <1>
32502 0000B0AD 3815[F8590100]         <1>       cmp    byte [DestinationFile_Drv], dl ; 1 ; write to floppy disk ?
32503 0000B0B3 7706                   <1>       ja     short csftdf2_load_file
32504                                  <1>
32505                                  <1> csftdf2_enable_percentage_display:
32506 0000B0B5 8815[C85A0100]         <1>       mov    [csftdf_percentage], dl ; 1
32507                                  <1>
32508                                  <1> csftdf2_load_file:
32509                                  <1>       ; 13/05/2016
32510                                  <1>       ; 19/03/2016
32511                                  <1>       ; 18/03/2016
32512                                  <1>       ; 17/03/2016
32513 0000B0BB B40F                   <1>       mov    ah, 0Fh
32514 0000B0BD E8D863FFFF             <1>       call   _int10h
32515                                  <1>       ; 13/05/2016
32516 0000B0C2 883D[C95A0100]         <1>       mov    [csftdf_videopage], bh ; active video page
32517 0000B0C8 B403                   <1>       mov    ah, 03h
32518 0000B0CA E8CB63FFFF             <1>       call   _int10h
32519 0000B0CF 668915[CA5A0100]       <1>       mov    [csftdf_cursorpos], dx
32520                                  <1>
32521 0000B0D6 29C0                   <1>       sub    eax, eax
32522 0000B0D8 A2[A15A0100]           <1>       mov    [csftdf_rw_err], al ; 0
32523                                  <1>
32524                                  <1> ; ///
32525                                  <1> csftdf_sf_amb: ; 15/03/2016
32526 0000B0DD 8B0D[A45A0100]         <1>       mov    ecx, [csftdf_filesize]    ; 23/03/2016
32527                                  <1>
32528                                  <1>       ; TRDOS 386 (TRDOS v2.0)
32529                                  <1>       ; Allocate contiguous memory block for loading the file
32530                                  <1>
32531                                  <1>       ;mov   ecx, [SourceFile_DirEntry+DirEntry_FileSize]
32532                                  <1>
```

```
32533                                  <1>       ;sub   eax, eax ; First free memory aperture
32534                                  <1>
32535                                  <1>       ; eax = 0 (Allocate memory from the beginning)
32536                                  <1>       ; ecx = File (Allocation) size in bytes
32537                                  <1>
32538 0000B0E3 E83BA3FFFF             <1>       call   allocate_memory_block
32539 0000B0E8 7304                   <1>       jnc    short loc_check_sf_save_loading_parms
32540                                  <1>
32541 0000B0EA 29C0                   <1>       sub    eax, eax
32542 0000B0EC 29C9                   <1>       sub    ecx, ecx
32543                                  <1>
32544                                  <1> loc_check_sf_save_loading_parms:
32545 0000B0EE A3[A85A0100]           <1>       mov    [csftdf_sf_mem_addr], eax ; loading address
32546 0000B0F3 890D[AC5A0100]         <1>       mov    [csftdf_sf_mem_bsize], ecx ; block size
32547                                  <1> ; ///
32548                                  <1>       ; 19/03/2016
32549 0000B0F9 8B35[CC5A0100]         <1>       mov    esi, [csftdf_sf_drv_dt] ; logical dos drv desc. tbl.
32550                                  <1>
32551                                  <1>       ; 17/03/2016
32552 0000B0FF 09C0                   <1>       or     eax, eax ; contiguous free memory block address
32553 0000B101 0F845B010000           <1>         jz      csftdf2_read_sf_cluster
32554                                  <1>
32555                                  <1>       ; 18/03/2016
32556 0000B107 8B1D[A85A0100]         <1>       mov    ebx, [csftdf_sf_mem_addr] ; memory block address
32557                                  <1>
32558 0000B10D 807E0300               <1>       cmp    byte [esi+LD_FATType], 0
32559 0000B111 0F8605020000           <1>         jna     csftdf2_load_fs_file
32560                                  <1>
32561                                  <1> csftdf2_load_fat_file:
32562 0000B117 53                     <1>       push   ebx ; *
32563                                  <1>
32564                                  <1> csftdf2_load_fat_file_next:
32565 0000B118 BE[E7080100]           <1>       mov    esi, msg_reading
32566 0000B11D E83BB2FFFF             <1>       call   print_msg
32567                                  <1>
32568 0000B122 803D[C85A0100]00       <1>       cmp    byte [csftdf_percentage], 0
32569 0000B129 7605                   <1>       jna    short csftdf2_load_fat_file_1
32570                                  <1>
32571 0000B12B E87C000000             <1>       call   csftdf2_print_percentage ; 19/03/2016
32572                                  <1>
32573                                  <1> csftdf2_load_fat_file_1:
32574 0000B130 8B35[CC5A0100]         <1>       mov    esi, [csftdf_sf_drv_dt]
32575 0000B136 5B                     <1>       pop    ebx ; *
32576                                  <1>
32577                                  <1> csftdf2_load_fat_file_2:
32578 0000B137 E8B8000000             <1>       call   csftdf2_read_fat_file_sectors ; 19/03/2016
32579 0000B13C 0F8250040000           <1>         jc      csftdf2_rw_error ; eocc! or disk error!
32580                                  <1>
32581 0000B142 09D2                   <1>       or     edx, edx ; edx > 0 -> EOF
32582 0000B144 7520                   <1>       jnz    short csftdf2_load_fat_file_ok
32583                                  <1>
32584 0000B146 803D[C85A0100]00       <1>       cmp    byte [csftdf_percentage], 0
32585 0000B14D 76E8                   <1>       jna    short csftdf2_load_fat_file_2
32586                                  <1>
32587 0000B14F 53                     <1>       push   ebx ; *
32588                                  <1>
32589                                  <1>       ; Set cursor position
32590                                  <1>       ; AH= 02h, BH= Page Number, DH= Row, DL= Column
32591 0000B150 8A3D[C95A0100]         <1>       mov    bh, [csftdf_videopage]
32592 0000B156 668B15[CA5A0100]       <1>       mov    dx, [csftdf_cursorpos]
32593 0000B15D B402                   <1>       mov    ah, 2
32594 0000B15F E83663FFFF             <1>       call   _int10h
32595 0000B164 EBB2                   <1>       jmp    short csftdf2_load_fat_file_next
32596                                  <1>
32597                                  <1> csftdf2_load_fat_file_ok:
32598 0000B166 803D[C85A0100]00       <1>       cmp    byte [csftdf_percentage], 0
32599 0000B16D 0F8651020000           <1>         jna     csftdf2_save_file ; 25/03/2016
32600                                  <1>
32601                                  <1>       ; "Reading... 100%"
32602 0000B173 BF[FF080100]           <1>       mov    edi, percentagestr
32603 0000B178 B031                   <1>       mov    al, '1'
32604 0000B17A AA                     <1>       stosb
32605 0000B17B B030                   <1>       mov    al, '0'
32606 0000B17D AA                     <1>       stosb
32607 0000B17E AA                     <1>       stosb
32608                                  <1>
32609 0000B17F 8A3D[C95A0100]         <1>       mov    bh, [csftdf_videopage]
32610 0000B185 668B15[CA5A0100]       <1>       mov    dx, [csftdf_cursorpos]
32611 0000B18C B402                   <1>       mov    ah, 2
32612 0000B18E E80763FFFF             <1>       call   _int10h
32613                                  <1>
32614 0000B193 BE[E7080100]           <1>       mov    esi, msg_reading
32615 0000B198 E8C0B1FFFF             <1>       call   print_msg
32616                                  <1>
32617 0000B19D BE[FF080100]           <1>       mov    esi, percentagestr
32618 0000B1A2 E8B6B1FFFF             <1>       call   print_msg
32619                                  <1>
32620 0000B1A7 E918020000             <1>         jmp     csftdf2_save_file ; 25/03/2016
32621                                  <1>
32622                                  <1> csftdf2_print_percentage:
32623                                  <1>       ; 19/03/2016
32624                                  <1>       ; 18/03/2016
32625 0000B1AC B020                   <1>       mov    al, 20h
32626 0000B1AE BF[FF080100]           <1>       mov    edi, percentagestr
32627 0000B1B3 AA                     <1>       stosb
32628 0000B1B4 AA                     <1>       stosb
32629 0000B1B5 A1[C05A0100]           <1>       mov    eax, [csftdf_sf_rbytes]
32630 0000B1BA BA64000000             <1>       mov    edx, 100
32631 0000B1BF F7E2                   <1>       mul    edx
32632 0000B1C1 8B0D[A45A0100]         <1>       mov    ecx, [csftdf_filesize]
32633 0000B1C7 F7F1                   <1>       div    ecx
32634 0000B1C9 B10A                   <1>       mov    cl, 10
```

```
32635 0000B1CB F6F1              <1>       div    cl
32636 0000B1CD 80C430            <1>       add    ah, '0'
32637 0000B1D0 8827              <1>       mov    [edi], ah
32638 0000B1D2 20C0              <1>       and    al, al
32639 0000B1D4 740A              <1>       jz     short csftdf2_print_percent_1
32640 0000B1D6 4F                <1>       dec    edi
32641 0000B1D7 6698              <1>       cbw
32642 0000B1D9 F6F1              <1>       div    cl
32643 0000B1DB 80C430            <1>       add    ah, '0'
32644 0000B1DE 8827              <1>       mov    [edi], ah
32645                            <1>       ;and   al, al
32646                            <1>       ;jz    short csftdf2_print_percent_1
32647                            <1>       ;dec   edi
32648                            <1>       ;mov   [edi], '1' ; 100%
32649                            <1>
32650                            <1> csftdf2_print_percent_1:
32651 0000B1E0 BE[FF080100]      <1>       mov    esi, percentagestr
32652                            <1>       ;call  print_msg
32653                            <1>       ;retn
32654 0000B1E5 E973B1FFFF        <1>       jmp    print_msg
32655                            <1>
32656                            <1> csftdf2_read_file_sectors:
32657                            <1>       ; 19/03/2016
32658 0000B1EA 807E0300          <1>       cmp    byte [esi+LD_FATType], 0
32659 0000B1EE 0F8627070000      <1>         jna     csftdf2_read_fs_file_sectors
32660                            <1>
32661                            <1> csftdf2_read_fat_file_sectors:
32662                            <1>       ; 19/03/2016
32663                            <1>       ; 18/03/2016
32664                            <1>       ; return:
32665                            <1>       ;   CF = 0 & EDX > 0 -> END OF FILE
32666                            <1>       ;   CF = 0 & EDX = 0 -> not EOF
32667                            <1>       ;   CF = 1 -> read error (error code in AL)
32668                            <1>
32669                            <1> csftdf2_read_fat_file_secs_0:
32670 0000B1F4 8B15[A45A0100]    <1>       mov    edx, [csftdf_filesize]
32671 0000B1FA 2B15[C05A0100]    <1>       sub    edx, [csftdf_sf_rbytes]
32672 0000B200 3B15[B85A0100]    <1>       cmp    edx, [csftdf_r_size]
32673 0000B206 7306              <1>       jnb    short csftdf2_read_fat_file_secs_1
32674 0000B208 8915[B85A0100]    <1>       mov    [csftdf_r_size], edx
32675                            <1>
32676                            <1> csftdf2_read_fat_file_secs_1:
32677 0000B20E A1[B85A0100]      <1>       mov    eax, [csftdf_r_size]
32678 0000B213 29D2              <1>       sub    edx, edx
32679 0000B215 0FB74E11          <1>       movzx  ecx, word [esi+LD_BPB+BytesPerSec]
32680 0000B219 01C8              <1>       add    eax, ecx
32681 0000B21B 48                <1>       dec    eax
32682 0000B21C F7F1              <1>       div    ecx
32683 0000B21E 89C1              <1>       mov    ecx, eax ; sector count
32684 0000B220 A1[B05A0100]      <1>       mov    eax, [csftdf_sf_cluster]
32685                            <1>
32686                            <1>       ; EBX = memory block address (current)
32687                            <1>
32688 0000B225 E821090000        <1>       call   read_fat_file_sectors
32689 0000B22A 7235              <1>       jc     short csftdf2_read_fat_file_secs_3
32690                            <1>
32691                            <1>       ; EBX = next memory address
32692                            <1>
32693 0000B22C A1[C05A0100]      <1>       mov    eax, [csftdf_sf_rbytes]
32694 0000B231 0305[B85A0100]    <1>       add    eax, [csftdf_r_size]
32695 0000B237 8B15[A45A0100]    <1>       mov    edx, [csftdf_filesize]
32696 0000B23D 39D0              <1>       cmp    eax, edx
32697 0000B23F 7320              <1>       jnb    short csftdf2_read_fat_file_secs_3 ; edx > 0
32698 0000B241 A3[C05A0100]      <1>       mov    [csftdf_sf_rbytes], eax
32699                            <1>
32700 0000B246 53                <1>       push   ebx ; *
32701                            <1>       ; get next cluster (csftdf_r_size! bytes)
32702 0000B247 A1[B05A0100]      <1>       mov    eax, [csftdf_sf_cluster]
32703 0000B24C E8CC060000        <1>       call   get_next_cluster
32704 0000B251 5B                <1>       pop    ebx ; *
32705 0000B252 7306              <1>       jnc    short csftdf2_read_fat_file_secs_2
32706                            <1>
32707                            <1>       ; 15/10/2016
32708                            <1>       ;Disk read error instad of drv not ready err
32709 0000B254 B811000000        <1>       mov    eax, 17 ; Read error !
32710 0000B259 C3                <1>       retn
32711                            <1>
32712                            <1> csftdf2_read_fat_file_secs_2:
32713 0000B25A 29D2              <1>       sub    edx, edx ; 0
32714 0000B25C A3[B05A0100]      <1>       mov    [csftdf_sf_cluster], eax ; next cluster
32715                            <1>
32716                            <1> csftdf2_read_fat_file_secs_3:
32717 0000B261 C3                <1>       retn
32718                            <1>
32719                            <1> csftdf2_read_sf_cluster:
32720                            <1>       ; 19/03/2016
32721 0000B262 BB00000700        <1>       mov    ebx, Cluster_Buffer ; buffer address (64KB)
32722                            <1>
32723 0000B267 803D[C85A0100]00  <1>       cmp    byte [csftdf_percentage], 0
32724 0000B26E 760D              <1>       jna    short csftdf2_read_sf_clust_2
32725                            <1>
32726 0000B270 53                <1>       push   ebx ; *
32727                            <1>
32728                            <1> csftdf2_read_sf_clust_next:
32729 0000B271 E836FFFFFF        <1>       call   csftdf2_print_percentage
32730                            <1>
32731                            <1> csftdf2_read_sf_clust_0:
32732 0000B276 8B35[CC5A0100]    <1>       mov    esi, [csftdf_sf_drv_dt]
32733                            <1> csftdf2_read_sf_clust_1:
32734 0000B27C 5B                <1>       pop    ebx ; *
32735                            <1>
32736                            <1> csftdf2_read_sf_clust_2:
```

```
32737 0000B27D 89DA                <1>       mov    edx, ebx
32738 0000B27F 0315[B85A0100]      <1>       add    edx, [csftdf_r_size]
32739 0000B285 81FA00000800        <1>       cmp    edx, Cluster_Buffer + 65536
32740 0000B28B 772F                <1>       ja     short csftdf2_write_df_cluster
32741                              <1>
32742 0000B28D E858FFFFFF          <1>       call   csftdf2_read_file_sectors ; 19/03/2016
32743 0000B292 0F8280020000        <1>         jc     csftdf2_save_fat_file_err2 ; eocc! or disk error!
32744                              <1>
32745 0000B298 09D2                <1>       or     edx, edx ; edx > 0 -> EOF
32746 0000B29A 7520                <1>       jnz    short csftdf2_write_df_cluster
32747                              <1>
32748 0000B29C 803D[C85A0100]00    <1>       cmp    byte [csftdf_percentage], 0
32749 0000B2A3 76D8                <1>       jna    short csftdf2_read_sf_clust_2
32750                              <1>
32751 0000B2A5 53                  <1>       push   ebx ; *
32752                              <1>
32753                              <1>       ; Set cursor position
32754                              <1>       ; AH= 02h, BH= Page Number, DH= Row, DL= Column
32755 0000B2A6 8A3D[C95A0100]      <1>       mov    bh, [csftdf_videopage]
32756 0000B2AC 668B15[CA5A0100]    <1>       mov    dx, [csftdf_cursorpos]
32757 0000B2B3 B402                <1>       mov    ah, 2
32758 0000B2B5 E8E061FFFF          <1>       call   _int10h
32759 0000B2BA EBB5                <1>       jmp    short csftdf2_read_sf_clust_next
32760                              <1>
32761                              <1> csftdf2_write_df_cluster:
32762                              <1>       ; 19/03/2016
32763 0000B2BC 8B35[D05A0100]      <1>       mov    esi, [csftdf_df_drv_dt]
32764 0000B2C2 BB00000700          <1>       mov    ebx, Cluster_Buffer ; buffer address (64KB)
32765                              <1>
32766                              <1> csftdf2_write_df_clust_next:
32767 0000B2C7 E855000000          <1>       call   csftdf2_write_file_sectors ; 19/03/2016
32768 0000B2CC 0F8246020000        <1>         jc     csftdf2_save_fat_file_err2 ; eocc! or disk error!
32769                              <1>
32770 0000B2D2 09D2                <1>       or     edx, edx ; edx > 0 -> EOF
32771 0000B2D4 750A                <1>       jnz    short csftdf2_rw_f_clust_ok
32772                              <1>
32773 0000B2D6 81FB00000800        <1>       cmp    ebx, Cluster_Buffer + 65536
32774 0000B2DC 72E9                <1>       jb     short csftdf2_write_df_clust_next
32775                              <1>
32776 0000B2DE EB82                <1>       jmp    short csftdf2_read_sf_cluster
32777                              <1>
32778                              <1> csftdf2_rw_f_clust_ok:
32779 0000B2E0 803D[C85A0100]00    <1>       cmp    byte [csftdf_percentage], 0
32780 0000B2E7 0F86B2010000        <1>         jna    csftdf2_save_fat_file_4 ; 25/03/2016
32781                              <1>
32782                              <1>       ; "100%"
32783 0000B2ED BF[FF080100]        <1>       mov    edi, percentagestr
32784 0000B2F2 B031                <1>       mov    al, '1'
32785 0000B2F4 AA                  <1>       stosb
32786 0000B2F5 B030                <1>       mov    al, '0'
32787 0000B2F7 AA                  <1>       stosb
32788 0000B2F8 AA                  <1>       stosb
32789                              <1>
32790 0000B2F9 8A3D[C95A0100]      <1>       mov    bh, [csftdf_videopage]
32791 0000B2FF 668B15[CA5A0100]    <1>       mov    dx, [csftdf_cursorpos]
32792 0000B306 B402                <1>       mov    ah, 2
32793 0000B308 E88D61FFFF          <1>       call   _int10h
32794                              <1>
32795 0000B30D BE[FF080100]        <1>       mov    esi, percentagestr
32796 0000B312 E846B0FFFF          <1>       call   print_msg
32797                              <1>
32798 0000B317 E983010000          <1>         jmp    csftdf2_save_fat_file_4
32799                              <1>
32800                              <1> csftdf2_load_fs_file:
32801                              <1>       ; temporary - 18/03/2016
32802 0000B31C E96F020000          <1>         jmp    csftdf2_read_error
32803                              <1>
32804                              <1> csftdf2_write_file_sectors:
32805                              <1>       ; 19/03/2016
32806 0000B321 807E0300            <1>       cmp    byte [esi+LD_FATType], 0
32807 0000B325 0F86F1050000        <1>         jna    csftdf2_write_fs_file_sectors
32808                              <1>
32809                              <1> csftdf2_write_fat_file_sectors:
32810                              <1>       ; 19/03/2016
32811                              <1>       ; 18/03/2016
32812                              <1>       ; return:
32813                              <1>       ;   CF = 0 & EDX > 0 -> END OF FILE
32814                              <1>       ;   CF = 0 & EDX = 0 -> not EOF
32815                              <1>       ;   CF = 1 -> write error (error code in AL)
32816                              <1>
32817                              <1> csftdf2_write_fat_file_secs_0:
32818 0000B32B 8B15[A45A0100]      <1>       mov    edx, [csftdf_filesize]
32819 0000B331 2B15[C45A0100]      <1>       sub    edx, [csftdf_df_wbytes]
32820 0000B337 3B15[BC5A0100]      <1>       cmp    edx, [csftdf_w_size]
32821 0000B33D 7306                <1>       jnb    short csftdf2_write_fat_file_secs_1
32822 0000B33F 8915[BC5A0100]      <1>       mov    [csftdf_w_size], edx
32823                              <1>
32824                              <1> csftdf2_write_fat_file_secs_1:
32825 0000B345 A1[BC5A0100]        <1>       mov    eax, [csftdf_w_size]
32826 0000B34A 29D2                <1>       sub    edx, edx
32827 0000B34C 0FB74E11            <1>       movzx  ecx, word [esi+LD_BPB+BytesPerSec]
32828 0000B350 01C8                <1>       add    eax, ecx
32829 0000B352 48                  <1>       dec    eax
32830 0000B353 F7F1                <1>       div    ecx
32831 0000B355 89C1                <1>       mov    ecx, eax ; sector count
32832 0000B357 A1[B45A0100]        <1>       mov    eax, [csftdf_df_cluster]
32833                              <1>
32834                              <1>       ; EBX = memory block address (current)
32835                              <1>
32836 0000B35C E8A20F0000          <1>       call   write_fat_file_sectors
32837 0000B361 7259                <1>       jc     short csftdf2_write_fat_file_secs_4
32838                              <1>
```

```
                                <1>       ; EBX = next memory address
                                <1>
0000B363 A1[C45A0100]           <1>       mov    eax, [csftdf_df_wbytes]
0000B368 0305[BC5A0100]         <1>       add    eax, [csftdf_w_size]
0000B36E 8B15[A45A0100]         <1>       mov    edx, [csftdf_filesize]
0000B374 39D0                   <1>       cmp    eax, edx
0000B376 7344                   <1>       jnb    short csftdf2_write_fat_file_secs_4
0000B378 A3[C45A0100]           <1>       mov    [csftdf_df_wbytes], eax
                                <1>       ;
0000B37D A3[665A0100]           <1>       mov    [DestinationFile_DirEntry+DirEntry_FileSize], eax
                                <1>
0000B382 53                     <1>       push   ebx ; *
                                <1>
0000B383 803D[A25A0100]01       <1>       cmp    byte [DestinationFileFound], 1
0000B38A 7210                   <1>       jb     short csftdf2_write_fat_file_secs_2
                                <1>
                                <1>       ; get next cluster (csftdf_w_size! bytes)
0000B38C A1[B45A0100]           <1>       mov    eax, [csftdf_df_cluster]
0000B391 E887050000             <1>       call   get_next_cluster
0000B396 731C                   <1>       jnc    short csftdf2_write_fat_file_secs_3
                                <1>
0000B398 21C0                   <1>       and    eax, eax ; end of cluster chain!?
0000B39A 7521                   <1>       jnz    short csftdf2_write_fat_file_secs_5 ; disk error !
                                <1>
                                <1> csftdf2_write_fat_file_secs_2:
0000B39C A1[B45A0100]           <1>       mov    eax, [csftdf_df_cluster] ; last cluster
0000B3A1 E8800E0000             <1>       call   add_new_cluster
0000B3A6 7215                   <1>       jc     short csftdf2_write_fat_file_secs_5
                                <1>
                                <1>       ; NOTE: Destination file size may be bigger than
                                <1>       ; source file size when the last reading fails after here.
                                <1>       ; (The last -empty- cluster of destination file must be
                                <1>       ; truncated and LMDT must be current date&time for partial
                                <1>       ; copy result!)
0000B3A8 8B15[BC5A0100]         <1>       mov    edx, [csftdf_w_size] ; bytes per cluster
0000B3AE 0115[665A0100]         <1>       add    [DestinationFile_DirEntry+DirEntry_FileSize], edx
                                <1>
                                <1> csftdf2_write_fat_file_secs_3:
0000B3B4 5B                     <1>       pop    ebx ; *
0000B3B5 29D2                   <1>       sub    edx, edx ; 0
0000B3B7 A3[B45A0100]           <1>       mov    [csftdf_df_cluster], eax ; next cluster
                                <1>
                                <1> csftdf2_write_fat_file_secs_4:
0000B3BC C3                     <1>       retn
                                <1>
                                <1> csftdf2_write_fat_file_secs_5:
0000B3BD 5B                     <1>       pop    ebx ; *
                                <1>       ; 16/10/2016 (1Dh -> 18)
0000B3BE B812000000             <1>       mov    eax, 18 ; Write error !
0000B3C3 C3                     <1>       retn
                                <1>
                                <1> csftdf2_save_file:
                                <1>       ; 25/03/2016
                                <1>       ; 19/03/2016
                                <1>       ; 18/03/2016
0000B3C4 8B35[D05A0100]         <1>       mov    esi, [csftdf_df_drv_dt] ; logical dos drv desc. tbl.
                                <1>
0000B3CA 8B1D[A85A0100]         <1>       mov    ebx, [csftdf_sf_mem_addr] ; memory block address
                                <1>
0000B3D0 807E0300               <1>       cmp    byte [esi+LD_FATType], 0
0000B3D4 0F86F4010000           <1>         jna     csftdf2_save_fs_file
                                <1>
                                <1> csftdf2_save_fat_file:
0000B3DA 53                     <1>       push   ebx; *
                                <1>
0000B3DB 803D[C85A0100]00       <1>       cmp    byte [csftdf_percentage], 0
0000B3E2 7724                   <1>       ja     short csftdf2_save_fat_file_0
                                <1>
                                <1>       ; Set cursor position
                                <1>       ; AH= 02h, BH= Page Number, DH= Row, DL= Column
0000B3E4 8A3D[C95A0100]         <1>       mov    bh, [csftdf_videopage]
0000B3EA 668B15[CA5A0100]       <1>       mov    dx, [csftdf_cursorpos]
0000B3F1 B402                   <1>       mov    ah, 2
0000B3F3 E8A260FFFF             <1>       call   _int10h
                                <1>
0000B3F8 BE[F3080100]           <1>       mov    esi, msg_writing
0000B3FD E85BAFFFFF             <1>       call   print_msg
                                <1>
                                <1> csftdf2_save_fat_file_next:
0000B402 8B35[D05A0100]         <1>       mov    esi, [csftdf_df_drv_dt] ; 25/03/2016
                                <1>
                                <1> csftdf2_save_fat_file_0:
0000B408 5B                     <1>       pop    ebx ; *
                                <1>
                                <1> csftdf2_save_fat_file_1:
0000B409 E813FFFFFF             <1>       call   csftdf2_write_file_sectors ; 19/03/2016
0000B40E 0F827E010000           <1>         jc      csftdf2_rw_error ; eocc! or disk error!
                                <1>
0000B414 09D2                   <1>       or     edx, edx ; edx > 0 -> EOF
0000B416 756D                   <1>         jnz     short csftdf2_save_fat_file_3 ; 25/03/2016
                                <1>
0000B418 803D[C85A0100]00       <1>       cmp    byte [csftdf_percentage], 0
0000B41F 76E8                   <1>       jna    short csftdf2_save_fat_file_1
                                <1>
0000B421 B020                   <1>       mov    al, 20h
0000B423 BF[FF080100]           <1>       mov    edi, percentagestr
0000B428 AA                     <1>       stosb
0000B429 AA                     <1>       stosb
0000B42A A1[C45A0100]           <1>       mov    eax, [csftdf_df_wbytes]
0000B42F BA64000000             <1>       mov    edx, 100
0000B434 F7E2                   <1>       mul    edx
0000B436 8B0D[A45A0100]         <1>       mov    ecx, [csftdf_filesize]
```

```
32941 0000B43C F7F1              <1>       div    ecx
32942 0000B43E B10A              <1>       mov    cl, 10
32943 0000B440 F6F1              <1>       div    cl
32944 0000B442 80C430            <1>       add    ah, '0'
32945 0000B445 8827              <1>       mov    [edi], ah
32946 0000B447 20C0              <1>       and    al, al
32947 0000B449 740A              <1>       jz     short csftdf2_save_fat_file_2
32948 0000B44B 4F                <1>       dec    edi
32949 0000B44C 6698              <1>       cbw
32950 0000B44E F6F1              <1>       div    cl
32951 0000B450 80C430            <1>       add    ah, '0'
32952 0000B453 8827              <1>       mov    [edi], ah
32953                            <1>       ;and   al, al
32954                            <1>       ;jz    short csftdf2_save_fat_file_2
32955                            <1>       ;dec   edi
32956                            <1>       ;mov   [edi], '1' ; 100%
32957                            <1>
32958                            <1> csftdf2_save_fat_file_2:
32959 0000B455 53                <1>       push   ebx ; *
32960                            <1>
32961 0000B456 E802000000        <1>       call   csftdf2_print_wr_percentage ; 25/03/2016
32962                            <1>
32963 0000B45B EBA5              <1>         jmp     csftdf2_save_fat_file_next
32964                            <1>
32965                            <1> csftdf2_print_wr_percentage:
32966                            <1>       ; Set cursor position
32967                            <1>       ; AH= 02h, BH= Page Number, DH= Row, DL= Column
32968 0000B45D 8A3D[C95A0100]    <1>       mov    bh, [csftdf_videopage]
32969 0000B463 668B15[CA5A0100]  <1>       mov    dx, [csftdf_cursorpos]
32970 0000B46A B402              <1>       mov    ah, 2
32971 0000B46C E82960FFFF        <1>       call   _int10h
32972                            <1>
32973 0000B471 BE[F3080100]      <1>       mov    esi, msg_writing
32974 0000B476 E8E2AEFFFF        <1>       call   print_msg
32975                            <1>
32976 0000B47B BE[FF080100]      <1>       mov    esi, percentagestr
32977                            <1>       ;call  print_msg
32978                            <1>       ;retn
32979 0000B480 E9D8AEFFFF        <1>       jmp    print_msg
32980                            <1>
32981                            <1> csftdf2_save_fat_file_3:
32982 0000B485 803D[C85A0100]00  <1>       cmp    byte [csftdf_percentage], 0
32983 0000B48C 7611              <1>         jna     csftdf2_save_fat_file_4 ; 25/03/2016
32984                            <1>
32985                            <1>       ; "100%"
32986 0000B48E BF[FF080100]      <1>       mov    edi, percentagestr
32987 0000B493 B031              <1>       mov    al, '1'
32988 0000B495 AA                <1>       stosb
32989 0000B496 B030              <1>       mov    al, '0'
32990 0000B498 AA                <1>       stosb
32991 0000B499 AA                <1>       stosb
32992                            <1>
32993 0000B49A E8BEFFFFFF        <1>       call   csftdf2_print_wr_percentage
32994                            <1>
32995                            <1> csftdf2_save_fat_file_4:
32996 0000B49F 803D[A25A0100]00  <1>       cmp    byte [DestinationFileFound], 0
32997 0000B4A6 7647              <1>       jna    short csftdf2_save_fat_file_6
32998                            <1>
32999 0000B4A8 8B35[D05A0100]    <1>       mov    esi, [csftdf_df_drv_dt] ; 31/03/2016
33000                            <1>
33001 0000B4AE A1[B45A0100]      <1>       mov    eax, [csftdf_df_cluster] ; last cluster
33002 0000B4B3 E865040000        <1>       call   get_next_cluster
33003 0000B4B8 7235              <1>       jc     short csftdf2_save_fat_file_6 ; eocc! or disk error!
33004                            <1>
33005 0000B4BA A1[B45A0100]      <1>       mov    eax, [csftdf_df_cluster] ; last cluster
33006                            <1>       ;xor   ecx, ecx
33007                            <1>       ;mov   [FAT_ClusterCounter], ecx ; 0 ; reset
33008                            <1>       ;dec   ecx ; 0FFFFFFFFh
33009                            <1>       ;shr   ecx, 4 ; 28 bit ; 0FFFFFFFh
33010 0000B4BF B9FFFFFF0F        <1>       mov    ecx, 0FFFFFFFh
33011 0000B4C4 E87E070000        <1>       call   update_cluster
33012 0000B4C9 7224              <1>       jc     short csftdf2_save_fat_file_6 ; really last cluster!?
33013                            <1>
33014 0000B4CB A3[B45A0100]      <1>       mov    [csftdf_df_cluster], eax ; next cluster
33015                            <1>
33016                            <1>       ; byte [FAT_BuffValidData] = 2
33017 0000B4D0 E82F0A0000        <1>       call   save_fat_buffer
33018 0000B4D5 730E              <1>       jnc    short csftdf2_save_fat_file_5
33019                            <1>
33020 0000B4D7 8B15[A45A0100]    <1>       mov    edx, [csftdf_filesize]
33021 0000B4DD 8915[665A0100]    <1>       mov    [DestinationFile_DirEntry+DirEntry_FileSize], edx
33022 0000B4E3 EB58              <1>       jmp    short csftdf2_save_fat_file_err3
33023                            <1>
33024                            <1> csftdf2_save_fat_file_5:
33025 0000B4E5 A1[B45A0100]      <1>       mov    eax, [csftdf_df_cluster]
33026                            <1>
33027                            <1>       ; EAX = First cluster to be truncated/unlinked
33028                            <1>       ; ESI = Logical dos drive description table address
33029 0000B4EA E8580C0000        <1>       call   truncate_cluster_chain
33030                            <1>
33031                            <1> csftdf2_save_fat_file_6:
33032                            <1>       ; 28/03/2016
33033 0000B4EF BE[D5590100]      <1>       mov    esi, SourceFile_DirEntry+DirEntry_Attr ; +11 to + 18
33034 0000B4F4 BF[555A0100]      <1>       mov    edi, DestinationFile_DirEntry+DirEntry_Attr ; +11 to + 18
33035 0000B4F9 A4                <1>       movsb ; +11
33036 0000B4FA A5                <1>       movsd ; +12 .. +15
33037 0000B4FB 66A5              <1>       movsw ; +16 .. +17
33038                            <1>             ; + 18
33039 0000B4FD 83C604            <1>       add    esi, 4
33040 0000B500 83C704            <1>       add    edi, 4
33041 0000B503 A5                <1>       movsd  ; DirEntry_WrtTime ; +22 .. +25
33042                            <1>
```

```
33043 0000B504 8B15[A45A0100]      <1>       mov    edx, [csftdf_filesize]
33044 0000B50A 8915[665A0100]      <1>       mov    [DestinationFile_DirEntry+DirEntry_FileSize], edx
33045                              <1>
33046 0000B510 E8B7F0FFFF          <1>       call   convert_current_date_time
33047                              <1>       ; DX = Date in dos dir entry format
33048                              <1>       ; AX = Time in dos dir entry format
33049 0000B515 EB4D                <1>       jmp    short csftdf2_save_fat_file_7
33050                              <1>
33051                              <1> csftdf2_save_fat_file_err1:
33052 0000B517 5B                  <1>       pop    ebx ; *
33053                              <1> csftdf2_save_fat_file_err2:
33054 0000B518 A1[C45A0100]        <1>       mov    eax, [csftdf_df_wbytes]
33055 0000B51D 8B15[665A0100]      <1>       mov    edx, [DestinationFile_DirEntry+DirEntry_FileSize]
33056 0000B523 39C2                <1>       cmp    edx, eax
33057 0000B525 7616                <1>       jna    short csftdf2_save_fat_file_err3
33058 0000B527 A1[B45A0100]        <1>       mov    eax, [csftdf_df_cluster] ; last (empty) cluster
33059                              <1>       ; ESI = Logical dos drive description table address
33060 0000B52C E8160C0000          <1>       call   truncate_cluster_chain
33061 0000B531 720A                <1>       jc     short csftdf2_save_fat_file_err3
33062 0000B533 A1[C45A0100]        <1>       mov    eax, [csftdf_df_wbytes]
33063 0000B538 A3[665A0100]        <1>       mov    [DestinationFile_DirEntry+DirEntry_FileSize], eax
33064                              <1> csftdf2_save_fat_file_err3:
33065 0000B53D E88AF0FFFF          <1>       call   convert_current_date_time
33066                              <1>       ; DX = Date in dos dir entry format
33067                              <1>       ; AX = Time in dos dir entry format
33068 0000B542 C605[575A0100]00    <1>       mov    byte [DestinationFile_DirEntry+DirEntry_CrtTimeTenth], 0
33069 0000B549 66A3[585A0100]      <1>       mov    [DestinationFile_DirEntry+DirEntry_CrtTime], ax
33070 0000B54F 668915[5A5A0100]    <1>       mov    [DestinationFile_DirEntry+DirEntry_CrtDate], dx
33071 0000B556 66A3[605A0100]      <1>       mov    [DestinationFile_DirEntry+DirEntry_WrtTime], ax
33072 0000B55C 668915[625A0100]    <1>       mov    [DestinationFile_DirEntry+DirEntry_WrtDate], dx
33073 0000B563 F9                  <1>       stc
33074                              <1> csftdf2_save_fat_file_7:
33075 0000B564 9C                  <1>       pushf
33076 0000B565 668915[5C5A0100]    <1>       mov    [DestinationFile_DirEntry+DirEntry_LastAccDate], dx
33077 0000B56C BE[4A5A0100]        <1>       mov    esi, DestinationFile_DirEntry
33078 0000B571 BF00000800          <1>       mov    edi, Directory_Buffer
33079 0000B576 0FB70D[725A0100]    <1>       movzx  ecx, word [DestinationFile_DirEntryNumber] ; (<2048)
33080 0000B57D 66C1E105            <1>       shl    cx, 5 ; 32 * directory entry number
33081 0000B581 01CF                <1>       add    edi, ecx
33082                              <1>       ;mov   ecx, 8
33083 0000B583 66B90800            <1>       mov    cx, 8
33084 0000B587 F3A5                <1>       rep    movsd
33085 0000B589 9D                  <1>       popf
33086 0000B58A 730B                <1>       jnc    short csftdf2_write_file_OK
33087                              <1>
33088                              <1> csftdf2_write_error:
33089                              <1>       ; 18/03/2016
33090 0000B58C B01D                <1>       mov    al, 1Dh ; write error
33091 0000B58E EB02                <1>       jmp    short csftdf2_rw_error
33092                              <1>
33093                              <1>       ; 16/03/2016
33094                              <1> csftdf2_read_error:
33095 0000B590 B011                <1>       mov    al, 17 ; ; Drive not ready or read error!
33096                              <1> csftdf2_rw_error:
33097 0000B592 A2[A15A0100]        <1>       mov    [csftdf_rw_err], al
33098                              <1>
33099                              <1> csftdf2_write_file_OK:
33100                              <1>       ; 18/03/2016
33101 0000B597 C605[B8560100]02    <1>       mov    byte [DirBuff_ValidData], 2
33102 0000B59E E8C7F0FFFF          <1>       call   save_directory_buffer
33103                              <1>
33104                              <1>       ; Update last modification date&time of destination
33105                              <1>       ; file's (parent) directory
33106 0000B5A3 E85DF1FFFF          <1>       call   update_parent_dir_lmdt
33107                              <1>       ;
33108 0000B5A8 A1[A85A0100]        <1>       mov    eax, [csftdf_sf_mem_addr] ; start address
33109                              <1>
33110 0000B5AD 21C0                <1>       and    eax, eax
33111 0000B5AF 750E                <1>       jnz    short csftdf2_dealloc_mblock
33112                              <1>
33113 0000B5B1 88C5                <1>       mov    ch, al ; 0 (Cluster r/w, not full loading)
33114                              <1> csftdf2_dealloc_retn:
33115 0000B5B3 8A0D[A15A0100]      <1>       mov    cl, [csftdf_rw_err]
33116 0000B5B9 A1[B45A0100]        <1>       mov    eax, [csftdf_df_cluster]
33117 0000B5BE C3                  <1>       retn
33118                              <1>
33119                              <1> csftdf2_dealloc_mblock:
33120 0000B5BF 8B0D[AC5A0100]      <1>       mov    ecx, [csftdf_sf_mem_bsize] ; block size
33121 0000B5C5 E866A0FFFF          <1>       call   deallocate_memory_block
33122 0000B5CA B5FF                <1>         mov     ch, 0FFh ; (File was full loaded at memory)
33123 0000B5CC EBE5                <1>       jmp    short csftdf2_dealloc_retn
33124                              <1>
33125                              <1> csftdf2_save_fs_file:
33126                              <1>       ; 16/10/2016 (1Dh -> 18)
33127                              <1>       ; temporary - (21/03/2016)
33128 0000B5CE B812000000          <1>       mov    eax, 18 ; write error
33129 0000B5D3 F9                  <1>       stc
33130 0000B5D4 C3                  <1>       retn
33131                              <1>
33132                              <1> create_file:
33133                              <1>       ; 16/10/2016
33134                              <1>       ; 24/03/2016, 31/03/2016
33135                              <1>       ; 20/03/2016, 21/03/2016, 23/03/2016
33136                              <1>       ; 19/03/2016 (TRDOS 396 = TRDOS v2.0)
33137                              <1>       ; 03/09/2011 (FILE.ASM, 'proc_create_file')
33138                              <1>       ; 09/08/2010
33139                              <1>       ;
33140                              <1>       ; INPUT ->
33141                              <1>       ;     EAX = File Size
33142                              <1>       ;     ESI = ASCIIZ File Name
33143                              <1>       ;     CL = File Attributes
33144                              <1>       ;     EBX = FFFFFFFFh -> create empty file
```

```
33145                                  <1>       ;                   (only for FAT fs)
33146                                  <1>       ; OUTPUT ->
33147                                  <1>       ;     CF = 0 ->
33148                                  <1>       ;      EAX = New file's first cluster
33149                                  <1>       ;      ESI = Logical Dos Drv Descr. Table Addr.
33150                                  <1>       ;      EBX = offset CreateFile_Size
33151                                  <1>       ;      ECX = Sectors per cluster (<256)
33152                                  <1>       ;      EDX = Directory entry index/number (<65536)
33153                                  <1>       ;     CF = 1 -> error code in AL
33154                                  <1>
33155                                  <1> ;     test   cl, 18h (directory or volume name)
33156                                  <1> ;     jnz    short loc_createfile_access_denied
33157 0000B5D5 80E107                  <1>       and    cl, 07h ; S, H, R
33158 0000B5D8 880D[F05A0100]          <1>         mov     [createfile_attrib], cl
33159                                  <1>
33160 0000B5DE 89D9                    <1>       mov    ecx, ebx
33161 0000B5E0 89F3                    <1>       mov    ebx, esi ; ASCIIZ File Name address
33162 0000B5E2 29D2                    <1>       sub    edx, edx
33163 0000B5E4 8A35[8E4E0100]          <1>         mov     dh, [Current_Drv]
33164 0000B5EA BE00010900              <1>         mov     esi, Logical_DOSDisks
33165 0000B5EF 01D6                    <1>       add    esi, edx
33166                                  <1>
33167 0000B5F1 8815[FB5A0100]          <1>       mov    [createfile_UpdatePDir], dl ; 0 ; 31/03/2016
33168                                  <1>
33169                                  <1>       ; LD_DiskType = 0 for write protection (read only)
33170 0000B5F7 807E0101                <1>       cmp    byte [esi+LD_DiskType], 1 ; 0 = Invalid
33171 0000B5FB 730A                    <1>       jnb    short loc_createfile_check_file_sytem
33172                                  <1>       ; 16/10/2016 (TRDOS Error code: 30, disk write protected)
33173 0000B5FD B81E000000              <1>       mov    eax, 30 ; 13h, MSDOS err : Disk write-protected
33174 0000B602 66BA0000                <1>       mov    dx, 0
33175                                  <1>       ; err retn: EDX = 0, EBX = File name offset
33176                                  <1>       ; ESI -> Dos drive description table address
33177 0000B606 C3                      <1>       retn
33178                                  <1>
33179                                  <1> ;loc_createfile_access_denied:
33180                                  <1> ;     mov    eax, 05h ; access denied (invalid attributes input)
33181                                  <1> ;     stc
33182                                  <1> ;     retn
33183                                  <1>
33184                                  <1> loc_createfile_check_file_sytem:
33185 0000B607 807E0301                <1>       cmp    byte [esi+LD_FATType], 1
33186 0000B60B 730A                    <1>       jnb    short loc_createfile_chk_empty_FAT_file_sign1
33187                                  <1>
33188 0000B60D A3[DC5A0100]            <1>       mov    [createfile_size], eax
33189                                  <1>       ; ESI = Logical Dos Drive Description Table address
33190                                  <1>       ; EBX = ASCIIZ File Name address
33191 0000B612 E9FE020000              <1>       jmp    create_fs_file
33192                                  <1>
33193                                  <1> loc_createfile_chk_empty_FAT_file_sign1:
33194                                  <1>       ; ECX = FFFFFFFFh -> create empty file if drive has FAT fs
33195 0000B617 41                      <1>       inc    ecx
33196 0000B618 7506                    <1>       jnz    short loc_createfile_chk_empty_FAT_file_sign2
33197 0000B61A 890D[DC5A0100]          <1>       mov    [createfile_size], ecx ; 0 ; empty file
33198                                  <1>
33199                                  <1> loc_createfile_chk_empty_FAT_file_sign2:
33200                                  <1>       ; 23/03/2016
33201 0000B620 668B4E11                <1>       mov    cx, [esi+LD_BPB+BytesPerSec]
33202 0000B624 66890D[F85A0100]        <1>       mov    [createfile_BytesPerSec], cx
33203                                  <1>
33204                                  <1>       ; EBX = ASCIIZ File Name address
33205 0000B62B 0FB65613                <1>       movzx  edx, byte [esi+LD_BPB+SecPerClust]
33206 0000B62F 8815[F15A0100]          <1>       mov    [createfile_SecPerClust], dl
33207 0000B635 8B4E74                  <1>       mov    ecx, [esi+LD_FreeSectors]
33208 0000B638 39D1                    <1>       cmp    ecx, edx ; byte [createfile_SecPerClust]
33209 0000B63A 7306                    <1>       jnb    short loc_create_fat_file
33210                                  <1>
33211                                  <1> loc_createfile_insufficient_disk_space:
33212 0000B63C B827000000              <1>       mov    eax, 27h
33213                                  <1> loc_createfile_gffc_retn:
33214 0000B641 C3                      <1>       retn
33215                                  <1>
33216                                  <1> loc_create_fat_file:
33217 0000B642 891D[D45A0100]          <1>       mov    [createfile_Name_Offset], ebx
33218 0000B648 890D[D85A0100]          <1>       mov    [createfile_FreeSectors], ecx
33219                                  <1>
33220                                  <1> loc_createfile_gffc_1:
33221 0000B64E E821050000              <1>       call   get_first_free_cluster
33222 0000B653 72EC                    <1>       jc     short loc_createfile_gffc_retn
33223                                  <1>
33224 0000B655 A3[E05A0100]            <1>       mov    [createfile_FFCluster], eax
33225                                  <1>
33226                                  <1> loc_createfile_locate_ffe_on_directory:
33227                                  <1>       ; Current directory fcluster <> Directory buffer cluster
33228                                  <1>       ; Current directory will be reloaded by
33229                                  <1>       ; 'locate_current_dir_file' procedure
33230                                  <1>       ;
33231                                  <1>       ; ESI = Logical Dos Drv Desc. Table Adress
33232 0000B65A 56                      <1>       push   esi ; *
33233 0000B65B 31C0                    <1>       xor    eax, eax
33234                                  <1>
33235 0000B65D A3[AE560100]            <1>       mov    dword [FAT_ClusterCounter], eax ; 0
33236                                  <1>       ; 21/03/2016
33237 0000B662 A2[FA5A0100]            <1>       mov    byte [createfile_wfc], al ; 0
33238                                  <1>
33239 0000B667 89C1                    <1>       mov    ecx, eax
33240 0000B669 6649                    <1>       dec    cx ; FFFFh
33241                                  <1>       ; CX = FFFFh -> find first deleted or free entry
33242                                  <1>       ; ESI would be ASCIIZ filename address if the call
33243                                  <1>       ; would not be for first free or deleted dir entry
33244 0000B66B E8D6E7FFFF              <1>       call   locate_current_dir_file
33245 0000B670 0F83EE000000            <1>       jnc    loc_createfile_set_ff_dir_entry
33246 0000B676 5E                      <1>       pop    esi ; *
```

```
33247                                  <1>       ; ESI = Logical DOS Drv. Description Table Address
33248 0000B677 83F802                  <1>       cmp    eax, 2
33249 0000B67A 7402                    <1>       je     short loc_createfile_add_new_cluster
33250                                  <1> loc_createfile_locate_file_stc_retn:
33251 0000B67C F9                      <1>       stc
33252 0000B67D C3                      <1>       retn
33253                                  <1>
33254                                  <1> loc_createfile_add_new_cluster:
33255 0000B67E 803D[8D4E0100]02        <1>       cmp    byte [Current_FATType], 2
33256                                  <1>       ;cmp   byte [esi+LD_FATType], 2
33257 0000B685 770C                    <1>       ja     short loc_createfile_add_new_cluster_check_fsc
33258 0000B687 803D[8C4E0100]01        <1>       cmp    byte [Current_Dir_Level], 1
33259                                  <1>       ;cmp   byte [esi+LD_CDirLevel], 1
33260 0000B68E 7303                    <1>       jnb    short loc_createfile_add_new_cluster_check_fsc
33261                                  <1>
33262                                  <1>       ;mov   eax, 12
33263 0000B690 B00C                    <1>       mov    al, 12 ; No more files
33264                                  <1>
33265                                  <1> loc_createfile_anc_retn:
33266 0000B692 C3                      <1>       retn
33267                                  <1>
33268                                  <1> loc_createfile_add_new_cluster_check_fsc:
33269 0000B693 8B0D[D85A0100]          <1>       mov    ecx, [createfile_FreeSectors]
33270 0000B699 0FB605[F15A0100]        <1>       movzx  eax, byte [createfile_SecPerClust]
33271 0000B6A0 66D1E0                  <1>       shl    ax, 1 ; AX = 2 * AX
33272 0000B6A3 39C1                    <1>       cmp    ecx, eax
33273 0000B6A5 7295                    <1>         jb   short loc_createfile_insufficient_disk_space
33274                                  <1>
33275                                  <1> loc_createfile_add_new_subdir_cluster:
33276 0000B6A7 8B15[BD560100]          <1>       mov    edx, [DirBuff_Cluster]
33277 0000B6AD 8915[E45A0100]          <1>       mov    [createfile_LastDirCluster], edx
33278                                  <1>
33279 0000B6B3 A1[E05A0100]            <1>       mov    eax, [createfile_FFCluster]
33280 0000B6B8 E846040000              <1>       call   load_FAT_sub_directory
33281 0000B6BD 72D3                    <1>       jc     short loc_createfile_anc_retn
33282                                  <1>
33283                                  <1> pass_createfile_add_new_subdir_cluster:
33284                                  <1>       ;movzx eax, word [esi+LD_BPB+BytesPerSec]
33285 0000B6BF 0FB705[F85A0100]        <1>       movzx  eax, word [createfile_BytesPerSec] ; 23/03/2016
33286 0000B6C6 F7E1                    <1>       mul    ecx ; ecx = directory buffer sector count
33287 0000B6C8 89C1                    <1>       mov    ecx, eax
33288 0000B6CA C1E902                  <1>       shr    ecx, 2 ; dword count
33289 0000B6CD 29C0                    <1>       sub    eax, eax ; 0
33290 0000B6CF F3AB                    <1>       rep    stosd
33291                                  <1>       ;
33292 0000B6D1 C605[B8560100]02        <1>       mov    byte [DirBuff_ValidData], 2
33293 0000B6D8 E88DEFFFFF              <1>       call   save_directory_buffer
33294 0000B6DD 72B3                    <1>       jc     short loc_createfile_anc_retn
33295                                  <1>
33296                                  <1> loc_createfile_save_added_subdir_cluster:
33297 0000B6DF A1[E45A0100]            <1>       mov    eax, [createfile_LastDirCluster]
33298 0000B6E4 8B0D[E05A0100]          <1>       mov    ecx, [createfile_FFCluster]
33299 0000B6EA E858050000              <1>       call   update_cluster
33300 0000B6EF 7304                    <1>       jnc    short loc_createfile_save_fat_buffer_0
33301 0000B6F1 09C0                    <1>       or     eax, eax ; EAX = 0 -> cluster value is 0 or eocc
33302 0000B6F3 751A                    <1>       jnz    short loc_createfile_save_fat_buffer_stc_retn
33303                                  <1>
33304                                  <1> loc_createfile_save_fat_buffer_0:
33305 0000B6F5 A1[E05A0100]            <1>       mov    eax, [createfile_FFCluster]
33306 0000B6FA A3[E45A0100]            <1>       mov    [createfile_LastDirCluster], eax
33307 0000B6FF B9FFFFFF0F              <1>       mov    ecx, 0FFFFFFFh ; 28 bit
33308 0000B704 E83E050000              <1>       call   update_cluster
33309 0000B709 7306                    <1>       jnc    short loc_createfile_save_fat_buffer_1
33310 0000B70B 09C0                    <1>       or     eax, eax ; Was it free cluster
33311 0000B70D 7402                    <1>       jz     short loc_createfile_save_fat_buffer_1
33312                                  <1>
33313                                  <1> loc_createfile_save_fat_buffer_stc_retn:
33314 0000B70F F9                      <1>       stc
33315                                  <1> loc_createfile_save_fat_buffer_retn:
33316                                  <1> loc_createfile_gffc_2_stc_retn:
33317 0000B710 C3                      <1>       retn
33318                                  <1>
33319                                  <1> loc_createfile_save_fat_buffer_1:
33320                                  <1>       ; byte [FAT_BuffValidData] = 2
33321 0000B711 E8EE070000              <1>       call   save_fat_buffer
33322 0000B716 72F8                    <1>       jc     short loc_createfile_save_fat_buffer_retn
33323                                  <1>
33324 0000B718 803D[AE560100]01        <1>       cmp    byte [FAT_ClusterCounter], 1
33325 0000B71F 7222                    <1>       jb     short loc_createfile_save_fat_buffer_2
33326                                  <1>
33327                                  <1>       ; ESI = Logical DOS Drive Description Table address
33328 0000B721 A1[AE560100]            <1>       mov    eax, [FAT_ClusterCounter]
33329                                  <1>
33330 0000B726 C605[AE560100]00        <1>       mov    byte [FAT_ClusterCounter], 0 ; 21/03/2016
33331                                  <1>
33332 0000B72D 66BB01FF                <1>       mov    bx, 0FF01h ; add free clusters
33333 0000B731 E863080000              <1>       call   calculate_fat_freespace
33334                                  <1>
33335                                  <1>       ;inc   eax ; 0FFFFFFFFh -> 0 ; recalculation is needed!
33336                                  <1>       ;jnz   short loc_createfile_save_fat_buffer_2
33337                                  <1>
33338                                  <1>       ; ecx > 0 -> Recalculation is needed
33339 0000B736 09C9                    <1>       or     ecx, ecx
33340 0000B738 7409                    <1>       jz     short loc_createfile_save_fat_buffer_2
33341                                  <1>
33342 0000B73A 66BB00FF                <1>       mov    bx, 0FF00h ; ; recalculate free space
33343 0000B73E E856080000              <1>       call   calculate_fat_freespace
33344                                  <1>
33345                                  <1> loc_createfile_save_fat_buffer_2:
33346                                  <1>       ;call  update_parent_dir_lmdt
33347                                  <1>
33348                                  <1> loc_createfile_gffc_2:
```

```
33349 0000B743 E82C040000          <1>       call   get_first_free_cluster
33350 0000B748 72C6                <1>       jc     short loc_createfile_gffc_2_stc_retn
33351                              <1>
33352 0000B74A A3[E05A0100]        <1>       mov    [createfile_FFCluster], eax
33353                              <1>
33354 0000B74F A1[E45A0100]        <1>       mov    eax, [createfile_LastDirCluster]
33355                              <1>
33356 0000B754 E8AA030000          <1>       call   load_FAT_sub_directory
33357 0000B759 72B5                <1>       jc     short loc_createfile_gffc_2_stc_retn
33358                              <1>
33359 0000B75B BF00000800          <1>       mov    edi, Directory_Buffer
33360                              <1>
33361 0000B760 6629DB              <1>       sub    bx, bx ; directory entry index/number = 0
33362                              <1>
33363 0000B763 56                  <1>       push   esi ; * ; 23/03/2016
33364                              <1>
33365                              <1> loc_createfile_set_ff_dir_entry:
33366 0000B764 66891D[F25A0100]    <1>       mov    [createfile_DirIndex], bx
33367                              <1>
33368                              <1>         ; EDI = Directory entry address
33369 0000B76B 8B35[D45A0100]      <1>       mov    esi, [createfile_Name_Offset]
33370 0000B771 A1[E05A0100]        <1>       mov    eax, [createfile_FFCluster]
33371 0000B776 A3[E85A0100]        <1>       mov    [createfile_Cluster], eax ; 24/03/2016
33372 0000B77B B5FF                <1>       mov    ch, 0FFh
33373 0000B77D 8A0D[F05A0100]      <1>         mov  cl, [createfile_attrib] ; file attributes
33374                              <1>       ; CH > 0 -> File size is in [EBX]
33375 0000B783 BB[DC5A0100]        <1>       mov    ebx, createfile_size
33376                              <1>
33377 0000B788 E800EEFFFF          <1>       call   make_directory_entry
33378                              <1>
33379 0000B78D 5E                  <1>       pop    esi ; * ; ESI = Logical Dos Drv Desc. Table address
33380                              <1>
33381 0000B78E C605[B8560100]02    <1>       mov    byte [DirBuff_ValidData], 2
33382 0000B795 E8D0EEFFFF          <1>       call   save_directory_buffer
33383 0000B79A 7221                <1>       jc     short loc_createfile_set_ff_dir_entry_retn
33384                              <1>
33385 0000B79C C605[FB5A0100]01    <1>       mov    byte [createfile_UpdatePDir], 1 ; 31/03/2016
33386                              <1>
33387                              <1> loc_createfile_get_set_write_file_cluster:
33388 0000B7A3 A1[DC5A0100]        <1>       mov    eax, [createfile_size]
33389 0000B7A8 09C0                <1>       or     eax, eax
33390 0000B7AA 7570                <1>       jnz    short loc_createfile_get_set_wfc_cont
33391 0000B7AC 40                  <1>       inc    eax
33392                              <1>       ; 23/03/2016
33393 0000B7AD 0FB61D[F15A0100]    <1>       movzx  ebx, byte [createfile_SecPerClust]
33394                              <1>       ;movzx ecx, word [esi+LD_BPB+BytesPerSec] ; 512
33395 0000B7B4 0FB70D[F85A0100]    <1>         movzx   ecx, word [createfile_BytesPerSec] ; 512
33396 0000B7BB EB7C                <1>       jmp    loc_createfile_set_cluster_count
33397                              <1>
33398                              <1> loc_createfile_set_ff_dir_entry_retn:
33399 0000B7BD C3                  <1>       retn
33400                              <1>
33401                              <1> loc_createfile_write_fcluster_to_disk:
33402 0000B7BE 034668              <1>       add    eax, [esi+LD_DATABegin] ; convert to physical address
33403 0000B7C1 BB00000700          <1>       mov    ebx, Cluster_Buffer
33404                              <1>       ; ESI = Logical DOS Drv. Desc. Tbl. address
33405                              <1>       ; EAX = Disk address
33406                              <1>       ; EBX = Sector Buffer
33407                              <1>       ; ECX = sectors per cluster
33408 0000B7C6 E8C4390000          <1>       call   disk_write
33409 0000B7CB 7211                <1>       jc     short loc_createfile_dsk_wr_err
33410                              <1>
33411                              <1> loc_createfile_update_fat_cluster:
33412                              <1>       ; 21/03/2016
33413 0000B7CD 803D[FA5A0100]00    <1>       cmp    byte [createfile_wfc], 0
33414 0000B7D4 7712                <1>       ja     short loc_createfile_update_fat_cluster_n1
33415                              <1>
33416 0000B7D6 FE05[FA5A0100]      <1>       inc    byte [createfile_wfc] ; 1
33417 0000B7DC EB24                <1>       jmp    short loc_createfile_update_fat_cluster_n2
33418                              <1>
33419                              <1> loc_createfile_dsk_wr_err:
33420                              <1>       ; 16/10/2016 (1Dh -> 18)
33421                              <1>       ; 23/03/2016
33422 0000B7DE B812000000          <1>       mov    eax, 18 ; Drive not ready or write error !
33423 0000B7E3 E9BD000000          <1>       jmp    loc_createfile_stc_retn
33424                              <1>
33425                              <1> loc_createfile_update_fat_cluster_n1:
33426 0000B7E8 A1[EC5A0100]        <1>       mov    eax, [createfile_PCluster]
33427 0000B7ED 8B0D[E85A0100]      <1>       mov    ecx, [createfile_Cluster]
33428 0000B7F3 E84F040000          <1>       call   update_cluster
33429 0000B7F8 7308                <1>       jnc    short loc_createfile_update_fat_cluster_n2
33430 0000B7FA 09C0                <1>       or     eax, eax ; EAX = 0 -> cluster value is 0 or eocc
33431 0000B7FC 0F85A3000000        <1>       jnz    loc_createfile_stc_retn
33432                              <1>
33433                              <1> loc_createfile_update_fat_cluster_n2:
33434 0000B802 A1[E85A0100]        <1>         mov  eax, [createfile_Cluster]
33435 0000B807 B9FFFFFF0F          <1>       mov    ecx, 0FFFFFFFh
33436 0000B80C E836040000          <1>       call   update_cluster
33437 0000B811 734E                <1>       jnc    short loc_createfile_save_fat_buffer_3
33438 0000B813 09C0                <1>       or     eax, eax ; EAX = 0 -> cluster value is 0 or eocc
33439 0000B815 744A                <1>       jz     short loc_createfile_save_fat_buffer_3
33440                              <1>
33441                              <1> loc_createfile_upd_fat_fcluster_stc_retn:
33442 0000B817 E989000000          <1>       jmp    loc_createfile_stc_retn
33443                              <1>
33444                              <1> loc_createfile_get_set_wfc_cont:
33445                              <1>       ;movzx ecx, word [esi+LD_BPB+BytesPerSec] ; 512
33446 0000B81C 0FB70D[F85A0100]    <1>       movzx  ecx, word [createfile_BytesPerSec] ; 512
33447 0000B823 01C8                <1>       add    eax, ecx
33448 0000B825 48                  <1>       dec    eax  ; add eax, 511
33449 0000B826 29D2                <1>       sub    edx, edx
33450 0000B828 F7F1                <1>       div    ecx
```

```
33451 0000B82A 0FB61D[F15A0100]    <1>       movzx  ebx, byte [createfile_SecPerClust]
33452 0000B831 01D8                <1>       add    eax, ebx
33453 0000B833 48                  <1>       dec    eax  ; add eax, SecPerClust - 1
33454 0000B834 6631D2              <1>       xor    dx, dx
33455 0000B837 F7F3                <1>       div    ebx
33456                              <1>
33457                              <1> loc_createfile_set_cluster_count:
33458 0000B839 A3[F45A0100]        <1>       mov    [createfile_CCount], eax
33459                              <1>
33460 0000B83E BF00000700          <1>       mov    edi, Cluster_Buffer
33461 0000B843 89C8                <1>       mov    eax, ecx ; Bytes per Sector
33462 0000B845 F7E3                <1>       mul    ebx ; Sectors per Cluster
33463                              <1>       ; EAX = Bytes per Cluster
33464 0000B847 89C1                <1>       mov    ecx, eax
33465 0000B849 C1E902              <1>       shr    ecx, 2 ; dword count
33466 0000B84C 31C0                <1>       xor    eax, eax
33467 0000B84E F3AB                <1>       rep    stosd ; clear cluster buffer
33468                              <1>
33469 0000B850 A1[E85A0100]        <1>       mov    eax, [createfile_Cluster] ; 24/03/2016
33470                              <1>
33471 0000B855 89D9                <1>       mov    ecx, ebx
33472                              <1>
33473                              <1> loc_createfile_get_set_wf_fclust_cont:
33474 0000B857 83E802              <1>       sub    eax, 2
33475 0000B85A F7E1                <1>       mul    ecx
33476                              <1>       ; EAX = Logical DOS disk address (offset)
33477 0000B85C E95DFFFFFF          <1>         jmp     loc_createfile_write_fcluster_to_disk
33478                              <1>
33479                              <1> loc_createfile_save_fat_buffer_3:
33480                              <1>       ; byte [FAT_BuffValidData] = 2
33481 0000B861 E89E060000          <1>       call   save_fat_buffer
33482 0000B866 723D                <1>       jc     loc_createfile_stc_retn
33483                              <1>
33484                              <1>       ; 21/03/2016
33485 0000B868 803D[AE560100]01    <1>       cmp    byte [FAT_ClusterCounter], 1
33486 0000B86F 721B                <1>       jb     short loc_createfile_save_fat_buffer_4
33487                              <1>
33488                              <1>       ; ESI = Logical DOS Drive Description Table address
33489 0000B871 A1[AE560100]        <1>       mov    eax, [FAT_ClusterCounter]
33490 0000B876 66BB01FF            <1>       mov    bx, 0FF01h ; add free clusters
33491 0000B87A E81A070000          <1>       call   calculate_fat_freespace
33492                              <1>
33493                              <1>       ;inc   eax ; 0FFFFFFFFh -> 0 ; recalculation is needed!
33494                              <1>       ;jnz   short loc_createfile_save_fat_buffer_4
33495                              <1>
33496                              <1>       ; ecx > 0 -> Recalculation is needed
33497 0000B87F 09C9                <1>       or     ecx, ecx
33498 0000B881 7409                <1>       jz     short loc_createfile_save_fat_buffer_4
33499                              <1>
33500 0000B883 66BB00FF            <1>       mov    bx, 0FF00h ; ; recalculate free space
33501 0000B887 E80D070000          <1>       call   calculate_fat_freespace
33502                              <1>
33503                              <1> loc_createfile_save_fat_buffer_4:
33504 0000B88C FF0D[F45A0100]      <1>       dec    dword [createfile_CCount]
33505                              <1>       ;jz    short loc_createfile_upd_dir_modif_date_time
33506 0000B892 743F                <1>       jz     short loc_createfile_stc_retn_cc ; 31/03/2016
33507                              <1>
33508                              <1> loc_createfile_get_set_write_next_cluster:
33509 0000B894 E8DB020000          <1>       call   get_first_free_cluster
33510 0000B899 720A                <1>       jc     short loc_createfile_stc_retn
33511                              <1>
33512                              <1> loc_createfile_get_set_write_next_cluster_1:
33513 0000B89B 83F8FF              <1>       cmp    eax, 0FFFFFFFFh
33514 0000B89E 7213                <1>       jb     short loc_createfile_get_set_write_next_cluster_2
33515                              <1>
33516                              <1> loc_createfile_wnc_insufficient_disk_space:
33517 0000B8A0 B827000000          <1>       mov    eax, 27h ; Insufficient disk space
33518                              <1>
33519                              <1> loc_createfile_stc_retn:
33520 0000B8A5 803D[FA5A0100]01    <1>       cmp    byte [createfile_wfc], 1
33521 0000B8AC 7324                <1>       jnb    short loc_createfile_err_retn
33522 0000B8AE C3                  <1>       retn
33523                              <1>
33524                              <1> loc_createfile_wnc_inv_format_retn:
33525                              <1>       ;mov   eax, 28
33526 0000B8AF B01C                <1>       mov    al, 28 ; Invalid format
33527 0000B8B1 EBF2                <1>       jmp    short loc_createfile_stc_retn
33528                              <1>
33529                              <1> loc_createfile_get_set_write_next_cluster_2:
33530 0000B8B3 83F802              <1>       cmp    eax, 2
33531 0000B8B6 72F7                <1>       jb     short loc_createfile_wnc_inv_format_retn
33532                              <1>
33533                              <1> loc_createfile_get_set_write_next_cluster_3:
33534 0000B8B8 8B0D[E85A0100]      <1>       mov    ecx, [createfile_Cluster]
33535 0000B8BE A3[E85A0100]        <1>       mov    [createfile_Cluster], eax
33536 0000B8C3 890D[EC5A0100]      <1>       mov    [createfile_PCluster], ecx
33537 0000B8C9 0FB60D[F15A0100]    <1>       movzx  ecx, byte [createfile_SecPerClust]
33538 0000B8D0 EB85                <1>       jmp    short loc_createfile_get_set_wf_fclust_cont
33539                              <1>
33540                              <1> loc_createfile_err_retn:
33541 0000B8D2 F9                  <1>       stc
33542                              <1>
33543                              <1> ;loc_createfile_upd_dir_modif_date_time:
33544                              <1> loc_createfile_stc_retn_cc: ; 31/03/2016
33545 0000B8D3 9C                  <1>       pushf  ; cpu is here for an error return or completion
33546 0000B8D4 50                  <1>       push   eax ; error code if cf = 1
33547                              <1>
33548                              <1>       ;call  update_parent_dir_lmdt
33549                              <1>
33550                              <1> ;loc_createfile_stc_retn_cc:
33551 0000B8D5 A1[AE560100]        <1>       mov    eax, [FAT_ClusterCounter]
33552 0000B8DA 09C0                <1>       or     eax, eax
```

```
33553 0000B8DC 741A                <1>       jz     short loc_createfile_stc_retn_pop_eax
33554 0000B8DE 8A3D[8E4E0100]      <1>       mov    bh, [Current_Drv]
33555 0000B8E4 B301                <1>       mov    bl, 01h ; BL = 1 -> add clusters
33556                              <1>       ; NOTE: EAX value will be added to Free Cluster Count
33557                              <1>       ; (If EAX value is negative, Free Cluster Count will be decreased)
33558 0000B8E6 E8AE060000          <1>       call   calculate_fat_freespace
33559                              <1>         ; ESI = Logical DOS Drive Description Table Address
33560                              <1>         ;jc  short loc_createfile_stc_retn_pop_eax_cf
33561 0000B8EB 21C9                <1>       and    ecx, ecx ; cx = 0 -> valid free sector count
33562 0000B8ED 7409                <1>       jz     short loc_createfile_stc_retn_pop_eax
33563                              <1>
33564                              <1> loc_createfile_stc_retn_recalc_FAT_freespace:
33565 0000B8EF 66BB00FF            <1>       mov    bx, 0FF00h ; bh = 0FFh ->
33566                              <1>       ; ESI = Logical DOS Drv DT Addr
33567                              <1>       ; BL = 0 -> Recalculate
33568 0000B8F3 E8A1060000          <1>       call   calculate_fat_freespace
33569                              <1>
33570                              <1> loc_createfile_stc_retn_pop_eax:
33571 0000B8F8 58                  <1>       pop    eax
33572 0000B8F9 9D                  <1>       popf
33573 0000B8FA 7218                <1>       jc     short loc_createfile_retn
33574                              <1>
33575                              <1> loc_createfile_retn_fcluster:
33576 0000B8FC A1[E05A0100]        <1>       mov    eax, [createfile_FFCluster]
33577 0000B901 BB[DC5A0100]        <1>       mov    ebx, createfile_size
33578                              <1>       ;movzx ecx, byte [esi+LD_BPB+SecPerClust]
33579 0000B906 0FB60D[F15A0100]    <1>       movzx  ecx, byte [createfile_SecPerClust] ; 23/03/2016
33580 0000B90D 0FB715[F25A0100]    <1>       movzx  edx, word [createfile_DirIndex]
33581                              <1>
33582                              <1> loc_createfile_retn:
33583 0000B914 C3                  <1>       retn
33584                              <1>
33585                              <1> create_fs_file:
33586                              <1>       ; temporary (21/03/2016)
33587 0000B915 C3                  <1>       retn
33588                              <1>
33589                              <1> delete_fs_file:
33590                              <1>       ; temporary (28/02/2016)
33591 0000B916 C3                  <1>       retn
33592                              <1>
33593                              <1> rename_fs_file_or_directory:
33594 0000B917 C3                  <1>       retn
33595                              <1>
33596                              <1> make_fs_directory:
33597                              <1>       ; temporary (21/02/2016)
33598 0000B918 C3                  <1>       retn
33599                              <1>
33600                              <1> add_new_fs_section:
33601                              <1>       ; temporary (11/03/2016)
33602 0000B919 C3                  <1>       retn
33603                              <1>
33604                              <1> delete_fs_directory_entry:
33605                              <1>       ; temporary (11/03/2016)
33606 0000B91A C3                  <1>       retn
33607                              <1>
33608                              <1> csftdf2_read_fs_file_sectors:
33609                              <1>       ; temporary (19/03/2016)
33610 0000B91B C3                  <1>       retn
33611                              <1>
33612                              <1> csftdf2_write_fs_file_sectors:
33613                              <1>       ; temporary (19/03/2016)
33614 0000B91C C3                  <1>       retn
33615                                  %include 'trdosk5.s' ; 24/01/2016
33616                              <1> ; ****************************************************************************
33617                              <1> ; TRDOS386.ASM (TRDOS 386 Kernel - v2.0.0) - File System Procedures : trdosk5s
33618                              <1> ; ----------------------------------------------------------------------------
33619                              <1> ; Last Update: 23/10/2016
33620                              <1> ; ----------------------------------------------------------------------------
33621                              <1> ; Beginning: 24/01/2016
33622                              <1> ; ----------------------------------------------------------------------------
33623                              <1> ; Assembler: NASM version 2.11 (trdos386.s)
33624                              <1> ; ----------------------------------------------------------------------------
33625                              <1> ; Derived from TRDOS Operating System v1.0 (8086) source code by Erdogan Tan
33626                              <1> ; DRV_FAT.ASM (21/08/2011)
33627                              <1> ; ****************************************************************************
33628                              <1> ; DRV_FAT.ASM (c) 2005-2011 Erdogan TAN [ 07/07/2009 ] Last Update: 21/08/2011
33629                              <1>
33630                              <1> get_next_cluster:
33631                              <1>       ; 15/10/2016
33632                              <1>       ; 23/03/2016
33633                              <1>       ; 01/02/2016 (TRDOS 386 =  TRDOS v2.0)
33634                              <1>       ; 05/07/2011
33635                              <1>       ; 07/07/2009
33636                              <1>       ; 2005
33637                              <1>       ; INPUT ->
33638                              <1>       ;     EAX = Cluster Number (32 bit)
33639                              <1>       ;     ESI = Logical DOS Drive Parameters Table
33640                              <1>       ; OUTPUT ->
33641                              <1>       ;     cf = 0 -> No Error, EAX valid
33642                              <1>       ;     cf = 1 & EAX = 0 -> End Of Cluster Chain
33643                              <1>       ;     cf = 1 & EAX > 0 -> Error
33644                              <1>       ;     ECX = Current/Previous cluster (if CF = 0)
33645                              <1>       ;     EAX = Next Cluster Number (32 bit)
33646                              <1>       ;
33647                              <1>       ; (Modified registers: EAX, ECX, EBX, EDX)
33648                              <1>
33649 0000B91D A3[A2560100]        <1>       mov    [FAT_CurrentCluster], eax
33650                              <1> check_next_cluster_fat_type:
33651 0000B922 29D2                <1>       sub    edx, edx ; 0
33652 0000B924 807E0302            <1>         cmp     byte [esi+LD_FATType], 2
33653 0000B928 7250                <1>       jb     short get_FAT12_next_cluster
33654 0000B92A 0F87AF000000        <1>         ja      get_FAT32_next_cluster
```

```
33655                                  <1> get_FAT16_next_cluster:
33656 0000B930 BB00030000              <1>       mov    ebx, 300h ;768
33657 0000B935 F7F3                    <1>       div    ebx
33658                                  <1>       ; EAX = Count of 3 FAT sectors
33659                                  <1>       ; EDX = Cluster Offset (< 768)
33660 0000B937 66D1E2                  <1>       shl    dx, 1 ; Multiply by 2
33661 0000B93A 89D3                    <1>       mov    ebx, edx ; Byte Offset
33662 0000B93C 81C3001C0900            <1>       add    ebx, FAT_Buffer
33663 0000B942 66BA0300                <1>       mov    dx, 3
33664 0000B946 F7E2                    <1>       mul    edx
33665                                  <1>       ; EAX = FAT Sector (<= 256)
33666                                  <1>       ; EDX = 0
33667 0000B948 8A0E                    <1>       mov    cl, [esi+LD_Name]
33668 0000B94A 803D[A6560100]00        <1>       cmp    byte [FAT_BuffValidData], 0
33669 0000B951 0F86CC000000            <1>         jna     load_FAT_sectors0
33670 0000B957 3A0D[A7560100]          <1>       cmp    cl, [FAT_BuffDrvName]
33671 0000B95D 0F85C0000000            <1>         jne     load_FAT_sectors0
33672 0000B963 3B05[AA560100]          <1>       cmp    eax, [FAT_BuffSector]
33673 0000B969 0F85BA000000            <1>         jne     load_FAT_sectors1
33674                                  <1>       ;movzx eax, word [ebx]
33675 0000B96F 668B03                  <1>       mov    ax, [ebx]
33676                                  <1>       ; 01/02/2016
33677                                  <1>       ; DRV_FAT.ASM (21/08/2011) had a FATal bug here !
33678                                  <1>       ; (cmp ah, 0Fh) ! (ax >= FF7h)
33679                                  <1>       ; (how can i do a such mistake!?)
33680                                  <1>       ;cmp   al, 0F7h
33681                                  <1>       ;jb    short loc_pass_gnc_FAT16_eoc_check
33682                                  <1>       ;cmp   ah, 0FFh
33683                                  <1>       ;jb    short loc_pass_gnc_FAT16_eoc_check
33684 0000B972 6683F8F7                <1>       cmp    ax, 0FFF7h
33685 0000B976 725A                    <1>       jb     short loc_pass_gnc_FAT16_eoc_check
33686                                  <1>       ; ax >= FFF7h (cluster 0002h to FFF6h is valid, in use)
33687 0000B978 EB56                    <1>       jmp    short loc_pass_gnc_FAT16_eoc_check_xor_eax
33688                                  <1>
33689                                  <1> get_FAT12_next_cluster:
33690 0000B97A BB00040000              <1>       mov    ebx, 400h ;1024
33691 0000B97F F7F3                    <1>       div    ebx
33692                                  <1>       ; EAX = Count of 3 FAT sectors
33693                                  <1>       ; EDX = Cluster Offset (< 1024)
33694 0000B981 6650                    <1>       push   ax
33695 0000B983 66B80300                <1>       mov    ax, 3
33696 0000B987 66F7E2                  <1>       mul    dx     ; Multiply by 3
33697 0000B98A 66D1E8                  <1>       shr    ax, 1  ; Divide by 2
33698 0000B98D 6689C3                  <1>         mov  bx, ax       ; Byte Offset
33699 0000B990 81C3001C0900            <1>       add    ebx, FAT_Buffer
33700 0000B996 6658                    <1>       pop    ax
33701 0000B998 66BA0300                <1>       mov    dx, 3
33702 0000B99C F7E2                    <1>       mul    edx
33703                                  <1>       ; EAX = FAT Sector (<= 12)
33704                                  <1>       ; EDX = 0
33705 0000B99E 8A0E                    <1>       mov    cl, [esi+LD_Name]
33706 0000B9A0 803D[A6560100]00        <1>       cmp    byte [FAT_BuffValidData], 0
33707 0000B9A7 767A                    <1>       jna    short load_FAT_sectors0
33708 0000B9A9 3A0D[A7560100]          <1>       cmp    cl, [FAT_BuffDrvName]
33709 0000B9AF 7572                    <1>       jne    short load_FAT_sectors0
33710 0000B9B1 3B05[AA560100]          <1>       cmp    eax, [FAT_BuffSector]
33711 0000B9B7 7570                    <1>       jne    short load_FAT_sectors1
33712 0000B9B9 A1[A2560100]            <1>       mov    eax, [FAT_CurrentCluster]
33713 0000B9BE 66D1E8                  <1>       shr    ax, 1
33714                                  <1>       ;movzx eax, word [ebx]
33715 0000B9C1 668B03                  <1>       mov    ax, [ebx]
33716 0000B9C4 7314                    <1>       jnc    short get_FAT12_nc_even
33717 0000B9C6 66C1E804                <1>       shr    ax, 4
33718                                  <1> loc_gnc_fat12_eoc_check:
33719                                  <1>       ;cmp   al, 0F7h
33720                                  <1>       ;jb    short loc_pass_gnc_FAT16_eoc_check
33721                                  <1>       ;cmp   ah, 0Fh
33722                                  <1>       ;jb    short loc_pass_gnc_FAT16_eoc_check
33723 0000B9CA 663DF70F                <1>       cmp    ax, 0FF7h
33724 0000B9CE 7202                    <1>       jb     short loc_pass_gnc_FAT16_eoc_check
33725                                  <1>       ; ax >= FF7h (cluster 0002h to FF6h is valid, in use)
33726                                  <1>
33727                                  <1> loc_pass_gnc_FAT16_eoc_check_xor_eax:
33728 0000B9D0 31C0                    <1>       xor    eax, eax ; 0
33729                                  <1> loc_pass_gnc_FAT16_eoc_check:
33730                                  <1> loc_pass_gnc_FAT32_eoc_check:
33731 0000B9D2 8B0D[A2560100]          <1>       mov    ecx, [FAT_CurrentCluster]
33732 0000B9D8 F5                      <1>       cmc
33733 0000B9D9 C3                      <1>       retn
33734                                  <1>
33735                                  <1> get_FAT12_nc_even:
33736 0000B9DA 80E40F                  <1>       and    ah, 0Fh
33737 0000B9DD EBEB                    <1>       jmp    short loc_gnc_fat12_eoc_check
33738                                  <1>
33739                                  <1> get_FAT32_next_cluster:
33740 0000B9DF BB80010000              <1>       mov    ebx, 180h ;384
33741 0000B9E4 F7F3                    <1>       div    ebx
33742                                  <1>       ; EAX = Count of 3 FAT sectors
33743                                  <1>       ; EDX = Cluster Offset (< 384)
33744 0000B9E6 66C1E202                <1>       shl    dx, 2  ; Multiply by 4
33745 0000B9EA 89D3                    <1>       mov    ebx, edx ; Byte Offset
33746 0000B9EC 81C3001C0900            <1>       add    ebx, FAT_Buffer
33747 0000B9F2 66BA0300                <1>       mov    dx, 3
33748 0000B9F6 F7E2                    <1>       mul    edx
33749                                  <1>         ; EAX = FAT Sector (<= 2097152) ; (FFFFFF7h * 4) / 512
33750                                  <1>       ;      for 32KB cluster size:
33751                                  <1>       ;      EAX <= 1024 = (4GB / 32KB) * 4) / 512
33752                                  <1>       ; EDX = 0
33753 0000B9F8 8A0E                    <1>       mov    cl, [esi+LD_Name]
33754 0000B9FA 803D[A6560100]00        <1>       cmp    byte [FAT_BuffValidData], 0
33755 0000BA01 7620                    <1>       jna    short load_FAT_sectors0
33756 0000BA03 3A0D[A7560100]          <1>       cmp    cl, [FAT_BuffDrvName]
```

```
33757 0000BA09 7518              <1>       jne    short load_FAT_sectors0
33758 0000BA0B 3B05[AA560100]    <1>       cmp    eax, [FAT_BuffSector] ; 0, 3, 6, 9 ...
33759 0000BA11 7516              <1>       jne    short load_FAT_sectors1
33760 0000BA13 8B03              <1>       mov    eax, [ebx]
33761 0000BA15 25FFFFFF0F        <1>       and    eax, 0FFFFFFFh ; 28 bit Cluster
33762 0000BA1A 3DF7FFFF0F        <1>       cmp    eax, 0FFFFFF7h
33763 0000BA1F 72B1              <1>       jb     short loc_pass_gnc_FAT32_eoc_check
33764                            <1>       ; eax >= FFFFFF7h (cluster 0002h to FFFFFF6h is valid)
33765 0000BA21 EBAD              <1>       jmp    short loc_pass_gnc_FAT16_eoc_check_xor_eax
33766                            <1>
33767                            <1> load_FAT_sectors0:
33768 0000BA23 880D[A7560100]    <1>       mov    [FAT_BuffDrvName], cl
33769                            <1> load_FAT_sectors1:
33770 0000BA29 A3[AA560100]      <1>       mov    [FAT_BuffSector], eax
33771 0000BA2E 89C3              <1>       mov    ebx, eax
33772 0000BA30 034660            <1>         add     eax, [esi+LD_FATBegin]
33773 0000BA33 807E0302          <1>       cmp    byte [esi+LD_FATType], 2
33774 0000BA37 7706              <1>         ja      short load_FAT_sectors3
33775 0000BA39 0FB74E1C          <1>       movzx  ecx, word [esi+LD_BPB+BPB_FATSz16]
33776 0000BA3D EB03              <1>       jmp    short load_FAT_sectors4
33777                            <1> load_FAT_sectors3:
33778 0000BA3F 8B4E2A            <1>       mov    ecx, [esi+LD_BPB+BPB_FATSz32]
33779                            <1> load_FAT_sectors4:
33780 0000BA42 29D9              <1>       sub    ecx, ebx ; [FAT_BuffSector]
33781 0000BA44 83F903            <1>         cmp     ecx, 3
33782 0000BA47 7605              <1>         jna     short load_FAT_sectors5
33783 0000BA49 B903000000        <1>       mov    ecx, 3
33784                            <1> load_FAT_sectors5:
33785 0000BA4E BB001C0900        <1>       mov    ebx, FAT_Buffer
33786 0000BA53 E846370000        <1>       call   disk_read
33787 0000BA58 730D              <1>       jnc    short load_FAT_sectors_ok
33788                            <1>       ; 15/10/2016 (15h -> 17)
33789                            <1>       ; 23/03/2016 (15h)
33790 0000BA5A B811000000        <1>       mov    eax, 17 ; Drive not ready or read error
33791 0000BA5F C605[A6560100]00  <1>       mov    byte [FAT_BuffValidData], 0
33792 0000BA66 C3                <1>       retn
33793                            <1> load_FAT_sectors_ok:
33794 0000BA67 C605[A6560100]01  <1>       mov    byte [FAT_BuffValidData], 1
33795 0000BA6E A1[A2560100]      <1>       mov    eax, [FAT_CurrentCluster]
33796 0000BA73 E9AAFEFFFF        <1>         jmp     check_next_cluster_fat_type
33797                            <1>
33798                            <1> load_FAT_root_directory:
33799                            <1>       ; 23/10/2016
33800                            <1>       ; 15/10/2016
33801                            <1>       ; 07/02/2016
33802                            <1>       ; 02/02/2016
33803                            <1>       ; 01/02/2016 (TRDOS 386 =  TRDOS v2.0)
33804                            <1>       ; 21/05/2011
33805                            <1>       ; 22/08/2009
33806                            <1>       ;
33807                            <1>       ; INPUT ->
33808                            <1>       ;     ESI = Logical DOS Drive Description Table
33809                            <1>       ; OUTPUT ->
33810                            <1>       ;     cf = 1 -> Root directory could not be loaded
33811                            <1>       ;         EAX > 0 -> Error number
33812                            <1>       ;     cf = 0 -> EAX = 0
33813                            <1>       ;     ECX = Directory buffer size in sectors (CL)
33814                            <1>       ;     EBX = Directory buffer address
33815                            <1>       ;     NOTE: DirBuffer_Size is in bytes ! (word)
33816                            <1>       ;
33817                            <1>       ; (Modified registers: EAX, ECX, EBX, EDX)
33818                            <1>
33819                            <1>       ; NOTE: Only for FAT12 and FAT16 file systems !
33820                            <1>       ; (FAT32 fs root dir must be loaded as sub directory)
33821                            <1>
33822 0000BA78 8A1E              <1>       mov    bl, [esi+LD_Name]
33823 0000BA7A 8A7E03            <1>       mov    bh, [esi+LD_FATType]
33824                            <1>
33825                            <1>       ;mov   [DirBuff_DRV], bl
33826                            <1>       ;mov   [DirBuff_FATType], bh
33827 0000BA7D 66891D[B6560100]  <1>       mov    [DirBuff_DRV], bx
33828                            <1>
33829                            <1>       ;cmp   bh, 2
33830                            <1>       ;ja    short load_FAT32_root_dir0 ; FAT32 root dir
33831                            <1>
33832                            <1> load_FAT_root_dir0: ; 23/10/2016
33833 0000BA84 0FB75617          <1>       movzx  edx, word [esi+LD_BPB+RootDirEnts]
33834                            <1>
33835                            <1>       ;or    dx, dx ; 0 for FAT32 file systems
33836                            <1>       ;jz    short load_FAT32_root_dir0 ; FAT32 root dir
33837                            <1>
33838 0000BA88 6681FA0002        <1>       cmp    dx, 512 ; Number of Root Dir Entries
33839 0000BA8D 7414              <1>       je     short lrd_mov_ecx_32
33840 0000BA8F 89D0              <1>       mov    eax, edx
33841                            <1>       ; 23/10/2016
33842 0000BA91 89C1              <1>       mov    ecx, eax
33843 0000BA93 6683C10F          <1>       add    cx, 15 ; round up
33844 0000BA97 66C1E904          <1>       shr    cx, 4  ; 16 entries per sector (512/32)
33845                            <1>       ; ecx = Root directory size in sectors
33846 0000BA9B 66C1E005          <1>       shl    ax, 5 ; Root directory size in bytes
33847 0000BA9F 664A              <1>       dec    dx    ; Last entry number of root dir
33848                            <1>       ; cx = Dir Buffer sector count
33849 0000BAA1 EB0B              <1>       jmp    short lrd_check_dir_buffer
33850                            <1>
33851                            <1> lrd_mov_ecx_32:
33852 0000BAA3 B920000000        <1>       mov    ecx, 32
33853 0000BAA8 664A              <1>       dec    dx ; 511
33854 0000BAAA 66B80040          <1>       mov    ax, 32*512
33855                            <1>
33856                            <1> lrd_check_dir_buffer:
33857 0000BAAE 29DB              <1>       sub    ebx, ebx ; 0
33858 0000BAB0 881D[B8560100]    <1>       mov    [DirBuff_ValidData], bl ; 0
```

```
33859 0000BAB6 668915[BB560100]    <1>       mov    [DirBuff_LastEntry], dx
33860 0000BABD 891D[BD560100]      <1>       mov    [DirBuff_Cluster], ebx ; 0
33861 0000BAC3 66A3[C1560100]      <1>       mov    [DirBuffer_Size], ax
33862                              <1>
33863 0000BAC9 8B4664              <1>       mov    eax, [esi+LD_ROOTBegin]
33864                              <1> read_directory:
33865 0000BACC BB00000800          <1>       mov    ebx, Directory_Buffer
33866 0000BAD1 51                  <1>       push   ecx ; Directory buffer sector count
33867 0000BAD2 53                  <1>       push   ebx
33868 0000BAD3 E8C6360000          <1>       call   disk_read
33869 0000BAD8 5B                  <1>       pop    ebx
33870 0000BAD9 720B                <1>       jc     short load_DirBuff_error
33871                              <1>
33872                              <1> validate_DirBuff_and_return:
33873 0000BADB 59                  <1>       pop    ecx ; Number of loaded sectors
33874 0000BADC C605[B8560100]01    <1>       mov    byte [DirBuff_ValidData], 1
33875 0000BAE3 31C0                <1>       xor    eax, eax ; 0 = no error
33876 0000BAE5 C3                  <1>       retn
33877                              <1>
33878                              <1> load_DirBuff_error:
33879 0000BAE6 89C8                <1>       mov    eax, ecx ; remaining sectors
33880 0000BAE8 59                  <1>       pop    ecx ; sector count
33881 0000BAE9 29C1                <1>       sub    ecx, eax ; Number of loaded sectors
33882                              <1>       ; 15/10/2016 (15h -> 17)
33883 0000BAEB B811000000          <1>       mov    eax, 17 ; DRV NOT READY OR READ ERROR !
33884 0000BAF0 F9                  <1>       stc
33885 0000BAF1 C3                  <1>         retn
33886                              <1>
33887                              <1> load_FAT32_root_directory:
33888                              <1>       ; 02/02/2016 (TRDOS 386 =  TRDOS v2.0)
33889                              <1>       ;
33890                              <1>       ; INPUT ->
33891                              <1>       ;      ESI = Logical DOS Drive Description Table
33892                              <1>       ; OUTPUT ->
33893                              <1>       ;      cf = 1 -> Root directory could not be loaded
33894                              <1>       ;          EAX > 0 -> Error number
33895                              <1>       ;      cf = 0 -> EAX = 0
33896                              <1>       ;      ECX = Directory buffer size in sectors (CL)
33897                              <1>       ;      EBX = Directory buffer address
33898                              <1>       ;      NOTE: DirBuffer_Size is in bytes ! (word)
33899                              <1>       ;
33900                              <1>       ; (Modified registers: EAX, ECX, EBX, EDX)
33901                              <1>
33902                              <1>
33903 0000BAF2 8A1E                <1>       mov    bl, [esi+LD_Name]
33904 0000BAF4 8A7E03              <1>       mov    bh, [esi+LD_FATType]
33905                              <1>
33906                              <1>       ;mov   [DirBuff_DRV], bl
33907                              <1>       ;mov   [DirBuff_FATType], bh
33908 0000BAF7 66891D[B6560100]    <1>       mov    [DirBuff_DRV], bx
33909                              <1>
33910                              <1> load_FAT32_root_dir0:
33911 0000BAFE 8B4632              <1>       mov    eax, [esi+LD_BPB+FAT32_RootFClust]
33912 0000BB01 EB0C                <1>       jmp    short load_FAT_sub_dir0
33913                              <1>
33914                              <1> load_FAT_sub_directory:
33915                              <1>       ; 01/02/2016 (TRDOS 386 =  TRDOS v2.0)
33916                              <1>       ; 05/07/2011
33917                              <1>       ; 23/08/2009
33918                              <1>       ;
33919                              <1>       ; INPUT ->
33920                              <1>       ;      ESI = Logical DOS Drive Description Table
33921                              <1>       ;      EAX = Cluster Number
33922                              <1>       ; OUTPUT ->
33923                              <1>       ;      cf = 1 -> Sub directory could not be loaded
33924                              <1>       ;          EAX > 0 -> Error number
33925                              <1>       ;      cf = 0 -> EAX = 0
33926                              <1>       ;      ECX = Directory buffer size in sectors (CL)
33927                              <1>       ;      EBX = Directory buffer address
33928                              <1>       ;
33929                              <1>       ;      NOTE: DirBuffer_Size is in bytes ! (word)
33930                              <1>       ;
33931                              <1>       ; (Modified registers: EAX, ECX, EBX, EDX)
33932                              <1>
33933 0000BB03 8A1E                <1>       mov    bl, [esi+LD_Name]
33934 0000BB05 8A7E03              <1>       mov    bh, [esi+LD_FATType]
33935                              <1>
33936                              <1>       ;mov   [DirBuff_DRV], bl
33937                              <1>       ;mov   [DirBuff_FATType], bh
33938 0000BB08 66891D[B6560100]    <1>       mov    [DirBuff_DRV], bx
33939                              <1>
33940                              <1> load_FAT_sub_dir0:
33941 0000BB0F 0FB64E13            <1>       movzx  ecx, byte [esi+LD_BPB+SecPerClust]
33942                              <1>
33943 0000BB13 882D[B8560100]      <1>       mov    [DirBuff_ValidData], ch ; 0
33944 0000BB19 A3[BD560100]        <1>       mov    [DirBuff_Cluster], eax
33945                              <1>
33946 0000BB1E 0FB74611            <1>       movzx  eax, word [esi+LD_BPB+BytesPerSec]
33947 0000BB22 F7E1                <1>       mul    ecx
33948 0000BB24 C1E805              <1>       shr    eax, 5 ; directory entry count (dir size / 32)
33949 0000BB27 6648                <1>       dec    ax ; last entry
33950 0000BB29 66A3[BB560100]      <1>       mov    [DirBuff_LastEntry], ax
33951                              <1>
33952 0000BB2F A1[BD560100]        <1>       mov    eax, [DirBuff_Cluster]
33953 0000BB34 83E802              <1>       sub    eax, 2
33954 0000BB37 F7E1                <1>       mul    ecx
33955 0000BB39 034668              <1>       add    eax, [esi+LD_DATABegin]
33956                              <1>       ; ecx = sector per cluster (dir buffer size = 32 sectors)
33957 0000BB3C EB8E                <1>       jmp    short read_directory
33958                              <1>
33959                              <1> ; DRV_FS.ASM
33960                              <1>
```

```
33961                                  <1> load_current_FS_directory:
33962 0000BB3E C3                      <1>       retn
33963                                  <1> load_FS_root_directory:
33964 0000BB3F C3                      <1>       retn
33965                                  <1> load_FS_sub_directory:
33966 0000BB40 C3                      <1>       retn
33967                                  <1>
33968                                  <1> read_cluster:
33969                                  <1>       ; 15/10/2016
33970                                  <1>       ; 18/03/2016
33971                                  <1>       ; 16/03/2016
33972                                  <1>       ; 17/02/2016
33973                                  <1>       ; 15/02/2016 (TRDOS 386 =  TRDOS v2.0)
33974                                  <1>       ;
33975                                  <1>       ; INPUT ->
33976                                  <1>       ;     EAX = Cluster Number (Sector index for SINGLIX FS)
33977                                  <1>       ;     ESI = Logical DOS Drive Description Table address
33978                                  <1>       ;     EBX = Cluster (File R/W) Buffer address (max. 64KB)
33979                                  <1>       ;     Only for SINGLIX FS:
33980                                  <1>       ;     EDX = File Number (The 1st FDT address)
33981                                  <1>       ; OUTPUT ->
33982                                  <1>       ;     cf = 1 -> Cluster can not be loaded at the buffer
33983                                  <1>       ;         EAX > 0 -> Error number
33984                                  <1>       ;     cf = 0 -> Cluster has been loaded at the buffer
33985                                  <1>       ;
33986                                  <1>       ; (Modified registers: EAX, ECX, EBX, EDX)
33987                                  <1>
33988 0000BB41 0FB64E13                <1>       movzx  ecx, byte [esi+LD_BPB+BPB_SecPerClust]
33989                                  <1>       ; CL = 1 = [esi+LD_FS_Reserved2] ; SectPerClust for Singlix FS
33990                                  <1>
33991                                  <1> read_file_sectors: ; 16/03/2016
33992 0000BB45 807E0300                <1>       cmp    byte [esi+LD_FATType], 0
33993 0000BB49 761C                    <1>       jna    short read_fs_cluster
33994                                  <1>
33995                                  <1> read_fat_file_sectors: ; 18/03/2016
33996 0000BB4B 83E802                  <1>       sub    eax, 2 ; Beginning cluster number is always 2
33997 0000BB4E 0FB65613                <1>       movzx  edx, byte [esi+LD_BPB+BPB_SecPerClust] ; 18/03/2016
33998 0000BB52 F7E2                    <1>       mul    edx
33999 0000BB54 034668                  <1>       add    eax, [esi+LD_DATABegin] ; absolute address of the cluster
34000                                  <1>
34001                                  <1>       ; EAX = Disk sector address
34002                                  <1>       ; ECX = Sector count
34003                                  <1>       ; EBX = Buffer address
34004                                  <1>       ; (EDX = 0)
34005                                  <1>       ; ESI = Logical DOS drive description table address
34006                                  <1>
34007 0000BB57 E842360000              <1>       call   disk_read
34008 0000BB5C 7306                    <1>       jnc    short rclust_retn
34009                                  <1>
34010                                  <1>       ; 15/10/2016 (15h -> 17)
34011 0000BB5E B811000000              <1>       mov    eax, 17 ; Drive not ready or read error !
34012 0000BB63 C3                      <1>       retn
34013                                  <1>
34014                                  <1> rclust_retn:
34015 0000BB64 29C0                    <1>       sub    eax, eax ; 0
34016 0000BB66 C3                      <1>       retn
34017                                  <1>
34018                                  <1> read_fs_cluster:
34019                                  <1>       ; 15/02/2016 (TRDOS 386 =  TRDOS v2.0)
34020                                  <1>       ; Singlix FS
34021                                  <1>
34022                                  <1>       ; EAX = Cluster number is sector index number of the file (eax)
34023                                  <1>
34024                                  <1>       ; EDX = File number is the first File Descriptor Table address
34025                                  <1>       ;     of the file. (Absolute address of the FDT).
34026                                  <1>
34027                                  <1>       ; eax = sector index (0 for the first sector)
34028                                  <1>       ; edx = FDT0 address
34029                                  <1>             ; 64 KB buffer = 128 sectors (limit)
34030 0000BB67 B980000000              <1>       mov    ecx, 128 ; maximum count of sectors (before eof)
34031 0000BB6C E801000000              <1>       call   read_fs_sectors
34032 0000BB71 C3                      <1>       retn
34033                                  <1>
34034                                  <1> read_fs_sectors:
34035                                  <1>       ; 15/02/2016 (TRDOS 386 =  TRDOS v2.0)
34036 0000BB72 F9                      <1>       stc
34037 0000BB73 C3                      <1>       retn
34038                                  <1>
34039                                  <1> get_first_free_cluster:
34040                                  <1>       ; 02/03/2016
34041                                  <1>       ; 21/02/2016 (TRDOS 386 =  TRDOS v2.0)
34042                                  <1>       ; 26/10/2010 (DRV_FAT.ASM, 'proc_get_first_free_cluster')
34043                                  <1>       ; 10/07/2010
34044                                  <1>       ; INPUT ->
34045                                  <1>       ;     ESI = Logical DOS Drive Description Table address
34046                                  <1>       ; OUTPUT ->
34047                                  <1>       ;     cf = 1 -> Error code in AL (EAX)
34048                                  <1>       ;     cf = 0 ->
34049                                  <1>       ;       EAX = Cluster number
34050                                  <1>       ;       If EAX = FFFFFFFFh -> no free space
34051                                  <1>       ;     If the drive has FAT32 fs:
34052                                  <1>       ;       EBX = FAT32 FSI sector buffer address (if > 0)
34053                                  <1>
34054 0000BB74 8B4678                  <1>       mov    eax, [esi+LD_Clusters]
34055 0000BB77 40                      <1>       inc    eax ; add eax, 1
34056 0000BB78 A3[40590100]            <1>       mov    [gffc_last_free_cluster], eax
34057                                  <1>
34058 0000BB7D 31DB                    <1>       xor    ebx, ebx ; 0 ; 02/03/2016
34059                                  <1>
34060 0000BB7F 807E0302                <1>       cmp    byte [esi+LD_FATType], 2
34061 0000BB83 760E                    <1>       jna    short loc_gffc_get_first_fat_free_cluster0
34062                                  <1>
```

```
34063                                  <1> loc_gffc_get_first_fat32_free_cluster:
34064                                  <1>       ; 02/03/2016
34065 0000BB85 E844060000              <1>       call   get_fat32_fsinfo_sector_parms
34066 0000BB8A 7207                    <1>       jc     short loc_gffc_get_first_fat_free_cluster0
34067                                  <1>
34068                                  <1> loc_gffc_check_fsinfo_parms:
34069                                  <1>       ;;mov  ebx, DOSBootSectorBuff
34070                                  <1>       ;cmp   dword [ebx], 41615252h
34071                                  <1>       ;jne   short loc_gffc_fat32_fsinfo_err
34072                                  <1>       ;cmp   dword [ebx+484], 61417272h
34073                                  <1>       ;jne   short loc_gffc_fat32_fsinfo_err
34074                                  <1>       ;mov   eax, [ebx+492] ; FSI_Next_Free
34075                                  <1>       ;EAX = First free cluster
34076                                  <1>       ;(from FAT32 FSInfo sector)
34077 0000BB8C 89D0                    <1>       mov    eax, edx ; FSI_Next_Free (First Free Cluster)
34078 0000BB8E 83F8FF                  <1>       cmp    eax, 0FFFFFFFFh ; invalid (unknown) !
34079 0000BB91 7205                    <1>       jb     short loc_gffc_get_first_fat_free_cluster1
34080                                  <1>
34081                                  <1>       ; Start from the 1st cluster of the FAT(32) file system
34082                                  <1> loc_gffc_get_first_fat_free_cluster0:
34083 0000BB93 B802000000              <1>       mov    eax, 2
34084                                  <1>       ;xor   edx, edx
34085                                  <1>
34086                                  <1> loc_gffc_get_first_fat_free_cluster1:
34087 0000BB98 53                      <1>       push   ebx ; 02/03/2016
34088                                  <1>
34089                                  <1> loc_gffc_get_first_fat_free_cluster2:
34090 0000BB99 A3[3C590100]            <1>       mov    [gffc_first_free_cluster], eax
34091 0000BB9E A3[38590100]            <1>       mov    [gffc_next_free_cluster], eax
34092                                  <1>
34093                                  <1>       ; EBX = FAT32 FSINFO sector buffer address
34094                                  <1>       ; (EBX = 0, if the drive has not got FAT32 fs or
34095                                  <1>       ; FAT32 FSINFO sector buffer is invalid.)
34096                                  <1>
34097                                  <1> loc_gffc_get_first_fat_free_cluster3:
34098 0000BBA3 E875FDFFFF              <1>       call   get_next_cluster
34099 0000BBA8 7307                    <1>       jnc    short loc_gffc_get_first_fat_free_cluster4
34100 0000BBAA 09C0                    <1>       or     eax, eax
34101 0000BBAC 740B                    <1>       jz     short loc_gffc_first_free_fat_cluster_next
34102 0000BBAE 5B                      <1>       pop    ebx ; 02/03/2016
34103 0000BBAF F5                      <1>       cmc    ; stc
34104 0000BBB0 C3                      <1>       retn
34105                                  <1>
34106                                  <1> loc_gffc_get_first_fat_free_cluster4:
34107 0000BBB1 21C0                    <1>       and    eax, eax ; next cluster value
34108 0000BBB3 7504                    <1>       jnz    short loc_gffc_first_free_fat_cluster_next
34109 0000BBB5 89C8                    <1>       mov    eax, ecx ; current (previous cluster) value
34110 0000BBB7 EB22                    <1>       jmp    short loc_gffc_check_for_set
34111                                  <1>
34112                                  <1> loc_gffc_first_free_fat_cluster_next:
34113 0000BBB9 A1[38590100]            <1>       mov    eax, [gffc_next_free_cluster]
34114 0000BBBE 3B05[40590100]          <1>       cmp    eax, [gffc_last_free_cluster]
34115 0000BBC4 7308                    <1>       jnb    short retn_stc_from_get_first_free_cluster
34116                                  <1> pass_gffc_last_cluster_eax_check:
34117 0000BBC6 40                      <1>       inc    eax ; add eax, 1
34118 0000BBC7 A3[38590100]            <1>       mov    [gffc_next_free_cluster], eax
34119 0000BBCC EBD5                    <1>       jmp    short loc_gffc_get_first_fat_free_cluster3
34120                                  <1>
34121                                  <1> retn_stc_from_get_first_free_cluster:
34122 0000BBCE A1[3C590100]            <1>       mov    eax, [gffc_first_free_cluster]
34123 0000BBD3 83F802                  <1>       cmp    eax, 2
34124 0000BBD6 7709                    <1>       ja     short loc_gffc_check_previous_clusters
34125 0000BBD8 29C0                    <1>       sub    eax, eax
34126 0000BBDA 48                      <1>       dec    eax ; FFFFFFFFh
34127                                  <1>
34128                                  <1> loc_gffc_check_for_set:
34129                                  <1>       ; 02/03/2016
34130 0000BBDB 5B                      <1>       pop    ebx
34131                                  <1>
34132                                  <1>       ; EBX = FAT32 FSINFO sector buffer address
34133                                  <1>       ; (EBX = 0, if the drive has not got FAT32 fs or
34134                                  <1>       ; FAT32 FSINFO sector buffer is invalid.)
34135                                  <1>
34136 0000BBDC 09DB                    <1>       or     ebx, ebx
34137 0000BBDE 750E                    <1>       jnz    short loc_gffc_set_ffree_fat32_cluster
34138                                  <1>
34139                                  <1>       ;cmp   byte [esi+LD_FATType], 3
34140                                  <1>       ;jnb   short loc_gffc_set_ffree_fat32_cluster
34141                                  <1>
34142                                  <1>       ;xor   ebx, ebx ; 0
34143                                  <1>
34144                                  <1> loc_gffc_retn:
34145 0000BBE0 C3                      <1>       retn
34146                                  <1>
34147                                  <1> loc_gffc_check_previous_clusters:
34148 0000BBE1 48                      <1>       dec    eax ; sub eax, 1
34149 0000BBE2 A3[40590100]            <1>       mov    [gffc_last_free_cluster], eax
34150 0000BBE7 B802000000              <1>       mov    eax, 2
34151                                  <1>       ;xor   edx, edx
34152 0000BBEC EBAB                    <1>       jmp    short loc_gffc_get_first_fat_free_cluster2
34153                                  <1>
34154                                  <1> loc_gffc_set_ffree_fat32_cluster:
34155                                  <1>       ;call  set_first_free_cluster
34156                                  <1>       ;retn
34157                                  <1>       ;jmp   short set_first_free_cluster
34158                                  <1>
34159                                  <1> set_first_free_cluster:
34160                                  <1>       ; 15/10/2016
34161                                  <1>       ; 23/03/2016
34162                                  <1>       ; 02/03/2016
34163                                  <1>       ; 29/02/2016
34164                                  <1>       ; 26/02/2016
```

```
34165                                  <1>       ; 21/02/2016 (TRDOS 386 =  TRDOS v2.0)
34166                                  <1>       ; 21/08/2011 (DRV_FAT.ASM, 'proc_set_first_free_cluster')
34167                                  <1>       ; 11/07/2010
34168                                  <1>       ; INPUT ->
34169                                  <1>       ;      ESI = Logical DOS Drive Description Table address
34170                                  <1>       ;      EAX = First free cluster
34171                                  <1>       ;      EBX = FSINFO sector buffer address
34172                                  <1>       ;      ;;If EBX > 0, it is FSINFO sector buffer address
34173                                  <1>       ;      ;;EBX = 0, if FSINFO sector is not loaded
34174                                  <1>       ; OUTPUT->
34175                                  <1>       ;      ESI = Logical DOS Drive Description Table address
34176                                  <1>       ;      If EBX > 0, it is FSINFO sector buffer address
34177                                  <1>       ;      EBX = 0, if FSINFO sector could not be loaded
34178                                  <1>       ;      CF = 1 -> Error code in AL (EAX)
34179                                  <1>       ;      CF = 0 -> first free cluster is successfully updated
34180                                  <1>
34181                                  <1>       ;cmp   byte [esi+LD_FATType], 3
34182                                  <1>       ;jb    short loc_sffc_invalid_drive
34183                                  <1>
34184                                  <1>       ; Save First Free Cluster value for 'update_cluster'
34185 0000BBEE 89463E                 <1>       mov    [esi+LD_BPB+BPB_Reserved+4], eax ; First free Cluster
34186                                  <1>
34187                                  <1>       ;or    ebx, ebx
34188                                  <1>       ;jnz   short loc_sffc_read_fsinfo_sector
34189                                  <1>
34190 0000BBF1 813B52526141           <1>       cmp     dword [ebx], 41615252h
34191 0000BBF7 7540                   <1>       jne    short loc_sffc_read_fsinfo_sector
34192 0000BBF9 81BBE4010000727241-     <1>       cmp    dword [ebx+484], 61417272h
34193 0000BC02 61                      <1>
34194 0000BC03 7534                   <1>       jne    short loc_sffc_read_fsinfo_sector
34195                                  <1>
34196 0000BC05 3B83EC010000           <1>       cmp    eax, [ebx+492]  ; FSI_Next_Free
34197 0000BC0B 741F                   <1>       je     short loc_sffc_retn
34198                                  <1>
34199                                  <1> loc_sffc_write_fsinfo_sector:
34200                                  <1>       ; EBX = FSINFO sector buffer
34201                                  <1>       ; [CFS_FAT32FSINFOSEC] is set in 'get_fat32_fsinfo_sector_parms'
34202 0000BC0D 8983EC010000           <1>       mov    [ebx+492], eax
34203 0000BC13 A1[50590100]           <1>       mov    eax, [CFS_FAT32FSINFOSEC]
34204 0000BC18 B901000000             <1>       mov    ecx, 1
34205 0000BC1D 53                     <1>       push   ebx
34206 0000BC1E E86C350000             <1>       call   disk_write
34207 0000BC23 7208                   <1>       jc      short loc_sffc_read_fsinfo_sector_err1
34208 0000BC25 5B                     <1>       pop    ebx
34209                                  <1>
34210 0000BC26 8B83EC010000           <1>       mov    eax, [ebx+492] ; First (Next) Free Cluster
34211                                  <1>
34212                                  <1> loc_sffc_retn:
34213 0000BC2C C3                     <1>       retn
34214                                  <1>
34215                                  <1> ;loc_sffc_invalid_drive:
34216                                  <1> ;     mov    eax, 0Fh ; MSDOS Error : Invalid drive
34217                                  <1> ;     push   edx
34218                                  <1>
34219                                  <1> loc_sffc_read_fsinfo_sector_err1:
34220 0000BC2D BB00000000             <1>       mov    ebx, 0
34221                                  <1>       ; 15/10/2016 (1Dh -> 18)
34222                                  <1>       ; 23/03/2016 (1Dh)
34223 0000BC32 B812000000             <1>       mov    eax, 18 ; Drive not ready or write error
34224                                  <1>
34225                                  <1> loc_sffc_read_fsinfo_sector_err2:
34226 0000BC37 5A                     <1>       pop    edx
34227 0000BC38 C3                     <1>       retn
34228                                  <1>
34229                                  <1> loc_sffc_read_fsinfo_sector:
34230 0000BC39 50                     <1>       push   eax
34231                                  <1>
34232 0000BC3A E88F050000             <1>       call   get_fat32_fsinfo_sector_parms
34233 0000BC3F 72F6                   <1>       jc     short loc_sffc_read_fsinfo_sector_err2
34234                                  <1>
34235 0000BC41 58                     <1>       pop    eax
34236                                  <1>       ; EDX = First (Next) Free Cluster value from FSINFO sector
34237                                  <1>       ; EAX = First Free Cluster value from 'get_next_cluster'
34238                                  <1>       ; (edx = old value)
34239 0000BC42 39D0                   <1>       cmp    eax, edx ; First free Cluster (eax = new value)
34240 0000BC44 75C7                   <1>       jne    short loc_sffc_write_fsinfo_sector
34241                                  <1>
34242 0000BC46 C3                     <1>       retn
34243                                  <1>
34244                                  <1> update_cluster:
34245                                  <1>       ; 23/10/2016
34246                                  <1>       ; 23/03/2016
34247                                  <1>       ; 02/03/2016
34248                                  <1>       ; 01/03/2016
34249                                  <1>       ; 29/02/2016
34250                                  <1>       ; 27/02/2016
34251                                  <1>       ; 26/02/2016
34252                                  <1>       ; 22/02/2016 (TRDOS 386 =  TRDOS v2.0)
34253                                  <1>       ; 11/08/2011
34254                                  <1>       ; 09/02/2005
34255                                  <1>       ; INPUT ->
34256                                  <1>       ;      EAX = Cluster Number
34257                                  <1>       ;      ECX = New Cluster Value
34258                                  <1>       ;      ESI = Logical Dos Drive Parameters Table
34259                                  <1>       ;
34260                                  <1>       ;      /// dword [FAT_ClusterCounter] ///
34261                                  <1>       ;
34262                                  <1>       ; OUTPUT ->
34263                                  <1>       ;      cf = 0 -> No Error, EAX is valid
34264                                  <1>       ;      cf = 1 & EAX = 0 -> End Of Cluster Chain
34265                                  <1>       ;      cf = 1 & EAX > 0 -> Error
34266                                  <1>       ;            (ECX -> any value)
```

```
34267                                  <1>       ;      EAX = Next Cluster
34268                                  <1>       ;      ECX = New Cluster Value
34269                                  <1>       ;
34270                                  <1>       ;      /// [FAT_ClusterCounter] is updated,
34271                                  <1>       ;      /// decreased when a free cluster is assigned,
34272                                  <1>       ;      /// increased if an assigned cluster is freed.
34273                                  <1>       ;
34274                                  <1>       ;
34275                                  <1>       ; (Modified registers: EAX, EBX, -ECX-, EDX)
34276                                  <1>
34277 0000BC47 A3[A2560100]           <1>       mov    [FAT_CurrentCluster], eax
34278 0000BC4C 890D[44590100]         <1>       mov    [ClusterValue], ecx
34279                                  <1>
34280                                  <1> loc_update_cluster_check_fat_buffer:
34281 0000BC52 8A1E                   <1>       mov    bl, [esi+LD_Name]
34282 0000BC54 381D[A7560100]         <1>       cmp    [FAT_BuffDrvName], bl
34283 0000BC5A 741A                   <1>       je     short loc_update_cluster_check_fat_type
34284 0000BC5C 803D[A6560100]02       <1>       cmp    byte [FAT_BuffValidData], 2
34285 0000BC63 0F84C2000000           <1>         je      loc_uc_save_fat_buffer
34286                                  <1>
34287                                  <1> loc_uc_reset_fat_buffer_validation:
34288 0000BC69 C605[A6560100]00       <1>       mov    byte [FAT_BuffValidData], 0
34289                                  <1>
34290                                  <1> loc_uc_check_fat_type_reset_drvname:
34291 0000BC70 881D[A7560100]         <1>       mov    [FAT_BuffDrvName], bl
34292                                  <1>
34293                                  <1> loc_update_cluster_check_fat_type:
34294 0000BC76 29D2                   <1>       sub    edx, edx ; 26/02/2016
34295 0000BC78 8A5E03                 <1>       mov    bl, [esi+LD_FATType]
34296 0000BC7B 83F802                 <1>       cmp    eax, 2
34297 0000BC7E 0F82BE000000           <1>         jb      update_cluster_inv_data
34298 0000BC84 80FB02                 <1>       cmp    bl, 2
34299 0000BC87 0F877A010000           <1>         ja      update_fat32_cluster
34300                                  <1>       ;cmp   bl, 1
34301                                  <1>       ;jb    short update_cluster_inv_data
34302 0000BC8D 8B4E78                 <1>       mov    ecx, [esi+LD_Clusters]
34303 0000BC90 41                     <1>       inc    ecx
34304 0000BC91 890D[B2560100]         <1>       mov    [LastCluster], ecx
34305 0000BC97 39C8                   <1>       cmp    eax, ecx ; dword [LastCluster]
34306 0000BC99 0F87A6000000           <1>         ja      return_uc_fat_stc
34307                                  <1>       ; TRDOS v1 has a FATal bug here !
34308                                  <1>             ; or bl, bl ; cmp bl, 0
34309                                  <1>             ; jz short update_fat12_cluster
34310                                  <1>       ; !! It would destroy FAT12 floppy disk fs here !!
34311                                  <1>       ; ('A:' disks of TRDOS v1 operating system project
34312                                  <1>       ; had 'singlix fs', so, I could not differ this mistake
34313                                  <1>       ; on a drive 'A:')
34314 0000BC9F 80FB01                 <1>       cmp    bl, 1 ; correct comparison is this !
34315 0000BCA2 0F86A2000000           <1>         jna     update_fat12_cluster
34316                                  <1>
34317                                  <1> update_fat16_cluster:
34318                                  <1> pass_uc_fat16_errc:
34319                                  <1>       ;sub   edx, edx
34320 0000BCA8 BB00030000             <1>       mov    ebx, 300h ;768
34321 0000BCAD F7F3                   <1>       div    ebx
34322                                  <1>       ; EAX = Count of 3 FAT sectors
34323                                  <1>       ; DX = Cluster offset in FAT buffer
34324 0000BCAF 6689D3                 <1>       mov    bx, dx
34325 0000BCB2 66D1E3                 <1>       shl    bx, 1 ; Multiply by 2
34326 0000BCB5 66BA0300               <1>       mov    dx, 3
34327 0000BCB9 F7E2                   <1>       mul    edx
34328                                  <1>       ; EAX = FAT Sector
34329                                  <1>       ; EDX = 0
34330                                  <1>       ; EBX = Byte offset in FAT buffer
34331 0000BCBB 8A0D[A6560100]         <1>       mov    cl, [FAT_BuffValidData]
34332 0000BCC1 80F902                 <1>       cmp    cl, 2
34333 0000BCC4 750A                   <1>       jne    short loc_uc_check_fat16_buff_sector_load
34334                                  <1>
34335                                  <1> loc_uc_check_fat16_buff_sector_save:
34336 0000BCC6 3B05[AA560100]         <1>       cmp    eax, [FAT_BuffSector]
34337 0000BCCC 755D                   <1>       jne    short loc_uc_save_fat_buffer
34338 0000BCCE EB15                   <1>       jmp    short loc_update_fat16_cell
34339                                  <1>
34340                                  <1> loc_uc_check_fat16_buff_sector_load:
34341 0000BCD0 80F901                 <1>       cmp    cl, 1 ; byte [FAT_BuffValidData]
34342 0000BCD3 0F85FB010000           <1>         jne     loc_uc_load_fat_sectors
34343 0000BCD9 3B05[AA560100]         <1>       cmp    eax, [FAT_BuffSector]
34344 0000BCDF 0F85EF010000           <1>         jne     loc_uc_load_fat_sectors
34345                                  <1>
34346                                  <1> loc_update_fat16_cell:
34347                                  <1> loc_update_fat16_buffer:
34348 0000BCE5 81C3001C0900           <1>       add    ebx, FAT_Buffer ; 26/02/2016
34349                                  <1>       ;movzx eax, word [ebx]
34350 0000BCEB 668B03                 <1>       mov    ax, [ebx]
34351                                  <1>       ; 01/03/2016
34352 0000BCEE 89C2                   <1>       mov    edx, eax ; old value of the cluster
34353 0000BCF0 A3[A2560100]           <1>       mov    [FAT_CurrentCluster], eax
34354 0000BCF5 8B0D[44590100]         <1>       mov    ecx, [ClusterValue] ; 32 bits
34355 0000BCFB 66890B                 <1>       mov    [ebx], cx ; 16 bits !
34356                                  <1>
34357 0000BCFE C605[A6560100]02       <1>       mov    byte [FAT_BuffValidData], 2
34358                                  <1>
34359 0000BD05 6683F802               <1>       cmp    ax, 2
34360 0000BD09 723A                   <1>       jb     short return_uc_fat_stc
34361 0000BD0B 3B05[B2560100]         <1>       cmp    eax, [LastCluster]
34362 0000BD11 7732                   <1>       ja     short return_uc_fat_stc
34363                                  <1>
34364                                  <1> loc_fat_buffer_updated:
34365                                  <1>       ; 01/03/2016
34366 0000BD13 F8                     <1>       clc
34367                                  <1> loc_fat_buffer_stc_1:
34368 0000BD14 9C                     <1>       pushf
```

```
34369 0000BD15 21C9                <1>       and    ecx, ecx
34370 0000BD17 7506                <1>       jnz    short loc_fat_buffer_updated_1
34371                              <1>
34372                              <1>       ; 01/03/2016
34373                              <1>       ; new value of the cluster = 0 (free)
34374                              <1>       ; increase free(d) cluster count
34375 0000BD19 FF05[AE560100]      <1>       inc    dword [FAT_ClusterCounter]
34376                              <1>
34377                              <1> loc_fat_buffer_updated_1: ; new value of the cluster > 0
34378 0000BD1F 09D2                <1>       or     edx, edx ; 02/03/2016
34379 0000BD21 7506                <1>       jnz    short loc_fat_buffer_updated_2
34380                              <1>       ; old value of the cluster = 0 (it was free cluster)
34381                              <1>       ; decrease free(d) cluster count
34382 0000BD23 FF0D[AE560100]      <1>       dec    dword [FAT_ClusterCounter] ; it may be negative number
34383                              <1>
34384                              <1> loc_fat_buffer_updated_2:
34385 0000BD29 9D                  <1>       popf
34386 0000BD2A C3                  <1>       retn
34387                              <1>
34388                              <1> loc_uc_save_fat_buffer:
34389                              <1>       ; byte [FAT_BuffValidData] = 2
34390 0000BD2B E8D4010000          <1>       call   save_fat_buffer
34391 0000BD30 0F8297010000        <1>         jc      loc_fat_sectors_rw_error2
34392                              <1>       ;mov   byte [FAT_BuffValidData], 1
34393 0000BD36 A1[A2560100]        <1>       mov    eax, [FAT_CurrentCluster]
34394                              <1>       ;mov   ecx, [ClusterValue]
34395                              <1>       ;jmp   short loc_update_cluster_check_fat_buffer
34396 0000BD3B 8A1E                <1>       mov    bl, [esi+LD_Name] ; 01/03/2016
34397 0000BD3D E927FFFFFF          <1>         jmp     loc_uc_reset_fat_buffer_validation
34398                              <1>
34399                              <1> update_cluster_inv_data:
34400                              <1>       ;mov   eax, 0Dh
34401 0000BD42 B00D                <1>       mov    al, 0Dh  ; Invalid Data
34402 0000BD44 C3                  <1>       retn
34403                              <1>
34404                              <1> return_uc_fat_stc:
34405                              <1>       ; 01/03/2016
34406 0000BD45 31C0                <1>       xor    eax, eax
34407 0000BD47 F9                  <1>       stc
34408 0000BD48 EBCA                <1>       jmp    short loc_fat_buffer_stc_1
34409                              <1>
34410                              <1> update_fat12_cluster:
34411                              <1> pass_uc_fat12_errc:
34412                              <1>       ;sub   edx, edx
34413 0000BD4A BB00040000          <1>       mov    ebx, 400h ;1024
34414 0000BD4F F7F3                <1>       div    ebx
34415                              <1>       ; EAX = Count of 3 FAT sectors
34416                              <1>       ; DX = Cluster offset in FAT buffer
34417 0000BD51 66B90300            <1>       mov    cx, 3
34418 0000BD55 6689C3              <1>       mov    bx, ax
34419 0000BD58 6689C8              <1>       mov    ax, cx ; 3
34420 0000BD5B 66F7E2              <1>       mul    dx     ; Multiply by 3
34421 0000BD5E 66D1E8              <1>       shr    ax, 1  ; Divide by 2
34422 0000BD61 6693                <1>       xchg   bx, ax
34423                              <1>       ; EAX = Count of 3 FAT sectors
34424                              <1>       ; EBX = Byte Offset in FAT buffer
34425 0000BD63 66F7E1              <1>       mul    cx  ; 3 * AX
34426                              <1>       ; EAX = FAT Beginning Sector
34427                              <1>       ; EDX = 0
34428 0000BD66 8A0D[A6560100]      <1>       mov    cl, [FAT_BuffValidData]
34429                              <1>       ; TRDOS v1 has a FATal bug here !
34430                              <1>       ; (it does not have 'cmp cl, 2' instruction here !
34431                              <1>       ;  while 'jne' is existing !)
34432 0000BD6C 80F902              <1>       cmp    cl, 2 ; 2 = dirty buffer (must be written to disk)
34433 0000BD6F 750A                <1>       jne    short loc_uc_check_fat12_buff_sector_load
34434                              <1>
34435                              <1> loc_uc_check_fat12_buff_sector_save:
34436 0000BD71 3B05[AA560100]      <1>       cmp    eax, [FAT_BuffSector]
34437 0000BD77 75B2                <1>         jne     short loc_uc_save_fat_buffer
34438 0000BD79 EB15                <1>       jmp    short loc_update_fat12_cell
34439                              <1>
34440                              <1> loc_uc_check_fat12_buff_sector_load:
34441 0000BD7B 80F901              <1>       cmp    cl, 1 ; byte ptr [FAT_BuffValidData]
34442 0000BD7E 0F8550010000        <1>         jne     loc_uc_load_fat_sectors
34443 0000BD84 3B05[AA560100]      <1>       cmp    eax, [FAT_BuffSector]
34444 0000BD8A 0F8544010000        <1>         jne     loc_uc_load_fat_sectors
34445                              <1>
34446                              <1> loc_update_fat12_cell:
34447 0000BD90 81C3001C0900        <1>       add    ebx, FAT_Buffer ; 26/02/2016
34448 0000BD96 668B0D[A2560100]    <1>       mov    cx, [FAT_CurrentCluster]
34449 0000BD9D 66D1E9              <1>       shr    cx, 1
34450 0000BDA0 668B03              <1>       mov    ax, [ebx]
34451 0000BDA3 6689C2              <1>       mov    dx, ax
34452 0000BDA6 7344                <1>       jnc    short uc_fat12_nc_even
34453                              <1>
34454 0000BDA8 6683E00F            <1>       and    ax, 0Fh
34455 0000BDAC 8B0D[44590100]      <1>       mov    ecx, [ClusterValue] ; 32 bits
34456 0000BDB2 66C1E104            <1>       shl    cx, 4
34457 0000BDB6 6609C1              <1>       or     cx, ax
34458 0000BDB9 6689D0              <1>       mov    ax, dx
34459 0000BDBC 66890B              <1>       mov    [ebx], cx  ; 16 bits !
34460 0000BDBF 66C1E804            <1>       shr    ax, 4 ; al(bit4..7)+ah(bit0..7)
34461                              <1>
34462                              <1> update_fat12_buffer:
34463 0000BDC3 A3[A2560100]        <1>       mov    [FAT_CurrentCluster], eax
34464 0000BDC8 89C2                <1>       mov    edx, eax ; 01/03/2016
34465 0000BDCA C605[A6560100]02    <1>       mov    byte [FAT_BuffValidData], 2
34466 0000BDD1 6683F802            <1>       cmp    ax, 2
34467 0000BDD5 0F826AFFFFFF        <1>         jb      return_uc_fat_stc
34468 0000BDDB 3B05[B2560100]      <1>       cmp    eax, [LastCluster]
34469 0000BDE1 0F875EFFFFFF        <1>         ja      return_uc_fat_stc
34470 0000BDE7 E927FFFFFF          <1>         jmp     loc_fat_buffer_updated
```

```
34471                                  <1>
34472                                  <1> uc_fat12_nc_even:
34473 0000BDEC 662500F0                <1>         and    ax, 0F000h
34474 0000BDF0 8B0D[44590100]          <1>         mov    ecx, [ClusterValue] ; 32 bits
34475 0000BDF6 80E50F                  <1>         and    ch, 0Fh
34476 0000BDF9 6609C1                  <1>         or     cx, ax
34477 0000BDFC 6689D0                  <1>         mov    ax, dx
34478 0000BDFF 66890B                  <1>         mov    [ebx], cx ; 16 bits !
34479 0000BE02 80E40F                  <1>         and    ah, 0Fh ; al(bit0..7)+ah(bit0..3)
34480 0000BE05 EBBC                    <1>         jmp    short update_fat12_buffer
34481                                  <1>
34482                                  <1> update_fat32_cluster:
34483 0000BE07 8B4E78                  <1>         mov    ecx, [esi+LD_Clusters]
34484 0000BE0A 41                      <1>         inc    ecx
34485 0000BE0B 890D[B2560100]          <1>         mov    [LastCluster], ecx
34486                                  <1>
34487 0000BE11 39C8                    <1>         cmp    eax, ecx
34488 0000BE13 0F872CFFFFFF            <1>           ja      return_uc_fat_stc
34489                                  <1>
34490                                  <1> pass_uc_fat32_errc:
34491                                  <1>         ;sub   edx, edx
34492 0000BE19 BB80010000              <1>         mov    ebx, 180h ;384
34493 0000BE1E F7F3                    <1>         div    ebx
34494                                  <1>         ; EAX = Count of 3 FAT sectors
34495                                  <1>         ; DX = Cluster offset in FAT buffer
34496 0000BE20 89D3                    <1>         mov    ebx, edx
34497 0000BE22 C1E302                  <1>         shl    ebx, 2 ; Multiply by 4
34498 0000BE25 BA03000000              <1>         mov    edx, 3
34499 0000BE2A F7E2                    <1>         mul    edx
34500                                  <1>         ; EBX = Cluster Offset in FAT buffer
34501                                  <1>         ; EAX = FAT Sector
34502                                  <1>         ; EDX = 0
34503 0000BE2C 8A0D[A6560100]          <1>         mov    cl, [FAT_BuffValidData]
34504 0000BE32 80F902                  <1>         cmp    cl, 2
34505 0000BE35 750E                    <1>         jne    short loc_uc_check_fat32_buff_sector_load
34506                                  <1>
34507                                  <1> loc_uc_check_fat32_buff_sector_save:
34508 0000BE37 3B05[AA560100]          <1>         cmp    eax, [FAT_BuffSector]
34509 0000BE3D 0F85E8FEFFFF            <1>           jne     loc_uc_save_fat_buffer
34510 0000BE43 EB11                    <1>         jmp    short loc_update_fat32_cell
34511                                  <1>
34512                                  <1> loc_uc_check_fat32_buff_sector_load:
34513 0000BE45 80F901                  <1>         cmp    cl, 1 ; byte [FAT_BuffValidData]
34514 0000BE48 0F8586000000            <1>           jne     loc_uc_load_fat_sectors
34515 0000BE4E 3B05[AA560100]          <1>         cmp    eax, [FAT_BuffSector]
34516 0000BE54 757E                    <1>           jne     loc_uc_load_fat_sectors
34517                                  <1>
34518                                  <1> loc_update_fat32_cell:
34519                                  <1> loc_update_fat32_buffer:
34520 0000BE56 81C3001C0900            <1>         add    ebx, FAT_Buffer ; 26/02/2016
34521 0000BE5C 8B03                    <1>         mov    eax, [ebx]
34522 0000BE5E 25FFFFFF0F              <1>         and    eax, 0FFFFFFFh ; 28 bit cluster value
34523                                  <1>
34524 0000BE63 8B15[A2560100]          <1>         mov    edx, [FAT_CurrentCluster] ; 01/03/2016
34525                                  <1>
34526 0000BE69 A3[A2560100]            <1>         mov    [FAT_CurrentCluster], eax
34527 0000BE6E 8B0D[44590100]          <1>         mov    ecx, [ClusterValue]
34528 0000BE74 890B                    <1>         mov    [ebx], ecx ; 29/02/2016
34529                                  <1>
34530 0000BE76 C605[A6560100]02        <1>         mov    byte [FAT_BuffValidData], 2
34531                                  <1>
34532                                  <1>         ; 01/03/2016
34533 0000BE7D 21C0                    <1>         and    eax, eax ; was it free cluster ?
34534 0000BE7F 7514                    <1>         jnz    short loc_upd_fat32_c0
34535                                  <1>
34536                                  <1>         ;or    ecx, ecx ; it will be left free ?!
34537                                  <1>         ;jz    short loc_upd_fat32_c3
34538                                  <1>
34539 0000BE81 3B563E                  <1>         cmp    edx, [esi+LD_BPB+BPB_Reserved+4] ; First free cluster
34540 0000BE84 7520                    <1>         jne    short loc_upd_fat32_c3
34541                                  <1>
34542 0000BE86 3B15[B2560100]          <1>         cmp    edx, [LastCluster]
34543 0000BE8C 7207                    <1>         jb     short loc_upd_fat32_c0
34544                                  <1>
34545 0000BE8E BA02000000              <1>         mov    edx, 2 ; rewind !
34546 0000BE93 EB0E                    <1>         jmp    short loc_upd_fat32_c2
34547                                  <1>
34548                                  <1> loc_upd_fat32_c0:
34549 0000BE95 FF463E                  <1>         inc    dword [esi+LD_BPB+BPB_Reserved+4] ; set it to next cluster
34550 0000BE98 EB0C                    <1>         jmp    short loc_upd_fat32_c3
34551                                  <1>
34552                                  <1> loc_upd_fat32_c1:
34553 0000BE9A 09C9                    <1>         or     ecx, ecx ; will it be free cluster ?
34554 0000BE9C 7508                    <1>         jnz    short loc_upd_fat32_c3
34555                                  <1>
34556 0000BE9E 3B563E                  <1>         cmp    edx, [esi+LD_BPB+BPB_Reserved+4] ; First free cluster
34557 0000BEA1 7303                    <1>         jnb    short loc_upd_fat32_c3
34558                                  <1>
34559                                  <1> loc_upd_fat32_c2:
34560 0000BEA3 89563E                  <1>         mov    [esi+LD_BPB+BPB_Reserved+4], edx
34561                                  <1>
34562                                  <1> loc_upd_fat32_c3:
34563 0000BEA6 89C2                    <1>         mov    edx, eax
34564                                  <1>
34565                                  <1> loc_upd_fat32_c4:
34566 0000BEA8 83F802                  <1>         cmp    eax, 2
34567 0000BEAB 0F8294FEFFFF            <1>           jb      return_uc_fat_stc
34568                                  <1>
34569                                  <1> pass_uc_fat32_c_zero_check_2:
34570 0000BEB1 3B05[B2560100]          <1>         cmp    eax, [LastCluster]
34571 0000BEB7 0F8788FEFFFF            <1>           ja      return_uc_fat_stc
34572                                  <1>
```

```
34573 0000BEBD E951FEFFFF          <1>       jmp     loc_fat_buffer_updated
34574                              <1>
34575                              <1> loc_fat_sectors_rw_error1:
34576                              <1>       ;mov   byte [FAT_BuffValidData], 0
34577                              <1>       ; 23/10/2016 (15h -> 17)
34578                              <1>       ; 23/03/2016
34579 0000BEC2 B811000000          <1>       mov    eax, 17 ; Drive not ready or read error
34580 0000BEC7 8825[A6560100]      <1>       mov    [FAT_BuffValidData], ah ; 0
34581                              <1>
34582                              <1> loc_fat_sectors_rw_error2:
34583                              <1>       ;mov   eax, error code
34584                              <1>       ;mov   edx, 0
34585 0000BECD 8B0D[44590100]      <1>       mov    ecx, [ClusterValue]
34586 0000BED3 C3                  <1>       retn
34587                              <1>
34588                              <1> loc_uc_load_fat_sectors:
34589 0000BED4 A3[AA560100]        <1>       mov    [FAT_BuffSector], eax
34590                              <1>
34591                              <1> load_uc_fat_sectors_zero:
34592 0000BED9 034660              <1>       add    eax, [esi+LD_FATBegin]
34593 0000BEDC BB001C0900          <1>       mov    ebx, FAT_Buffer
34594 0000BEE1 B903000000          <1>       mov    ecx, 3
34595 0000BEE6 E8B3320000          <1>       call   disk_read
34596 0000BEEB 72D5                <1>       jc     short loc_fat_sectors_rw_error1
34597                              <1>
34598 0000BEED C605[A6560100]01    <1>         mov     byte [FAT_BuffValidData], 1
34599 0000BEF4 A1[A2560100]        <1>       mov    eax, [FAT_CurrentCluster]
34600 0000BEF9 8B0D[44590100]      <1>       mov    ecx, [ClusterValue]
34601 0000BEFF E972FDFFFF          <1>         jmp     loc_update_cluster_check_fat_type
34602                              <1>
34603                              <1> save_fat_buffer:
34604                              <1>       ; 15/10/2016
34605                              <1>       ; 01/03/2016
34606                              <1>       ; 22/02/2016 (TRDOS 386 =  TRDOS v2.0)
34607                              <1>       ; 11/08/2011
34608                              <1>       ; 09/02/2005
34609                              <1>       ; INPUT ->
34610                              <1>       ;      None
34611                              <1>       ; OUTPUT ->
34612                              <1>       ;      cf = 0 -> OK.
34613                              <1>       ;      cf = 1 -> error code in AL (EAX)
34614                              <1>       ;
34615                              <1>       ;      EBX = FAT_Buffer address
34616                              <1>       ;
34617                              <1>       ; (EAX, EDX, ECX will be modified)
34618                              <1>
34619                              <1>       ;cmp   byte [FAT_BuffValidData], 2
34620                              <1>       ;je    short loc_save_fat_buff
34621                              <1>
34622                              <1> ;loc_save_fat_buffer_retn:
34623                              <1> ;     xor    eax, eax
34624                              <1> ;     retn
34625                              <1>
34626                              <1> loc_save_fat_buff:
34627 0000BF04 31D2                <1>       xor    edx, edx
34628 0000BF06 8A35[A7560100]      <1>       mov    dh, [FAT_BuffDrvName]
34629 0000BF0C 80FE41              <1>       cmp    dh, 'A'
34630 0000BF0F 722E                <1>       jb     short loc_save_fat_buffer_inv_data_retn
34631 0000BF11 80EE41              <1>       sub    dh, 'A'
34632 0000BF14 56                  <1>       push   esi ; *
34633 0000BF15 BE00010900          <1>         mov     esi, Logical_DOSDisks
34634 0000BF1A 01D6                <1>       add    esi, edx
34635                              <1>
34636 0000BF1C 8A5603              <1>       mov    dl, [esi+LD_FATType]
34637 0000BF1F 20D2                <1>       and    dl, dl
34638 0000BF21 741B                <1>       jz     short loc_save_fat_buffer_inv_data_pop_retn
34639                              <1>
34640 0000BF23 A1[AA560100]        <1>       mov    eax, [FAT_BuffSector]
34641 0000BF28 80FA02              <1>       cmp    dl, 2
34642 0000BF2B 770A                <1>       ja     short loc_save_fat32_buff
34643                              <1>
34644                              <1> loc_save_fat_12_16_buff:
34645                              <1>       ; 01/03/2016
34646                              <1>       ; TRDOS v1 has a FATal bug here!
34647                              <1>       ; Correct code: mov dx, word ptr [FAT_BuffSector]+2
34648                              <1>       ; (DX:AX in TRDOS v1 -> EAX in TRDOS v2)
34649                              <1>       ;
34650 0000BF2D 0FB74E1C            <1>       movzx  ecx, word [esi+LD_BPB+FATSecs]
34651 0000BF31 29C1                <1>       sub    ecx, eax
34652                              <1>       ; TRDOS v1 has a bug here... ('pop esi' was forgotten!)
34653                              <1>       ;jna   short loc_save_fat_buffer_inv_data_retn ; wrong addr!
34654 0000BF33 7609                <1>       jna    short loc_save_fat_buffer_inv_data_pop_retn ; correct addr.
34655 0000BF35 EB15                <1>       jmp    short loc_save_fat_buffer_check_rs3
34656                              <1>
34657                              <1> loc_save_fat32_buff:
34658 0000BF37 8B4E2A              <1>       mov    ecx, [esi+LD_BPB+FAT32_FAT_Size]
34659 0000BF3A 29C1                <1>       sub    ecx, eax
34660 0000BF3C 770E                <1>       ja     short loc_save_fat_buffer_check_rs3
34661                              <1>
34662                              <1> loc_save_fat_buffer_inv_data_pop_retn:
34663 0000BF3E 5E                  <1>       pop    esi ; *
34664                              <1> loc_save_fat_buffer_inv_data_retn:
34665 0000BF3F B80D000000          <1>       mov    eax, 0Dh ; Invalid DATA
34666 0000BF44 C3                  <1>       retn
34667                              <1>
34668                              <1> loc_save_fat_buff_remain_sectors_3:
34669 0000BF45 B903000000          <1>       mov    ecx, 3
34670 0000BF4A EB05                <1>       jmp    short loc_save_fat_buff_continue
34671                              <1>
34672                              <1> loc_save_fat_buffer_check_rs3:
34673 0000BF4C 83F903              <1>       cmp    ecx, 3
34674 0000BF4F 77F4                <1>       ja     short loc_save_fat_buff_remain_sectors_3
```

```
34675                                  <1>
34676                                  <1> loc_save_fat_buff_continue:
34677 0000BF51 BB001C0900              <1>       mov    ebx, FAT_Buffer
34678 0000BF56 034660                  <1>       add    eax, [esi+LD_FATBegin]
34679 0000BF59 51                      <1>       push   ecx
34680 0000BF5A E830320000              <1>       call   disk_write
34681 0000BF5F 59                      <1>       pop    ecx
34682 0000BF60 722B                    <1>       jc     short loc_save_FAT_buff_write_err
34683                                  <1>
34684 0000BF62 807E0302                <1>       cmp    byte [esi+LD_FATType], 2
34685 0000BF66 7605                    <1>       jna    short loc_calc_2nd_fat12_16_addr
34686                                  <1>
34687                                  <1> loc_calc_2nd_fat32_addr:
34688 0000BF68 8B462A                  <1>       mov    eax, [esi+LD_BPB+FAT32_FAT_Size]
34689 0000BF6B EB04                    <1>       jmp    short loc_calc_2nd_fat_addr
34690                                  <1>
34691                                  <1> loc_calc_2nd_fat12_16_addr:
34692 0000BF6D 0FB7461C                <1>       movzx  eax, word [esi+LD_BPB+FATSecs]
34693                                  <1>
34694                                  <1> loc_calc_2nd_fat_addr:
34695 0000BF71 034660                  <1>       add    eax, [esi+LD_FATBegin]
34696 0000BF74 0305[AA560100]          <1>       add    eax, [FAT_BuffSector]
34697 0000BF7A BB001C0900              <1>       mov    ebx, FAT_Buffer
34698                                  <1>       ; ecx = 1 to 3
34699 0000BF7F E80B320000              <1>       call   disk_write
34700 0000BF84 7207                    <1>       jc     short loc_save_FAT_buff_write_err
34701                                  <1>       ; Valid  buffer (1 = valid but do not save)
34702 0000BF86 C605[A6560100]01        <1>       mov    byte [FAT_BuffValidData], 1
34703                                  <1>
34704                                  <1> loc_save_FAT_buff_write_err:
34705 0000BF8D 5E                      <1>       pop    esi ; *
34706 0000BF8E BB001C0900              <1>       mov    ebx, FAT_Buffer
34707                                  <1>       ; 15/10/2016 (1Dh -> 18)
34708                                  <1>       ; 23/03/2016 (1Dh)
34709 0000BF93 B812000000              <1>       mov    eax, 18 ; Drive not ready or write error
34710 0000BF98 C3                      <1>       retn
34711                                  <1>
34712                                  <1> calculate_fat_freespace:
34713                                  <1>       ; 23/03/2016
34714                                  <1>       ; 02/03/2016
34715                                  <1>       ; 01/03/2016
34716                                  <1>       ; 29/02/2016
34717                                  <1>       ; 22/02/2016 (TRDOS 386 =  TRDOS v2.0)
34718                                  <1>       ; 30/04/2011
34719                                  <1>       ; 03/04/2010
34720                                  <1>       ; 2005
34721                                  <1>       ; INPUT ->
34722                                  <1>       ;      EAX = Cluster count to be added or subtracted
34723                                  <1>       ;      If BH = FFh, ESI = TR-DOS Logical Drive Description Table
34724                                  <1>       ;      If BH < FFh, BH = TR-DOS Logical Drive Number
34725                                  <1>       ;      BL:
34726                                  <1>       ;      0 = Calculate, 1 = Add, 2 = Subtract, 3 = Get (Not Set/Calc)
34727                                  <1>       ; OUTPUT ->
34728                                  <1>       ;      EAX = Free Space in sectors
34729                                  <1>       ;      ESI = Logical Dos Drive Description Table address
34730                                  <1>       ;      BH = Logical Dos Drive Number (same with input value of BH)
34731                                  <1>       ;      BL = Type of operation (same with input value of BL)
34732                                  <1>       ;      ECX = 0 -> valid
34733                                  <1>       ;      ECX > 0 -> error or invalid
34734                                  <1>       ;      If EAX = FFFFFFFFh, it is 're-calculation needed'
34735                                  <1>       ;                              sign due to r/w error
34736                                  <1>
34737 0000BF99 66891D[4A590100]        <1>       mov    [CFS_OPType], bx
34738 0000BFA0 A3[4C590100]            <1>       mov    [CFS_CC], eax
34739                                  <1>
34740 0000BFA5 80FFFF                  <1>       cmp    bh, 0FFh
34741 0000BFA8 740B                    <1>       je     short pass_calculate_freespace_get_drive_dt_offset
34742                                  <1>
34743                                  <1> loc_calculate_freespace_get_drive_dt_offset:
34744 0000BFAA 31C0                    <1>       xor    eax, eax
34745 0000BFAC 88FC                    <1>         mov     ah, bh
34746 0000BFAE BE00010900              <1>       mov    esi, Logical_DOSDisks
34747 0000BFB3 01C6                    <1>         add     esi, eax
34748                                  <1>
34749                                  <1> pass_calculate_freespace_get_drive_dt_offset:
34750 0000BFB5 08DB                    <1>       or     bl, bl
34751 0000BFB7 7435                    <1>       jz     short loc_reset_fcc
34752                                  <1>
34753                                  <1> loc_get_free_sectors:
34754 0000BFB9 8B4674                  <1>       mov    eax, [esi+LD_FreeSectors]
34755                                  <1>
34756                                  <1>       ;xor   ecx, ecx
34757                                  <1>       ;dec   ecx ; 0FFFFFFFFh
34758                                  <1>       ;cmp   eax, ecx ; 29/02/2016
34759                                  <1>       ;je    short loc_get_free_sectors_retn ; recalculation is needed!
34760                                  <1>
34761                                  <1>       ; 23/03/2016
34762 0000BFBC 8B4E70                  <1>       mov    ecx, [esi+LD_TotalSectors]
34763 0000BFBF 39C1                    <1>       cmp    ecx, eax ; Total sectors must be greater than Free sectors !
34764 0000BFC1 7707                    <1>       ja     short loc_get_free_sectors_check_optype
34765                                  <1>
34766 0000BFC3 31C0                    <1>       xor    eax, eax
34767 0000BFC5 48                      <1>       dec    eax ; 0FFFFFFFFh  ; recalculation is needed!
34768 0000BFC6 894674                  <1>       mov    [esi+LD_FreeSectors], eax ; reset (for recalculation)
34769                                  <1>
34770                                  <1> loc_get_free_sectors_retn:
34771 0000BFC9 C3                      <1>       retn
34772                                  <1>
34773                                  <1> loc_get_free_sectors_check_optype:
34774 0000BFCA 80FB03                  <1>       cmp    bl, 3
34775 0000BFCD 7203                    <1>       jb     short loc_set_fcc
34776                                  <1>
```

```
34777 0000BFCF 29C9               <1>       sub    ecx, ecx ; 0
34778                             <1>
34779 0000BFD1 C3                 <1>       retn
34780                             <1>
34781                             <1> loc_set_fcc:
34782 0000BFD2 807E0302           <1>       cmp    byte [esi+LD_FATType], 2
34783 0000BFD6 0F87DF000000       <1>         ja      loc_update_FAT32_fs_info_fcc
34784                             <1>
34785                             <1>       ;mov   eax, [esi+LD_FreeSectors]
34786 0000BFDC 0FB64E13           <1>       movzx  ecx, byte [esi+LD_BPB+SecPerClust]
34787 0000BFE0 29D2               <1>       sub    edx, edx
34788 0000BFE2 F7F1               <1>       div    ecx
34789                             <1>       ;or    dx, dx
34790                             <1>       ;      ; DX -> Remain sectors < SecPerClust
34791                             <1>       ;      ; DX > 0 -> invalid free sector count
34792                             <1>       ;jnz   short loc_reset_fcc
34793                             <1>
34794                             <1> ;pass_set_fcc_div32:
34795 0000BFE4 A3[C3560100]       <1>       mov    [FreeClusterCount], eax
34796 0000BFE9 E988000000         <1>         jmp     loc_set_free_sectors_FAT12_FAT16
34797                             <1>
34798                             <1> loc_reset_fcc:
34799 0000BFEE 31C0               <1>       xor    eax, eax
34800 0000BFF0 A3[C3560100]       <1>       mov    [FreeClusterCount], eax ; 0
34801 0000BFF5 8B5678             <1>       mov    edx, [esi+LD_Clusters]
34802 0000BFF8 42                 <1>       inc    edx
34803 0000BFF9 8915[B2560100]     <1>       mov    [LastCluster], edx
34804                             <1>
34805 0000BFFF 807E0302           <1>       cmp    byte [esi+LD_FATType], 2
34806 0000C003 7647               <1>       jna    short loc_count_free_fat_clusters_0
34807                             <1>
34808 0000C005 48                 <1>       dec    eax ; FFFFFFFFh
34809 0000C006 A3[54590100]       <1>       mov    [CFS_FAT32FC], eax
34810                             <1>
34811                             <1>       ; 29/02/2016
34812 0000C00B 89463A             <1>       mov    [esi+LD_BPB+BPB_Reserved], eax ; reset
34813 0000C00E 89463E             <1>       mov    [esi+LD_BPB+BPB_Reserved+4], eax ; reset
34814                             <1>
34815 0000C011 B802000000         <1>       mov    eax, 2
34816                             <1>
34817                             <1> loc_count_fc_next_cluster_0:
34818 0000C016 50                 <1>       push   eax
34819 0000C017 E801F9FFFF         <1>       call   get_next_cluster
34820 0000C01C 7310               <1>       jnc    short loc_check_fat32_ff_cluster
34821 0000C01E 09C0               <1>       or     eax, eax
34822 0000C020 741E               <1>       jz     short pass_inc_cfs_fcc_0
34823                             <1>
34824                             <1> loc_put_fcc_unknown_sign:
34825 0000C022 58                 <1>       pop    eax
34826                             <1>       ; "Free count is Unknown" sign
34827                             <1>       ;mov   dword [FreeClusterCount], 0FFFFFFFFh
34828                             <1>
34829                             <1>       ; 29/02/2016
34830                             <1>       ; Save Free Cluster Count value in FAT32 'BPB_Reserved' area
34831                             <1>       ;mov   [esi+LD_BPB+BPB_Reserved], 0FFFFFFFFh ; unknown!
34832 0000C023 8B15[54590100]     <1>       mov    edx, [CFS_FAT32FC] ; First Free Cluster
34833                             <1>       ; Save First Free Cluster value in FAT32 'BPB_Reserved+4' area
34834 0000C029 89563E             <1>       mov    [esi+LD_BPB+BPB_Reserved+4], edx
34835                             <1>
34836 0000C02C EB7D               <1>         jmp     loc_put_fcc_invalid_sign
34837                             <1>
34838                             <1> loc_check_fat32_ff_cluster:
34839 0000C02E 09C0               <1>       or     eax, eax
34840 0000C030 750E               <1>       jnz    short pass_inc_cfs_fcc_0
34841 0000C032 58                 <1>       pop    eax
34842 0000C033 A3[54590100]       <1>       mov    [CFS_FAT32FC], eax
34843                             <1>       ;mov   dword [FreeClusterCount], 1
34844 0000C038 FF05[C3560100]     <1>       inc    dword [FreeClusterCount]
34845 0000C03E EB27               <1>       jmp    short pass_inc_cfs_fcc_1
34846                             <1>
34847                             <1> pass_inc_cfs_fcc_0:
34848 0000C040 58                 <1>       pop    eax
34849                             <1>
34850                             <1> pass_inc_cfs_fcc_0c:
34851 0000C041 40                 <1>       inc    eax ; add eax, 1
34852 0000C042 3B05[B2560100]     <1>       cmp    eax, [LastCluster]
34853 0000C048 76CC               <1>       jna    short loc_count_fc_next_cluster_0
34854 0000C04A EB6F               <1>       jmp    short loc_update_FAT32_fs_info_fcc
34855                             <1>
34856                             <1> loc_count_free_fat_clusters_0:
34857                             <1>       ;mov   eax, 2
34858 0000C04C B002               <1>       mov    al, 2
34859                             <1>
34860                             <1> loc_count_fc_next_cluster:
34861 0000C04E 50                 <1>       push   eax
34862 0000C04F E8C9F8FFFF         <1>       call   get_next_cluster
34863 0000C054 720C               <1>       jc     short loc_count_fcc_stc
34864                             <1>
34865                             <1> loc_count_free_clusters_1:
34866 0000C056 21C0               <1>       and    eax, eax
34867 0000C058 750C               <1>       jnz    short pass_inc_cfs_fcc
34868                             <1>
34869 0000C05A FF05[C3560100]     <1>       inc    dword [FreeClusterCount]
34870 0000C060 EB04               <1>       jmp    short pass_inc_cfs_fcc
34871                             <1>
34872                             <1> loc_count_fcc_stc:
34873 0000C062 09C0               <1>       or     eax, eax
34874 0000C064 75BC               <1>       jnz    short loc_put_fcc_unknown_sign ; 29/02/2016
34875                             <1>
34876                             <1> pass_inc_cfs_fcc:
34877 0000C066 58                 <1>       pop    eax
34878                             <1>
```

```
34879                                  <1> pass_inc_cfs_fcc_1:
34880 0000C067 40                      <1>       inc    eax ; add eax, 1
34881 0000C068 3B05[B2560100]          <1>       cmp    eax, [LastCluster]
34882 0000C06E 76DE                    <1>       jna    short loc_count_fc_next_cluster
34883                                  <1>
34884                                  <1> loc_set_free_sectors:
34885 0000C070 807E0302                <1>       cmp    byte [esi+LD_FATType], 2
34886 0000C074 7745                    <1>       ja     short loc_update_FAT32_fs_info_fcc
34887                                  <1>
34888                                  <1> loc_set_free_sectors_FAT12_FAT16:
34889 0000C076 803D[4A590100]00        <1>       cmp    byte [CFS_OPType], 0
34890 0000C07D 761C                    <1>       jna    short pass_FAT_add_sub_fcc
34891 0000C07F A1[4C590100]            <1>       mov    eax, [CFS_CC]
34892 0000C084 803D[4A590100]01        <1>       cmp    byte [CFS_OPType], 1
34893 0000C08B 7708                    <1>       ja     short pass_FAT_add_fcc
34894 0000C08D 0105[C3560100]          <1>       add    [FreeClusterCount], eax
34895 0000C093 EB06                    <1>       jmp    short pass_FAT_add_sub_fcc
34896                                  <1>
34897                                  <1> pass_FAT_add_fcc:
34898 0000C095 2905[C3560100]          <1>       sub    [FreeClusterCount], eax
34899                                  <1>
34900                                  <1> pass_FAT_add_sub_fcc:
34901 0000C09B 0FB64613                <1>       movzx  eax, byte [esi+LD_BPB+SecPerClust]
34902 0000C09F 8B15[C3560100]          <1>       mov    edx, [FreeClusterCount]
34903 0000C0A5 F7E2                    <1>       mul    edx
34904                                  <1>
34905 0000C0A7 31C9                    <1>       xor    ecx, ecx
34906 0000C0A9 EB05                    <1>       jmp    short loc_cfs_retn_params
34907                                  <1>
34908                                  <1> loc_put_fcc_invalid_sign:
34909 0000C0AB 29C0                    <1>             sub    eax, eax ; 0
34910 0000C0AD 48                      <1>       dec    eax ; FFFFFFFFh
34911                                  <1> loc_fat32_ffc_recalc_needed:
34912 0000C0AE 89C1                    <1>       mov    ecx, eax
34913                                  <1>
34914                                  <1> loc_cfs_retn_params:
34915 0000C0B0 894674                  <1>       mov    [esi+LD_FreeSectors], eax
34916 0000C0B3 0FB71D[4A590100]        <1>       movzx  ebx, word [CFS_OPType]
34917 0000C0BA C3                      <1>       retn
34918                                  <1>
34919                                  <1> loc_update_FAT32_fs_info_fcc:
34920                                  <1> loc_check_fcc_FSINFO_op:
34921                                  <1>       ; 29/02/2016
34922                                  <1>       ; EAX = Free cluster count (before this update) ; value from disk
34923                                  <1>       ; EDX = First Free Cluster (before this update) ; value from disk
34924 0000C0BB 803D[4A590100]01        <1>       cmp    byte [CFS_OPType], 1
34925 0000C0C2 7221                    <1>       jb     short loc_cfs_FAT32_get_rcalc_parms ; 0 = recalculated
34926 0000C0C4 7406                    <1>       je     short loc_check_fcc_FSINFO_op1 ; 1 = add
34927                                  <1> loc_check_fcc_FSINFO_op2: ; subtract
34928 0000C0C6 F71D[4C590100]          <1>       neg    dword [CFS_CC] ; prepare to subtract ; 2 = sub (add negative)
34929                                  <1> loc_check_fcc_FSINFO_op1:
34930                                  <1>       ; 01/03/2016
34931 0000C0CC 31D2                    <1>       xor    edx, edx ; 0
34932 0000C0CE 4A                      <1>       dec    edx ; 0FFFFFFFFh
34933 0000C0CF 8B463A                  <1>       mov    eax, [esi+LD_BPB+BPB_Reserved]
34934 0000C0D2 39D0                    <1>       cmp    eax, edx
34935 0000C0D4 73D5                    <1>       jnb    short loc_put_fcc_invalid_sign
34936 0000C0D6 0305[4C590100]          <1>         add     eax, [CFS_CC] ; free cluster count on disk + current count
34937 0000C0DC 72CD                    <1>       jc     short loc_put_fcc_invalid_sign
34938                                  <1>
34939 0000C0DE A3[C3560100]            <1>       mov    [FreeClusterCount], eax
34940 0000C0E3 EB0E                    <1>       jmp    short loc_cfs_write_FSINFO_sector
34941                                  <1>
34942                                  <1> loc_cfs_FAT32_get_rcalc_parms:
34943 0000C0E5 8B15[54590100]          <1>       mov    edx, [CFS_FAT32FC]
34944 0000C0EB A1[C3560100]            <1>       mov    eax, [FreeClusterCount]
34945 0000C0F0 89563E                  <1>       mov    [esi+LD_BPB+BPB_Reserved+4], edx ; First Free Cluster
34946                                  <1> loc_cfs_write_FSINFO_sector:
34947 0000C0F3 89463A                  <1>       mov    [esi+LD_BPB+BPB_Reserved], eax ; Free cluster count
34948                                  <1>       ; 01/03/2016
34949 0000C0F6 E8AA000000              <1>       call   set_fat32_fsinfo_sector_parms
34950 0000C0FB 72AE                    <1>         jc     short loc_put_fcc_invalid_sign
34951                                  <1>
34952                                  <1> loc_set_FAT32_free_sectors:
34953                                  <1>       ; 29/02/2016
34954                                  <1>       ;mov   eax, [FreeClusterCount]
34955                                  <1>       ;mov   ecx, eax
34956                                  <1>       ;cmp   eax, 0FFFFFFFFh ; Invalid !
34957                                  <1>       ;je    short loc_cfs_retn_params
34958                                  <1>       ;
34959 0000C0FD 8B0D[C3560100]          <1>       mov    ecx, [FreeClusterCount]
34960 0000C103 0FB64613                <1>       movzx  eax, byte [esi+LD_BPB+SecPerClust]
34961 0000C107 F7E1                    <1>       mul    ecx
34962                                  <1>       ; 29/02/2016
34963 0000C109 31C9                    <1>       xor    ecx, ecx ; 0
34964 0000C10B 09D2                    <1>       or     edx, edx ; 0 ?
34965 0000C10D 759C                    <1>         jnz     loc_put_fcc_invalid_sign
34966 0000C10F 394670                  <1>       cmp    [esi+LD_TotalSectors], eax ; Volume size in sectors
34967 0000C112 7697                    <1>         jna     short loc_put_fcc_invalid_sign
34968                                  <1>       ;
34969                                  <1> loc_set_FAT32_free_sectors_ok:
34970 0000C114 31D2                    <1>       xor    edx, edx ; 0
34971 0000C116 EB98                    <1>         jmp     short loc_cfs_retn_params
34972                                  <1>       ;
34973                                  <1>
34974                                  <1> get_last_cluster:
34975                                  <1>       ; 22/10/2016
34976                                  <1>       ; 27/02/2016 (TRDOS 386 =  TRDOS v2.0)
34977                                  <1>       ; 12/06/2010 (DRV_FAT.ASM, 'proc_get_last_custer')
34978                                  <1>       ; 06/06/2010
34979                                  <1>       ; INPUT ->
34980                                  <1>       ;      EAX = First Cluster Number
```

```
34981                                  <1>       ;      ESI = Logical Dos Drive Parameters Table
34982                                  <1>       ; OUTPUT ->
34983                                  <1>       ;      cf = 0 -> No Error, EAX is valid
34984                                  <1>       ;      cf = 1 -> EAX > 0 -> Error
34985                                  <1>       ;      EAX = Last Cluster Number
34986                                  <1>       ;       ECX = Previous Cluster -just before the last cluster-
34987                                  <1>       ;       ; 22/10/2016
34988                                  <1>       ;      [glc_index] = cluster index number of the last cluster
34989                                  <1>       ;
34990                                  <1>       ; (Modified registers: EAX, ECX, EBX, EDX)
34991                                  <1>
34992 0000C118 89C1                   <1>       mov    ecx, eax
34993                                  <1>
34994 0000C11A C705[5C590100]FFFF-     <1>       mov    dword [glc_index], 0FFFFFFFFh ; 22/10/2016
34995 0000C122 FFFF                   <1>
34996                                  <1>
34997                                  <1> loc_glc_get_next_cluster_1:
34998 0000C124 890D[58590100]         <1>       mov    [glc_prevcluster], ecx
34999                                  <1>       ; 22/10/2016
35000 0000C12A FF05[5C590100]         <1>       inc    dword [glc_index]
35001                                  <1>
35002                                  <1> loc_glc_get_next_cluster_2:
35003 0000C130 E8E8F7FFFF             <1>       call   get_next_cluster
35004                                  <1>       ; ecx = current/previous cluster
35005                                  <1>       ; eax = next/last cluster
35006 0000C135 73ED                   <1>       jnc    short loc_glc_get_next_cluster_1
35007                                  <1>
35008 0000C137 09C0                   <1>       or     eax, eax
35009 0000C139 7509                   <1>       jnz    short loc_glc_stc_retn
35010                                  <1>
35011                                  <1>       ; ecx = previous cluster
35012 0000C13B 89C8                   <1>         mov  eax, ecx
35013                                  <1>
35014                                  <1>       ; previous cluster becomes last cluster (ecx -> eax)
35015                                  <1>       ; previous of previous cluster becomes previous cluster (ecx)
35016                                  <1>
35017                                  <1> loc_glc_prev_cluster_retn:
35018 0000C13D 8B0D[58590100]         <1>       mov    ecx, [glc_prevcluster]
35019 0000C143 C3                     <1>       retn
35020                                  <1>
35021                                  <1> loc_glc_stc_retn:
35022 0000C144 F5                     <1>       cmc    ;stc
35023 0000C145 EBF6                   <1>         jmp  short loc_glc_prev_cluster_retn
35024                                  <1>
35025                                  <1> truncate_cluster_chain:
35026                                  <1>       ; 01/03/2016
35027                                  <1>       ; 28/02/2016 (TRDOS 386 =  TRDOS v2.0)
35028                                  <1>       ; 22/01/2011 (DRV_FAT.ASM, 'proc_truncate_cluster_chain')
35029                                  <1>       ; 11/09/2010
35030                                  <1>       ; INPUT ->
35031                                  <1>       ;      ESI = Logical dos drive description table address
35032                                  <1>       ;      EAX = First cluster to be truncated/unlinked
35033                                  <1>       ; OUTPUT ->
35034                                  <1>       ;      ESI = Logical dos drive description table address
35035                                  <1>       ;      ECX = Count of truncated/removed clusters
35036                                  <1>       ;      CF = 0 -> EAX = Free sectors
35037                                  <1>       ;      CF = 1 -> Error code in EAX (AL)
35038                                  <1>
35039                                  <1>       ; NOTE: This procedure does not update lm date&time !
35040                                  <1>
35041                                  <1> loc_truncate_cc:
35042 0000C147 31C9                   <1>       xor    ecx, ecx ; mov ecx, 0
35043                                  <1>       ;mov   byte [FAT_BuffValidData], 0
35044 0000C149 890D[AE560100]         <1>       mov    [FAT_ClusterCounter], ecx ; 0 ; reset
35045                                  <1>
35046                                  <1> loc_tcc_unlink_clusters:
35047 0000C14F E8F3FAFFFF             <1>       call   update_cluster
35048                                  <1>       ; EAX = Next Cluster
35049                                  <1>       ; ECX = Cluster Value
35050                                  <1>       ; Note:
35051                                  <1>       ; Returns count of unlinked clusters in
35052                                  <1>       ; dword ptr FAT_ClusterCounter
35053 0000C154 73F9                   <1>       jnc short loc_tcc_unlink_clusters
35054                                  <1>
35055                                  <1> pass_tcc_unlink_clusters:
35056 0000C156 A2[63590100]           <1>       mov    byte [TCC_FATErr], al
35057 0000C15B 803D[A6560100]02       <1>       cmp    byte [FAT_BuffValidData], 2
35058 0000C162 750E                   <1>       jne    short loc_tcc_calculate_FAT_freespace
35059 0000C164 E89BFDFFFF             <1>       call   save_fat_buffer
35060 0000C169 7307                   <1>       jnc    short loc_tcc_calculate_FAT_freespace
35061 0000C16B A2[63590100]           <1>       mov    byte [TCC_FATErr], al ; Error
35062                                  <1>       ;mov   byte [FAT_BuffValidData], 0
35063                                  <1>
35064                                  <1>       ; 01/03/2016
35065 0000C170 EB12                   <1>       jmp    short loc_tcc_recalculate_FAT_freespace
35066                                  <1>
35067                                  <1> loc_tcc_calculate_FAT_freespace:
35068 0000C172 A1[AE560100]           <1>       mov    eax, [FAT_ClusterCounter] ; signed (+-) number
35069 0000C177 66BB01FF               <1>       mov    bx, 0FF01h ; BH = FFh -> ESI = Dos drv desc. table
35070                                  <1>                         ; BL = 1 -> add cluster
35071 0000C17B E819FEFFFF             <1>       call   calculate_fat_freespace
35072 0000C180 21C9                   <1>       and    ecx, ecx ; cx = 0 -> valid free sector count
35073 0000C182 7409                   <1>       jz     short pass_truncate_cc_recalc_FAT_freespace
35074                                  <1>
35075                                  <1> loc_tcc_recalculate_FAT_freespace:
35076 0000C184 66BB00FF               <1>       mov    bx, 0FF00h ; recalculate !
35077 0000C188 E80CFEFFFF             <1>       call   calculate_fat_freespace
35078                                  <1>
35079                                  <1> loc_tcc_calculate_FAT_freespace_err:
35080                                  <1> pass_truncate_cc_recalc_FAT_freespace:
35081 0000C18D 8B0D[AE560100]         <1>       mov    ecx, [FAT_ClusterCounter]
35082                                  <1>
```

```
35083 0000C193 803D[63590100]00    <1>       cmp    byte [TCC_FATErr], 0
35084 0000C19A 7608                <1>       jna    short loc_tcc_unlink_clusters_retn
35085                              <1>
35086                              <1> loc_tcc_unlink_clusters_error:
35087 0000C19C 0FB605[63590100]    <1>       movzx  eax, byte [TCC_FATErr]
35088 0000C1A3 F9                  <1>       stc
35089                              <1> loc_tcc_unlink_clusters_retn:
35090 0000C1A4 C3                  <1>       retn
35091                              <1>
35092                              <1> set_fat32_fsinfo_sector_parms:
35093                              <1>       ; 15/10/2016
35094                              <1>       ; 23/03/2016
35095                              <1>       ; 29/02/2016 (TRDOS 386 =  TRDOS v2.0)
35096                              <1>       ; INPUT ->
35097                              <1>       ;      ESI = Logical dos drive description table address
35098                              <1>       ;      [esi+LD_BPB+BPB_Reserved] = Free Cluster Count
35099                              <1>       ;      [esi+LD_BPB+BPB_Reserved+4] = First Free Cluster
35100                              <1>       ; OUTPUT ->
35101                              <1>       ;      ESI = Logical dos drive description table address
35102                              <1>       ;      CF = 0 -> OK..
35103                              <1>       ;      CF = 1 -> Error code in EAX (AL)
35104                              <1>       ;
35105                              <1>       ; (Modified registers: EAX, EBX, ECX, EDX)
35106                              <1>
35107 0000C1A5 E824000000          <1>       call   get_fat32_fsinfo_sector_parms
35108 0000C1AA 7221                <1>       jc     short update_fat32_fsinfo_sector_retn
35109                              <1>
35110 0000C1AC 8B463A              <1>       mov    eax, [esi+LD_BPB+BPB_Reserved] ; Free Cluster Count
35111 0000C1AF 8B563E              <1>       mov    edx, [esi+LD_BPB+BPB_Reserved+4] ; First free Cluster
35112                              <1>
35113                              <1>         ;mov ebx, DOSBootSectorBuff
35114 0000C1B2 8983E8010000        <1>       mov    [ebx+488], eax
35115 0000C1B8 8993EC010000        <1>       mov    [ebx+492], edx
35116                              <1>
35117 0000C1BE A1[50590100]        <1>       mov    eax, [CFS_FAT32FSINFOSEC]
35118 0000C1C3 B901000000          <1>       mov    ecx, 1
35119 0000C1C8 E8C22F0000          <1>       call   disk_write
35120                              <1>       ;jnc     short update_fat32_fsinfo_sector_retn
35121                              <1>
35122                              <1>       ; 15/10/2016 (1Dh -> 18)
35123                              <1>       ; 23/03/2016 (1Dh)
35124                              <1>       ;mov   eax, 18 ; Drive not ready or write error
35125                              <1>
35126                              <1> update_fat32_fsinfo_sector_retn:
35127 0000C1CD C3                  <1>       retn
35128                              <1>
35129                              <1> get_fat32_fsinfo_sector_parms:
35130                              <1>       ; 15/10/2016
35131                              <1>       ; 23/03/2016
35132                              <1>       ; 01/03/2016
35133                              <1>       ; 29/02/2016 (TRDOS 386 =  TRDOS v2.0)
35134                              <1>       ; INPUT ->
35135                              <1>       ;      ESI = Logical dos drive description table address
35136                              <1>       ; OUTPUT ->
35137                              <1>       ;      ESI = Logical dos drive description table address
35138                              <1>       ;      EBX = FSINFO sector buffer address (DOSBootSectorBuff)
35139                              <1>       ;      CF = 0 -> OK..
35140                              <1>       ;         EAX = FsInfo sector address
35141                              <1>       ;         ECX = Free cluster count
35142                              <1>       ;         EDX = First free cluster
35143                              <1>       ;      CF = 1 -> Error code in AL (EAX)
35144                              <1>       ;         EBX = 0
35145                              <1>       ;
35146                              <1>       ;      [CFS_FAT32FSINFOSEC] = FAT32 FSINFO sector address
35147                              <1>         ;
35148                              <1>       ; (Modified registers: EAX, EBX, ECX, EDX)
35149                              <1>
35150 0000C1CE 0FB74636            <1>       movzx  eax, word [esi+LD_BPB+FAT32_FSInfoSec]
35151 0000C1D2 03466C              <1>       add    eax, [esi+LD_StartSector]
35152 0000C1D5 A3[50590100]        <1>       mov    [CFS_FAT32FSINFOSEC], eax
35153                              <1>
35154 0000C1DA BB[A2540100]        <1>         mov     ebx, DOSBootSectorBuff
35155 0000C1DF B901000000          <1>       mov    ecx, 1
35156 0000C1E4 E8B52F0000          <1>       call   disk_read
35157 0000C1E9 7232                <1>       jc     short loc_read_FAT32_fsinfo_sec_err
35158                              <1>
35159 0000C1EB BB[A2540100]        <1>       mov    ebx, DOSBootSectorBuff
35160                              <1>
35161 0000C1F0 813B52526141        <1>       cmp    dword [ebx], 41615252h
35162 0000C1F6 751E                <1>       jne    short loc_read_FAT32_fsinfo_sec_stc
35163                              <1>
35164 0000C1F8 81BBE4010000727241- <1>       cmp    dword [ebx+484], 61417272h
35165 0000C201 61                  <1>
35166 0000C202 7512                <1>       jne    short loc_read_FAT32_fsinfo_sec_stc
35167                              <1>
35168 0000C204 A1[50590100]        <1>       mov    eax, [CFS_FAT32FSINFOSEC]
35169 0000C209 8B8BE8010000        <1>       mov    ecx, [ebx+488] ; free cluster count
35170 0000C20F 8B93EC010000        <1>       mov    edx, [ebx+492] ; first (next) free cluster
35171                              <1>
35172 0000C215 C3                  <1>       retn
35173                              <1>
35174                              <1> loc_read_FAT32_fsinfo_sec_stc:
35175                              <1>       ; 15/10/2016 (0Bh -> 28)
35176 0000C216 B81C000000          <1>       mov    eax, 28 ; Invalid format!
35177 0000C21B EB05                <1>       jmp    short loc_read_FAT32_fsinfo_sec_stc_retn
35178                              <1>
35179                              <1> loc_read_FAT32_fsinfo_sec_err:
35180                              <1>       ; 15/10/2016 (15h -> 17)
35181                              <1>       ; 23/03/2016 (15h)
35182 0000C21D B811000000          <1>       mov    eax, 17 ; Drive not ready or read error
35183                              <1>
35184                              <1> loc_read_FAT32_fsinfo_sec_stc_retn:
```

```
35185 0000C222 29DB              <1>       sub    ebx, ebx ; 0
35186 0000C224 F9                <1>       stc
35187 0000C225 C3                <1>       retn
35188                            <1>
35189                            <1> add_new_cluster:
35190                            <1>       ; 15/10/2016
35191                            <1>       ; 16/05/2016
35192                            <1>       ; 18/03/2016, 24/03/2016
35193                            <1>       ; 11/03/2016 (TRDOS 386 =  TRDOS v2.0)
35194                            <1>       ; 30/07/2011 (DRV_FAT.ASM)
35195                            <1>       ; 11/09/2010
35196                            <1>       ; INPUT ->
35197                            <1>       ;      ESI = Logical dos drv desc. table address
35198                            <1>       ;      EAX = Last cluster
35199                            <1>       ; OUTPUT ->
35200                            <1>       ;      ESI = Logical dos drv desc. table address
35201                            <1>       ;      EAX = New Last cluster (next cluster)
35202                            <1>       ;      cf = 1 -> error code in EAX (AL)
35203                            <1>       ;      cf = 1 -> DX = sectors per cluster
35204                            <1>       ;      ECX = Free sectors
35205                            <1>       ; NOTE:
35206                            <1>       ; This procedure does not update lm date&time !
35207                            <1>       ;
35208                            <1>       ; (Modified registers: EAX, EBX, ECX, EDX, EDI)
35209                            <1>       ;
35210                            <1>
35211 0000C226 A3[805A0100]      <1>       mov    [FAT_anc_LCluster], eax
35212                            <1>
35213 0000C22B E844F9FFFF        <1>       call   get_first_free_cluster
35214 0000C230 720B              <1>       jc     short loc_add_new_cluster_retn
35215                            <1>       ; EAX >= 2 and EAX < FFFFFFFFh is valid
35216                            <1>
35217 0000C232 89C2              <1>       mov    edx, eax
35218                            <1>
35219 0000C234 42                <1>       inc    edx
35220                            <1>       ;jnz   short loc_add_new_cluster_check_ffc_eax
35221 0000C235 7516              <1>       jnz    short loc_add_new_cluster_save_fcc
35222                            <1>
35223                            <1> loc_add_new_cluster_no_disk_space_retn:
35224 0000C237 B827000000        <1>       mov    eax, 27h ; MSDOS err => insufficient disk space
35225                            <1> loc_add_new_cluster_stc_retn:
35226 0000C23C F9                <1>       stc
35227                            <1> loc_add_new_cluster_retn:
35228 0000C23D 0FB65E13          <1>       movzx  ebx, byte [esi+LD_BPB+SecPerClust]
35229 0000C241 8B4E74            <1>       mov    ecx, [esi+LD_FreeSectors]
35230                            <1>       ;xor   edx, edx
35231                            <1>       ;stc
35232 0000C244 C3                <1>       retn
35233                            <1>
35234                            <1> loc_anc_invalid_format_stc_retn:
35235 0000C245 F9                <1>       stc
35236                            <1> loc_add_new_cluster_invalid_format_retn:
35237                            <1>       ; 15/10/2016 (0Bh -> 28)
35238 0000C246 B81C000000        <1>       mov    eax, 28 ; Invalid format
35239 0000C24B EBF0              <1>       jmp    short loc_add_new_cluster_retn
35240                            <1>
35241                            <1> ;loc_add_new_cluster_check_ffc_eax:
35242                            <1> ;     cmp    eax, 2
35243                            <1> ;     jb     short loc_add_new_cluster_invalid_format_retn
35244                            <1>
35245                            <1> loc_add_new_cluster_save_fcc:
35246 0000C24D A3[845A0100]      <1>       mov    [FAT_anc_FFCluster], eax
35247                            <1>
35248 0000C252 83E802            <1>       sub    eax, 2
35249 0000C255 0FB65E13          <1>         movzx   ebx, byte [esi+LD_BPB+SecPerClust]
35250 0000C259 F7E3              <1>       mul    ebx
35251 0000C25B 09D2              <1>       or     edx, edx
35252 0000C25D 75E6              <1>       jnz    short loc_anc_invalid_format_stc_retn
35253                            <1>
35254                            <1> loc_add_new_cluster_allocate_cluster:
35255                            <1>       ; 18/03/2016
35256 0000C25F 92                <1>       xchg   edx, eax ; eax = 0
35257                            <1>       ; 16/05/2016
35258                            <1>       ;cmp   [ClusterBuffer_Valid], al ; 0
35259                            <1>       ;jna   short loc_anc_clear_cluster_buffer
35260                            <1>       ;; 'copy' command,
35261                            <1>       ;; writing destination file clust after reading source file clust
35262                            <1>       ;mov   [ClusterBuffer_Valid], al ; 0 ; reset
35263                            <1>       ;jmp   short loc_add_new_cluster_write_nc_to_disk
35264                            <1>
35265                            <1> loc_anc_clear_cluster_buffer:
35266                            <1>       ; 11/03/2016
35267                            <1>       ; Clear buffer
35268 0000C260 BF00000700        <1>       mov    edi, Cluster_Buffer ; 70000h (for current TRDOS 386 version)
35269 0000C265 89D9              <1>       mov    ecx, ebx ; sector count
35270 0000C267 C1E107            <1>       shl    ecx, 7 ; 1 sector = 512 bytes -> 128 double words
35271                            <1>       ;xor   eax, eax ; 0
35272 0000C26A F3AB              <1>       rep    stosd
35273                            <1>
35274                            <1> loc_add_new_cluster_write_nc_to_disk:
35275                            <1>       ; 11/03/2016
35276                            <1>       ;xchg  eax, edx ; edx = 0, eax = sector offset
35277 0000C26C 89D0              <1>       mov    eax, edx
35278 0000C26E 034668            <1>         add     eax, [esi+LD_DATABegin]
35279 0000C271 72D3              <1>       jc     short loc_add_new_cluster_invalid_format_retn
35280                            <1>
35281 0000C273 89D9              <1>       mov    ecx, ebx ; ECX = sectors per cluster (<256)
35282 0000C275 BB00000700        <1>       mov    ebx, Cluster_Buffer
35283 0000C27A E8102F0000        <1>       call   disk_write
35284 0000C27F 7307              <1>       jnc    short loc_add_new_cluster_update_fat_nlc
35285                            <1>
35286                            <1>       ; 15/10/2016 (1Dh -> 18)
```

```
35287 0000C281 B812000000          <1>       mov    eax, 18 ; Write Error
35288 0000C286 EBB4                <1>       jmp    short loc_add_new_cluster_stc_retn
35289                              <1>
35290                              <1> loc_add_new_cluster_update_fat_nlc:
35291 0000C288 A1[845A0100]        <1>       mov    eax, [FAT_anc_FFCluster]
35292 0000C28D 31C9                <1>       xor    ecx, ecx
35293 0000C28F 890D[AE560100]      <1>       mov    [FAT_ClusterCounter], ecx ; 0 ; reset
35294 0000C295 49                  <1>       dec    ecx ; 0FFFFFFFFh
35295 0000C296 E8ACF9FFFF          <1>       call   update_cluster
35296 0000C29B 7304                <1>       jnc    short loc_add_new_cluster_update_fat_plc
35297 0000C29D 09C0                <1>       or     eax, eax ;EAX = 0 -> cluster value is 0 or eocc
35298 0000C29F 759B                <1>       jnz    short loc_add_new_cluster_stc_retn
35299                              <1>
35300                              <1> loc_add_new_cluster_update_fat_plc:
35301 0000C2A1 A1[805A0100]        <1>       mov    eax, [FAT_anc_LCluster]
35302 0000C2A6 8B0D[845A0100]      <1>       mov    ecx, [FAT_anc_FFCluster]
35303 0000C2AC E896F9FFFF          <1>       call   update_cluster
35304 0000C2B1 7314                <1>       jnc    short loc_add_new_cluster_save_fat_buffer
35305 0000C2B3 09C0                <1>       or     eax, eax ; EAX = 0 -> cluster value is 0 or eocc
35306 0000C2B5 7410                <1>       jz     short loc_add_new_cluster_save_fat_buffer
35307                              <1>
35308                              <1> loc_anc_save_fat_buffer_err_retn:
35309                              <1>       ;cmp   byte [FAT_ClusterCounter], 1
35310                              <1>       ;jb    short loc_add_new_cluster_retn
35311                              <1>
35312 0000C2B7 66BB00FF           <1>       mov    bx, 0FF00h ; recalculate free space (BL = 0)
35313                              <1>                         ; (BH = FFh -> Use ESI as Drv Param. Tbl.)
35314 0000C2BB 50                  <1>       push   eax
35315 0000C2BC E8D8FCFFFF          <1>       call   calculate_fat_freespace
35316 0000C2C1 58                  <1>       pop    eax
35317 0000C2C2 E975FFFFFF          <1>         jmp     loc_add_new_cluster_stc_retn
35318                              <1>
35319                              <1> loc_add_new_cluster_save_fat_buffer:
35320                              <1>       ;cmp   byte [FAT_BuffValidData], 2
35321                              <1>       ;jne   short loc_add_new_cluster_calc_FAT_freespace
35322                              <1>       ;Byte [FAT_BuffValidData] =  2
35323 0000C2C7 E838FCFFFF          <1>       call   save_fat_buffer
35324 0000C2CC 72E9                <1>       jc     short loc_anc_save_fat_buffer_err_retn
35325                              <1>
35326                              <1> loc_add_new_cluster_calc_FAT_freespace:
35327                              <1>       ;mov   eax, 1 ; Only one Cluster
35328 0000C2CE A1[AE560100]        <1>       mov    eax, [FAT_ClusterCounter]
35329 0000C2D3 66BB01FF            <1>       mov    bx, 0FF01h ; BH = FFh -> ESI -> Dos drv desc. table
35330                              <1>               ; BL = 1 -> add cluster
35331 0000C2D7 B301                <1>       mov    bl, 01h ; BL = 1 -> add clusters
35332                              <1>       ; NOTE: EAX value will be added to Free Cluster Count
35333                              <1>       ; (Free Cluster Count is decreased when EAX value is negative)
35334 0000C2D9 E8BBFCFFFF          <1>         call    calculate_fat_freespace
35335                              <1>       ;ECX = 0 -> no error, ECX > 0 -> error or invalid return
35336 0000C2DE 21C9                <1>       and    ecx, ecx ; ECX = 0 -> valid free sector count
35337 0000C2E0 7409                <1>       jz     short loc_add_new_cluster_return_cluster_number
35338                              <1>
35339                              <1> loc_add_new_cluster_recalc_FAT_freespace:
35340 0000C2E2 66BB00FF            <1>       mov    bx, 0FF00h  ; recalculate free space
35341 0000C2E6 E8AEFCFFFF          <1>         call    calculate_fat_freespace
35342                              <1>       ; cf = 0
35343                              <1> loc_add_new_cluster_return_cluster_number:
35344 0000C2EB 89C1                <1>       mov    ecx, eax ; Free sector count
35345 0000C2ED A1[845A0100]        <1>       mov    eax, [FAT_anc_FFCluster]
35346 0000C2F2 0FB65E13            <1>       movzx  ebx, byte [esi+LD_BPB+SecPerClust]
35347                              <1>       ;mov   edi, Cluster_Buffer
35348 0000C2F6 31D2                <1>       xor    edx, edx
35349 0000C2F8 C3                  <1>         retn
35350                              <1>
35351                              <1> write_cluster:
35352                              <1>       ; 15/10/2016
35353                              <1>       ; 21/03/2016 (TRDOS 386 =  TRDOS v2.0)
35354                              <1>       ;
35355                              <1>       ; INPUT ->
35356                              <1>       ;      EAX = Cluster Number (Sector index for SINGLIX FS)
35357                              <1>       ;      ESI = Logical DOS Drive Description Table address
35358                              <1>       ;      EBX = Cluster (File R/W) Buffer address (max. 64KB)
35359                              <1>       ;      Only for SINGLIX FS:
35360                              <1>       ;      EDX = File Number (The 1st FDT address)
35361                              <1>       ; OUTPUT ->
35362                              <1>       ;      cf = 1 -> Cluster can not be written onto disk
35363                              <1>       ;          EAX > 0 -> Error number
35364                              <1>       ;      cf = 0 -> Cluster has been written successfully
35365                              <1>       ;
35366                              <1>       ; (Modified registers: EAX, ECX, EBX, EDX)
35367                              <1>
35368 0000C2F9 0FB64E13            <1>       movzx  ecx, byte [esi+LD_BPB+BPB_SecPerClust]
35369                              <1>       ; CL = 1 = [esi+LD_FS_Reserved2] ; SectPerClust for Singlix FS
35370                              <1>
35371                              <1> write_file_sectors: ; 16/03/2016
35372 0000C2FD 807E0300            <1>       cmp    byte [esi+LD_FATType], 0
35373 0000C301 761C                <1>       jna    short write_fs_cluster
35374                              <1>
35375                              <1> write_fat_file_sectors:
35376 0000C303 83E802              <1>       sub    eax, 2 ; Beginning cluster number is always 2
35377 0000C306 0FB65613            <1>       movzx  edx, byte [esi+LD_BPB+BPB_SecPerClust] ; 18/03/2016
35378 0000C30A F7E2                <1>       mul    edx
35379 0000C30C 034668              <1>       add    eax, [esi+LD_DATABegin] ; absolute address of the cluster
35380                              <1>
35381                              <1>       ; EAX = Disk sector address
35382                              <1>       ; ECX = Sector count
35383                              <1>       ; EBX = Buffer address
35384                              <1>       ; (EDX = 0)
35385                              <1>       ; ESI = Logical DOS drive description table address
35386                              <1>
35387 0000C30F E87B2E0000          <1>       call   disk_write
35388 0000C314 7306                <1>       jnc    short wclust_retn
```

```
35389                                  <1>
35390                                  <1>       ; 15/10/2016 (1Dh -> 18)
35391 0000C316 B812000000              <1>       mov    eax, 18 ; Drive not ready or write error !
35392 0000C31B C3                      <1>       retn
35393                                  <1>
35394                                  <1> wclust_retn:
35395 0000C31C 29C0                    <1>       sub    eax, eax ; 0
35396 0000C31E C3                      <1>       retn
35397                                  <1>
35398                                  <1> write_fs_cluster:
35399                                  <1>       ; 21/03/2016 (TRDOS 386 =  TRDOS v2.0)
35400                                  <1>       ; Singlix FS
35401                                  <1>
35402                                  <1>       ; EAX = Cluster number is sector index number of the file (eax)
35403                                  <1>
35404                                  <1>       ; EDX = File number is the first File Descriptor Table address
35405                                  <1>       ;      of the file. (Absolute address of the FDT).
35406                                  <1>
35407                                  <1>       ; eax = sector index (0 for the first sector)
35408                                  <1>       ; edx = FDT0 address
35409                                  <1>             ; 64 KB buffer = 128 sectors (limit)
35410 0000C31F B980000000              <1>       mov    ecx, 128 ; maximum count of sectors (before eof)
35411 0000C324 E801000000              <1>       call   write_fs_sectors
35412 0000C329 C3                      <1>       retn
35413                                  <1>
35414                                  <1> write_fs_sectors:
35415                                  <1>       ; 21/03/2016 (TRDOS 386 =  TRDOS v2.0)
35416 0000C32A F9                      <1>       stc
35417 0000C32B C3                      <1>       retn
35418                                  <1>
35419                                  <1> get_cluster_by_index:
35420                                  <1>       ; 29/04/2016 (TRDOS 386 =  TRDOS v2.0)
35421                                  <1>       ; INPUT ->
35422                                  <1>       ;      EAX = Beginning cluster
35423                                  <1>       ;      EDX = Sector index in disk/file section
35424                                  <1>       ;            (Only for SINGLIX file system!)
35425                                  <1>       ;      ECX = Cluster sequence number after the beginning cluster
35426                                  <1>       ;      ESI = Logical DOS Drive Description Table address
35427                                  <1>       ; OUTPUT ->
35428                                  <1>       ;      EAX = Cluster number
35429                                  <1>       ;      cf = 1 -> Error code in AL (EAX)
35430                                  <1>       ;
35431                                  <1>       ;(Modified registers: EAX, ECX, EBX, EDX)
35432                                  <1>       ;
35433 0000C32C 807E0301               <1>       cmp    byte [esi+LD_FATType], 1
35434 0000C330 721E                    <1>         jb      short get_fs_section_by_index
35435                                  <1>
35436 0000C332 3B4E78                  <1>       cmp    ecx, [esi+LD_Clusters]
35437 0000C335 7207                    <1>       jb     short gcbi_1
35438                                  <1> gcbi_0:
35439 0000C337 F9                      <1>       stc
35440 0000C338 B823000000              <1>       mov    eax, 23h ; Cluster not available !
35441                                  <1>                       ; MSDOS error code: FCB unavailable
35442 0000C33D C3                      <1>       retn
35443                                  <1> gcbi_1:
35444 0000C33E 51                      <1>       push   ecx
35445 0000C33F E8D9F5FFFF              <1>       call   get_next_cluster
35446 0000C344 59                      <1>       pop    ecx
35447 0000C345 7203                    <1>       jc     short gcbi_3
35448 0000C347 E2F5                    <1>       loop   gcbi_1
35449                                  <1> gcbi_2:
35450 0000C349 C3                      <1>       retn
35451                                  <1> gcbi_3:
35452 0000C34A 09C0                    <1>       or     eax, eax
35453 0000C34C 74E9                    <1>       jz     short gcbi_0
35454 0000C34E F5                      <1>       cmc    ; stc
35455 0000C34F C3                      <1>       retn
35456                                  <1>
35457                                  <1> get_fs_section_by_index:
35458                                  <1>       ; 29/04/2016 (TRDOS 386 =  TRDOS v2.0)
35459                                  <1>       ; INPUT ->
35460                                  <1>       ;      EAX = Beginning FDT number/address
35461                                  <1>       ;      EDX = Sector index in disk/file section
35462                                  <1>       ;      ECX = Sector sequence number after the beginning FDT
35463                                  <1>       ;      ESI = Logical DOS Drive Description Table address
35464                                  <1>       ; OUTPUT ->
35465                                  <1>       ;      EAX = FDT number/address
35466                                  <1>       ;      EDX = Sector index of the section (0,1,2,3,4...)
35467                                  <1>       ;      cf = 1 -> Error code in AL (EAX)
35468                                  <1>       ;
35469                                  <1>       ;(Modified registers: EAX, ECX, EBX, EDX)
35470                                  <1>       ;
35471 0000C350 B8FFFFFFFF              <1>       mov    eax, 0FFFFFFFFh
35472 0000C355 C3                      <1>       retn
35473                                  <1>
35474                                  <1> get_last_section:
35475                                  <1>       ; 22/10/2016 (TRDOS 386 =  TRDOS v2.0)
35476                                  <1>       ; INPUT ->
35477                                  <1>       ;      EAX = (The 1st) FDT number/address
35478                                  <1>       ;      ESI = Logical DOS Drive Description Table address
35479                                  <1>       ; OUTPUT ->
35480                                  <1>       ;      EAX = FDT number/address of the last section
35481                                  <1>       ;      EDX = Last sector of the section (0,1,2,3,4...)
35482                                  <1>       ;      [glc_index] = sector index number of the last sector
35483                                  <1>       ;                    (for file, not for the last section)
35484                                  <1>       ;
35485                                  <1>       ;      cf = 1 -> Error code in AL (EAX)
35486                                  <1>       ;
35487                                  <1>       ;(Modified registers: EAX, ECX, EBX, EDX)
35488                                  <1>       ;
35489 0000C356 B800000000              <1>       mov    eax, 0
35490 0000C35B BA00000000              <1>       mov    edx, 0
```

```
35491 0000C360 C3                   <1>       retn
35492                               %include 'trdosk6.s' ; 24/01/2016
35493                               <1> ; ****************************************************************************
35494                               <1> ; TRDOS386.ASM (TRDOS 386 Kernel - v2.0.0) - MAIN PROGRAM : trdosk6.s
35495                               <1> ; ----------------------------------------------------------------------------
35496                               <1> ; Last Update: 27/05/2017
35497                               <1> ; ----------------------------------------------------------------------------
35498                               <1> ; Beginning: 24/01/2016
35499                               <1> ; ----------------------------------------------------------------------------
35500                               <1> ; Assembler: NASM version 2.11 (trdos386.s)
35501                               <1> ; ----------------------------------------------------------------------------
35502                               <1> ; Derived from 'Retro UNIX 386 Kernel - v0.2.1.0' source code by Erdogan Tan
35503                               <1> ; u1.s (27/17/2015), u2.s (03/01/2016)
35504                               <1> ; ****************************************************************************
35505                               <1> ; Derived from TRDOS Operating System v1.0 (8086) source code by Erdogan Tan
35506                               <1> ; TRDOS2.ASM (09/11/2011)
35507                               <1> ; ----------------------------------------------------------------------------
35508                               <1> ; INT_21H.ASM (c) 2009-2011 Erdogan TAN  [14/11/2009] Last Update: 08/11/2011
35509                               <1>
35510                               <1> sysent: ; < enter to system call >
35511                               <1>       ; 17/03/2017
35512                               <1>       ; 03/03/2017
35513                               <1>       ; 19/02/2017
35514                               <1>       ; 13/01/2017
35515                               <1>       ; 06/06/2016
35516                               <1>       ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
35517                               <1>       ; 16/04/2015 - 19/10/2015 (Retro UNIX 386 v1)
35518                               <1>       ; 10/04/2013 - 18/01/2014 (Retro UNIX 8086 v1)
35519                               <1>       ;
35520                               <1>       ; 'unkni' or 'sysent' is sytem entry from various traps.
35521                               <1>       ; The trap type is determined and an indirect jump is made to
35522                               <1>       ; the appropriate system call handler. If there is a trap inside
35523                               <1>       ; the system a jump to panic is made. All user registers are saved
35524                               <1>       ; and u.sp points to the end of the users stack. The sys (trap)
35525                               <1>       ; instructor is decoded to get the the system code part (see
35526                               <1>       ; trap instruction in the PDP-11 handbook) and from this
35527                               <1>       ; the indirect jump address is calculated. If a bad system call is
35528                               <1>       ; made, i.e., the limits of the jump table are exceeded, 'badsys'
35529                               <1>       ; is called. If the call is legitimate control passes to the
35530                               <1>       ; appropriate system routine.
35531                               <1>       ;
35532                               <1>       ; Calling sequence:
35533                               <1>       ;     Through a trap caused by any sys call outside the system.
35534                               <1>       ; Arguments:
35535                               <1>       ;     Arguments of particular system call.
35536                               <1>       ; ..........................................................
35537                               <1>       ;
35538                               <1>       ; Retro UNIX 8086 v1 modification:
35539                               <1>       ;      System call number is in EAX register.
35540                               <1>       ;
35541                               <1>       ;      Other parameters are in EDX, EBX, ECX, ESI, EDI, EBP
35542                               <1>       ;     registers depending of function details.
35543                               <1>       ;
35544                               <1>       ; 16/04/2015
35545 0000C361 368925[5C030600]     <1>         mov     [ss:u.sp], esp ; Kernel stack points to return address
35546                               <1>
35547                               <1>       ; save user registers
35548 0000C368 1E                   <1>       push   ds
35549 0000C369 06                   <1>       push   es
35550 0000C36A 0FA0                 <1>       push   fs
35551 0000C36C 0FA8                 <1>       push   gs
35552 0000C36E 60                   <1>       pushad  ; eax, ecx, edx, ebx, esp -before pushad-, ebp, esi, edi
35553                               <1>       ;
35554                               <1>       ; ESPACE = [ss:u.sp] - esp ; 4*12 = 48 ; 17/09/2015 ; 06/06/2016
35555                               <1>       ;     (ESPACE is size of space in kernel stack
35556                               <1>       ;     for saving/restoring user registers.)
35557                               <1>       ;
35558 0000C36F 50                   <1>       push   eax ; 01/07/2015
35559 0000C370 66B81000             <1>       mov     ax, KDATA
35560 0000C374 8ED8                 <1>         mov     ds, ax
35561 0000C376 8EC0                 <1>         mov     es, ax
35562 0000C378 8EE0                 <1>         mov     fs, ax
35563 0000C37A 8EE8                 <1>         mov     gs, ax
35564 0000C37C A1[C84D0100]         <1>       mov    eax, [k_page_dir]
35565 0000C381 0F22D8               <1>       mov    cr3, eax
35566 0000C384 58                   <1>       pop    eax ; 01/07/2015
35567                               <1>       ; 19/10/2015
35568 0000C385 FC                   <1>       cld
35569                               <1>       ;
35570 0000C386 FE05[5B030600]       <1>       inc    byte [sysflg]
35571                               <1>              ; incb sysflg / indicate a system routine is in progress
35572 0000C38C FB                   <1>         sti  ; 18/01/2014
35573 0000C38D 0F851EA0FFFF         <1>       jnz     panic ; 24/05/2013
35574                               <1>              ; beq 1f
35575                               <1>              ; jmp panic ; / called if trap inside system
35576                               <1> ;1:
35577                               <1>       ; 17/03/2017
35578 0000C393 80642438FE           <1>       and    byte [esp+ESPACE+8], ~1 ; clear carry flag
35579                               <1>
35580                               <1>       ; 16/04/2015
35581 0000C398 A3[64030600]         <1>       mov    [u.r0], eax
35582 0000C39D 8925[60030600]       <1>       mov    [u.usp], esp ; kernel stack points to user's registers
35583                               <1>
35584                               <1>       ; 13/01/2017 (TRDOS 386 Feaure only !)
35585 0000C3A3 803D[D4030600]00     <1>       cmp    byte [u.t_lock], 0 ; timer interrupt lock ?
35586 0000C3AA 0F8799010000         <1>       ja     sysrele                ; yes, sys release only !!!
35587                               <1>
35588                               <1>              ; mov $s.syst+2,clockp
35589                               <1>              ; mov r0,-(sp) / save user registers
35590                               <1>              ; mov sp,u.r0 / pointer to bottom of users stack
35591                               <1>                       ; / in u.r0
35592                               <1>              ; mov r1,-(sp)
```

```
35593                                  <1>             ; mov r2,-(sp)
35594                                  <1>             ; mov r3,-(sp)
35595                                  <1>             ; mov r4,-(sp)
35596                                  <1>             ; mov r5,-(sp)
35597                                  <1>             ; mov ac,-(sp) / "accumulator" register for extended
35598                                  <1>                             ; / arithmetic unit
35599                                  <1>             ; mov mq,-(sp) / "multiplier quotient" register for the
35600                                  <1>                             ; / extended arithmetic unit
35601                                  <1>             ; mov sc,-(sp) / "step count" register for the extended
35602                                  <1>                             ; / arithmetic unit
35603                                  <1>             ; mov sp,u.sp / u.sp points to top of users stack
35604                                  <1>             ; mov 18.(sp),r0 / store pc in r0
35605                                  <1>             ; mov -(r0),r0 / sys inst in r0      10400xxx
35606                                  <1>             ; sub $sys,r0 / get xxx code
35607 0000C3B0 C1E002                 <1>         shl    eax, 2
35608                                  <1>             ; asl r0 / multiply by 2 to jump indirect in bytes
35609 0000C3B3 3DB4000000             <1>         cmp    eax, end_of_syscalls - syscalls
35610                                  <1>             ; cmp r0,$2f-1f / limit of table (35) exceeded
35611                                  <1>         ;jnb   short badsys
35612                                  <1>             ; bhis badsys / yes, bad system call
35613 0000C3B8 F5                     <1>         cmc
35614 0000C3B9 9C                     <1>         pushf
35615 0000C3BA 50                     <1>         push   eax
35616 0000C3BB 8B2D[5C030600]         <1>         mov    ebp, [u.sp] ; Kernel stack at the beginning of sys call
35617 0000C3C1 B0FE                   <1>         mov    al, 0FEh ; 11111110b
35618 0000C3C3 1400                   <1>         adc    al, 0 ; al = al + cf
35619 0000C3C5 204508                 <1>         and    [ebp+8], al ; flags (reset carry flag)
35620                                  <1>             ; bic $341,20.(sp) / set users processor priority to 0
35621                                  <1>                             ; / and clear carry bit
35622 0000C3C8 5D                     <1>         pop    ebp ; eax
35623 0000C3C9 9D                     <1>         popf
35624 0000C3CA 0F8204020000           <1>           jc     badsys
35625 0000C3D0 A1[64030600]           <1>         mov    eax, [u.r0]
35626                                  <1>         ; system call registers: EAX, EDX, ECX, EBX, ESI, EDI
35627 0000C3D5 FFA5[DBC30000]         <1>         jmp    dword [ebp+syscalls]
35628                                  <1>                ; jmp *1f(r0) / jump indirect thru table of addresses
35629                                  <1>                              ; / to proper system routine.
35630                                  <1> syscalls: ; 1:
35631                                  <1>         ; 28/02/2017
35632                                  <1>         ; 20/02/2017
35633                                  <1>         ; 19/02/2017
35634                                  <1>         ; 15/10/2016
35635                                  <1>         ; 20/05/2016
35636                                  <1>         ; 19/05/2016
35637                                  <1>         ; 16/05/2016
35638                                  <1>         ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
35639                                  <1>         ; 21/09/2015
35640                                  <1>         ; 01/07/2015
35641                                  <1>         ; 16/04/2015 (32 bit address modification)
35642                                  <1>         ;dd sysrele  ; / 0
35643 0000C3DB [B7E70000]              <1>         dd sysver    ; 0 ; Get TRDOS 386 version number (v2.0)
35644 0000C3DF [36C60000]              <1>         dd sysexit   ; / 1
35645 0000C3E3 [0CC80000]              <1>         dd sysfork   ; / 2
35646 0000C3E7 [40CC0000]              <1>         dd sysread   ; / 3
35647 0000C3EB [5FCC0000]              <1>         dd syswrite  ; / 4
35648 0000C3EF [F6C90000]              <1>         dd sysopen   ; / 5
35649 0000C3F3 [17CC0000]              <1>         dd sysclose  ; / 6
35650 0000C3F7 [8EC70000]              <1>         dd syswait   ; / 7
35651 0000C3FB [24C90000]              <1>         dd syscreat  ; / 8
35652 0000C3FF [0ED80000]              <1>         dd syslink   ; / 9
35653 0000C403 [59D80000]              <1>         dd sysunlink ; / 10
35654 0000C407 [E6D80000]              <1>         dd sysexec   ; / 11
35655 0000C40B [DDDC0000]              <1>         dd syschdir  ; / 12
35656 0000C40F [BCDD0000]              <1>         dd systime   ; / 13
35657 0000C413 [D9CB0000]              <1>         dd sysmkdir  ; / 14
35658 0000C417 [2FDD0000]              <1>         dd syschmod  ; / 15
35659 0000C41B [8CDD0000]              <1>         dd syschown  ; / 16
35660 0000C41F [EFDD0000]              <1>         dd sysbreak  ; / 17
35661 0000C423 [8DDA0000]              <1>         dd sysstat   ; / 18
35662 0000C427 [34DE0000]              <1>         dd sysseek   ; / 19
35663 0000C42B [46DE0000]              <1>         dd systell   ; / 20
35664 0000C42F [2FDF0000]              <1>         dd sysmount  ; / 21
35665 0000C433 [43DF0000]              <1>         dd sysumount ; / 22
35666 0000C437 [B8DE0000]              <1>         dd syssetuid ; / 23
35667 0000C43B [E9DE0000]              <1>         dd sysgetuid ; / 24
35668 0000C43F [CBDD0000]              <1>         dd sysstime  ; / 25
35669 0000C443 [ACDE0000]              <1>         dd sysquit   ; / 26
35670 0000C447 [A0DE0000]              <1>         dd sysintr   ; / 27
35671 0000C44B [A1DA0000]              <1>         dd sysfstat  ; / 28
35672 0000C44F [F6CC0000]              <1>         dd sysemt    ; / 29
35673 0000C453 [A7CE0000]              <1>         dd sysmdate  ; / 30
35674                                  <1>         ;dd sysstty  ; / 31
35675 0000C457 [BBCE0000]              <1>         dd sysvideo  ; 31 ; TRDOS 386 Video Functions (16/05/2016)
35676                                  <1>         ;dd sysgtty  ; / 32
35677 0000C45B [91FA0000]              <1>         dd sysaudio  ; 32 ; TRDOS 386 Audio Functions (16/05/2016)
35678                                  <1>         ;dd sysilgins; / 33
35679 0000C45F [0FCD0000]              <1>         dd systimer  ; 33 ; TRDOS 386 Timer Functions (18/05/2016)
35680 0000C463 [57DF0000]              <1>         dd syssleep  ; 34 ; Retro UNIX 8086 v1 feature only !
35681                                  <1>                              ; 11/06/2014
35682 0000C467 [86DF0000]              <1>         dd sysmsg    ; 35 ; Retro UNIX 386 v1 feature only !
35683                                  <1>                              ; 01/07/2015
35684 0000C46B [5CE00000]              <1>         dd sysgeterr ; 36 ; Retro UNIX 386 v1 feature only !
35685                                  <1>                              ; 21/09/2015 - get last error number
35686 0000C46F [5CF10000]              <1>         dd sysfpstat ; 37 ; TRDOS 386 FPU state option (28/02/2017)
35687 0000C473 [D5E70000]              <1>         dd syspri    ; 38 ; change priority - TRDOS 386 (20/05/2016)
35688 0000C477 [49C50000]              <1>         dd sysrele   ; 39 ; TRDOS 386 (19/05/2016) (0 -> 39)
35689 0000C47B [08E90000]              <1>         dd sysfff    ; 40 ; Find First File - TRDOS 386 (15/10/2016)
35690 0000C47F [E7E90000]              <1>         dd sysfnf    ; 41 ; Find Next File - TRDOS 386 (15/10/2016)
35691 0000C483 [57F00000]              <1>         dd sysalloc  ; 42 ; Allocate contiguous memory block/pages
35692                                  <1>                              ; TRDOS 386 (19/02/2017) DMA buff fuctions
35693 0000C487 [05F10000]              <1>         dd sysdalloc ; 43 ; Deallocate contiguous memory block/pages
35694                                  <1>                              ; TRDOS 386 (19/02/2017) DMA buff fuctions
```

```
35695 0000C48B [40F10000]          <1>       dd syscalbac ; 44 ; IRQ Callback and Signal Response Byte
35696                              <1>                          ; service setup - TRDOS 386 (20/02/2017)
35697                              <1>
35698                              <1> end_of_syscalls:
35699                              <1>
35700                              <1> error:
35701                              <1>       ; 18/05/2016
35702                              <1>       ; 13/05/2016
35703                              <1>       ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
35704                              <1>       ; 16/04/2015 - 17/09/2015 (Retro UNIX 386 v1)
35705                              <1>       ; 10/04/2013 - 07/08/2013 (Retro UNIX 8086 v1)
35706                              <1>       ;
35707                              <1>       ; 'error' merely sets the error bit off the processor status (c-bit)
35708                              <1>       ; then falls right into the 'sysret', 'sysrele' return sequence.
35709                              <1>       ;
35710                              <1>       ; INPUTS -> none
35711                              <1>       ; OUTPUTS ->
35712                              <1>       ;     processor status - carry (c) bit is set (means error)
35713                              <1>       ;
35714                              <1>       ; 26/05/2013 (Stack pointer must be reset here!
35715                              <1>       ;           Because, jumps to error procedure
35716                              <1>       ;           disrupts push-pop nesting balance)
35717                              <1>       ;
35718 0000C48F 8B2D[5C030600]      <1>       mov   ebp, [u.sp] ; interrupt (system call) return (iretd) address
35719 0000C495 804D0801            <1>       or    byte [ebp+8], 1  ; set carry bit of flags register
35720                              <1>                              ; (system call will return with cf = 1)
35721                              <1>             ; bis $1,20.(r1) / set c bit in processor status word below
35722                              <1>                               ; / users stack
35723                              <1>       ; 17/09/2015
35724 0000C499 83ED30              <1>       sub   ebp, ESPACE ; 48 ; total size of stack frame ('sysdefs.inc')
35725                              <1>                              ; for saving/restoring user registers
35726                              <1>       ;cmp   ebp, [u.usp]
35727                              <1>       ;je    short err0
35728 0000C49C 892D[60030600]      <1>       mov   [u.usp], ebp
35729                              <1> ;err0:
35730                              <1>       ; 01/09/2015
35731 0000C4A2 8B25[60030600]      <1>       mov   esp, [u.usp]     ; Retro Unix 8086 v1 modification!
35732                              <1>                              ; 10/04/2013
35733                              <1>                              ; (If an I/O error occurs during disk I/O,
35734                              <1>                              ; related procedures will jump to 'error'
35735                              <1>                              ; procedure directly without returning to
35736                              <1>                              ; the caller procedure. So, stack pointer
35737                              <1>                                   ; must be restored here.)
35738                              <1>       ; 13/05/2016
35739                              <1>       ; NOTE: (The last) error code is in 'u.error', it can be retrieved by
35740                              <1>       ;     'get last error' system call later.
35741                              <1>
35742                              <1>       ; 03/09/2015 - 09/06/2015 - 07/08/2013
35743 0000C4A8 C605[C6030600]00    <1>       mov   byte [u.kcall], 0 ; namei_r, mkdir_w reset
35744                              <1>
35745                              <1> sysret: ; < return from system call>
35746                              <1>       ; 01/03/2017
35747                              <1>       ; 28/02/2017
35748                              <1>       ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
35749                              <1>       ; 16/04/2015 - 10/09/2015 (Retro UNIX 386 v1)
35750                              <1>       ; 10/04/2013 - 23/02/2014 (Retro UNIX 8086 v1)
35751                              <1>       ;
35752                              <1>       ; 'sysret' first checks to see if process is about to be
35753                              <1>       ; terminated (u.bsys). If it is, 'sysexit' is called.
35754                              <1>       ; If not, following happens:
35755                              <1>       ;     1) The user's stack pointer is restored.
35756                              <1>       ;     2) r1=0 and 'iget' is called to see if last mentioned
35757                              <1>       ;        i-node has been modified. If it has, it is written out
35758                              <1>       ;        via 'ppoke'.
35759                              <1>       ;     3) If the super block has been modified, it is written out
35760                              <1>       ;        via 'ppoke'.
35761                              <1>       ;     4) If the dismountable file system's super block has been
35762                              <1>       ;        modified, it is written out to the specified device
35763                              <1>       ;        via 'ppoke'.
35764                              <1>       ;     5) A check is made if user's time quantum (uquant) ran out
35765                              <1>       ;        during his execution. If so, 'tswap' is called to give
35766                              <1>       ;        another user a chance to run.
35767                              <1>       ;     6) 'sysret' now goes into 'sysrele'.
35768                              <1>       ;         (See 'sysrele' for conclusion.)
35769                              <1>       ;
35770                              <1>       ; Calling sequence:
35771                              <1>       ;     jump table or 'br sysret'
35772                              <1>       ; Arguments:
35773                              <1>       ;     -
35774                              <1>       ; ....................................................
35775                              <1>       ;
35776                              <1>       ; ((AX=r1 for 'iget' input))
35777                              <1>       ;
35778 0000C4AF 31C0                <1>       xor   eax, eax ; 28/02/2017
35779                              <1> sysret0: ; 29/07/2015 (eax = 0, jump from sysexec)
35780 0000C4B1 FEC0                <1>       inc   al ; 04/05/2013
35781 0000C4B3 3805[B2030600]      <1>       cmp   [u.bsys], al ; 1
35782                              <1>             ; tstb u.bsys / is a process about to be terminated because
35783 0000C4B9 0F8377010000        <1>         jnb     sysexit ; 04/05/2013
35784                              <1>             ; bne sysexit / of an error? yes, go to sysexit
35785                              <1>       ;mov   esp, [u.usp] ; 24/05/2013 (that is not needed here)
35786                              <1>             ; mov u.sp,sp / no point stack to users stack
35787 0000C4BF FEC8                <1>       dec   al ; mov ax, 0
35788                              <1>             ; clr r1 / zero r1 to check last mentioned i-node
35789 0000C4C1 E8BF2C0000          <1>       call  iget
35790                              <1>             ; jsr r0,iget / if last mentioned i-node has been modified
35791                              <1>                           ; / it is written out
35792                              <1>       ; 10/01/2017
35793                              <1>       ; 09/01/2017
35794                              <1> ;sysrele: ; < release >
35795                              <1>       ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
35796                              <1>       ; 16/04/2015 - 14/10/2015 (Retro UNIX 386 v1)
```

```
35797                                  <1>       ; 10/04/2013 - 07/03/2014 (Retro UNIX 8086 v1)
35798                                  <1>       ;
35799                                  <1>       ; 'sysrele' first calls 'tswap' if the time quantum for a user is
35800                                  <1>       ;  zero (see 'sysret'). It then restores the user's registers and
35801                                  <1>       ; turns off the system flag. It then checked to see if there is
35802                                  <1>       ; an interrupt from the user by calling 'isintr'. If there is,
35803                                  <1>       ; the output gets flashed (see isintr) and interrupt action is
35804                                  <1>       ; taken by a branch to 'intract'. If there is no interrupt from
35805                                  <1>       ; the user, a rti is made.
35806                                  <1>       ;
35807                                  <1>       ; Calling sequence:
35808                                  <1>       ;     Fall through a 'bne' in 'sysret' & ?
35809                                  <1>       ; Arguments:
35810                                  <1>       ;     -
35811                                  <1>       ; ..........................................................
35812                                  <1>       ;
35813                                  <1>       ; 23/02/2014 (swapret)
35814                                  <1>       ; 22/09/2013
35815                                  <1> sysrel0: ;1:
35816 0000C4C6 803D[A8030600]00       <1>       cmp    byte [u.quant], 0 ; 16/05/2013
35817                                  <1>              ; tstb uquant / is the time quantum 0?
35818 0000C4CD 7705                   <1>         ja      short swapret
35819                                  <1>              ; bne 1f / no, don't swap it out
35820                                  <1> sysrelease: ; 07/12/2013 (jump from 'clock')
35821 0000C4CF E8AB210000             <1>       call   tswap
35822                                  <1>              ; jsr r0,tswap / yes, swap it out
35823                                  <1>
35824                                  <1> ; Retro Unix 8086 v1 feature: return from 'swap' to 'swapret' address.
35825                                  <1> swapret: ;1:
35826                                  <1>       ; 10/09/2015
35827                                  <1>       ; 01/09/2015
35828                                  <1>       ; 14/05/2015
35829                                  <1>       ; 16/04/2015 (Retro UNIX 386 v1 - 32 bit, pm modifications)
35830                                  <1>       ; 26/05/2013 (Retro UNIX 8086 v1)
35831                                  <1>       ; cli
35832                                  <1>       ; 24/07/2015
35833                                  <1>       ;
35834                                  <1>       ;; 'esp' must be already equal to '[u.usp]' here !
35835                                  <1>       ;; mov esp, [u.usp]
35836                                  <1>
35837                                  <1>       ; 22/09/2013
35838 0000C4D4 E8AD2C0000             <1>       call   isintr
35839                                  <1>       ; 20/10/2013
35840 0000C4D9 7405                   <1>       jz     short sysrel1
35841 0000C4DB E83F010000             <1>       call   intract
35842                                  <1>              ; jsr r0,isintr / is there an interrupt from the user
35843                                  <1>              ;     br intract / yes, output gets flushed, take interrupt
35844                                  <1>                                  ; / action
35845                                  <1> sysrel1:
35846 0000C4E0 FA                     <1>       cli    ; 14/10/2015
35847                                  <1> sysrel2:
35848                                  <1>       ; 28/02/2017
35849                                  <1>       ; Check if there is a (delayed) callback for current user/process
35850 0000C4E1 A0[D7030600]           <1>       mov    al, [u.irqwait]
35851 0000C4E6 240F                   <1>       and    al, 0Fh ; is there a waiting IRQ callback service ?
35852 0000C4E8 7444                   <1>       jz     short sysrel8 ; no
35853                                  <1>
35854                                  <1>       ; Set return to IRQ callback service and return from the service
35855 0000C4EA 0FB6D8                 <1>       movzx  ebx, al
35856 0000C4ED 883D[D7030600]         <1>       mov    [u.irqwait], bh ; 0 ; reset
35857 0000C4F3 8A9B[CC0B0100]         <1>       mov    bl, [ebx+IRQenum] ; (available) IRQ index +1 (1 to 9)
35858                                  <1>       ; 01/03/2017
35859 0000C4F9 FECB                   <1>       dec    bl ; IRQ index number, 0 to 8
35860 0000C4FB 7831                   <1>       js     short sysrel8 ; 0 -> FFh (not in use!?)
35861                                  <1>       ;
35862 0000C4FD A0[B3030600]           <1>       mov    al, [u.uno] ; current process (user) number
35863 0000C502 3883[FE600100]         <1>       cmp    [ebx+IRQ.owner], al
35864 0000C508 7524                   <1>       jne    short sysrel8 ; it is not the current user/process !?
35865 0000C50A F683[10610100]01       <1>       test   byte [ebx+IRQ.method], 1 ; callback ?
35866 0000C511 741B                   <1>       jz     short sysrel8 ; not a callback method !?
35867                                  <1>
35868 0000C513 8B93[22610100]         <1>       mov    edx, [ebx+IRQ.addr] ; IRQ callback service address (virtual)
35869 0000C519 C605[D8030600]01       <1>       mov    byte [u.r_lock], 1 ; IRQ callback service in progress flag
35870                                  <1>
35871 0000C520 E802220000             <1>       call   wswap ; save user's registers & status
35872                                  <1>                    ;      (for return from IRQ callback service)
35873                                  <1>
35874 0000C525 8B2D[5C030600]         <1>       mov    ebp, [u.sp]; kernel's stack, points to EIP (user)
35875 0000C52B 895500                 <1>       mov    [ebp], edx ; IRQ call back service address
35876                                  <1> sysrel8:
35877 0000C52E FE0D[5B030600]         <1>       dec    byte [sysflg]
35878                                  <1>              ; decb sysflg / turn system flag off
35879                                  <1>
35880 0000C534 A1[B8030600]           <1>       mov    eax, [u.pgdir]
35881 0000C539 0F22D8                 <1>       mov    cr3, eax  ; 1st PDE points to Kernel Page Table 0 (1st 4 MB)
35882                                  <1>                      ; (others are different than kernel page tables)
35883                                  <1>       ; 10/09/2015
35884 0000C53C 61                     <1>       popad ; edi, esi, ebp, temp (icrement esp by 4), ebx, edx, ecx, eax
35885                                  <1>              ; mov (sp)+,sc / restore user registers
35886                                  <1>              ; mov (sp)+,mq
35887                                  <1>              ; mov (sp)+,ac
35888                                  <1>              ; mov (sp)+,r5
35889                                  <1>              ; mov (sp)+,r4
35890                                  <1>              ; mov (sp)+,r3
35891                                  <1>              ; mov (sp)+,r2
35892                                  <1>       ;
35893 0000C53D A1[64030600]           <1>       mov    eax, [u.r0]  ; ((return value in EAX))
35894 0000C542 0FA9                   <1>       pop    gs
35895 0000C544 0FA1                   <1>       pop    fs
35896 0000C546 07                     <1>       pop    es
35897 0000C547 1F                     <1>       pop    ds
35898                                  <1>       ;or    word [esp+8], 200h ; 22/01/2017 ; force enabling interrupts
```

```
35899 0000C548 CF                  <1>       iretd
35900                              <1>             ; rti / no, return from interrupt
35901                              <1>
35902                              <1> sysrele:
35903                              <1>       ; 24/03/2017
35904                              <1>       ; 28/02/2017
35905                              <1>       ; 27/02/2017
35906                              <1>       ; 29/01/2017
35907                              <1>       ; 14/01/2017
35908                              <1>       ; 13/01/2017
35909                              <1>       ; 09/01/2017, 10/01/2017, 12/01/2017
35910                              <1>       ; Major modification for TRDOS 386 (CallBack return)
35911                              <1>       ;
35912                              <1>       ; 'sysrele' system call restores previously saved
35913                              <1>       ; registers and addresses of the process
35914                              <1>       ; (Main purpose -in TRDOS 386- is to return from
35915                              <1>       ; timer callback service routine in ring 3 -user mode-.)
35916                              <1>       ;
35917                              <1>       ; check if the process is in timer callback phase
35918 0000C549 803D[D4030600]00    <1>       cmp    byte [u.t_lock], 0 ; TIMER INT LOCK
35919                              <1>       ;je    short sysrel0 ; classic (Retro UNIX 386 type) sysrele
35920 0000C550 7734                <1>       ja     short sysrel3
35921                              <1>       ; 27/02/2017
35922 0000C552 803D[D8030600]00    <1>       cmp    byte [u.r_lock], 0 ; IRQ callback lock
35923 0000C559 0F8667FFFFFF        <1>       jna    sysrel0 ; classic sysrele ; 24/03/2017
35924 0000C55F E859000000          <1>       call   sysrel7
35925 0000C564 803D[D8030600]00    <1>       cmp    byte [u.r_lock], 0 ; IRQ callback service lock
35926 0000C56B 7628                <1>       jna    short sysrel4
35927 0000C56D C605[D8030600]00    <1>       mov    byte [u.r_lock], 0 ; reset
35928                              <1>       ;mov   byte [u.irqwait], 0 ; reset ; 28/02/2017
35929 0000C574 A0[D9030600]        <1>       mov    al, [u.r_mode]
35930 0000C579 08C0                <1>       or     al, al
35931 0000C57B 7518                <1>       jnz    short sysrel4
35932 0000C57D FEC8                <1>       dec    al
35933 0000C57F A2[D9030600]        <1>       mov    [u.r_mode], al ; 0FFh ; not necessary !?
35934 0000C584 EB32                <1>       jmp    short sysrel6
35935                              <1> sysrel3:
35936                              <1>       ; 27/02/2017
35937 0000C586 E832000000          <1>       call   sysrel7
35938                              <1>       ; 14/01/2017
35939 0000C58B 28C0                <1>       sub    al, al
35940 0000C58D 3805[D4030600]      <1>       cmp    [u.t_lock], al ; 0 ; TIMER INT LOCK
35941 0000C593 770E                <1>       ja     short sysrel5 ; yes
35942                              <1> sysrel4:
35943                              <1>       ; 29/01/2017
35944 0000C595 8B44241C            <1>       mov    eax, [esp+28] ; eax
35945 0000C599 A3[64030600]        <1>       mov    [u.r0], eax
35946 0000C59E E93EFFFFFF          <1>       jmp    sysrel2
35947                              <1> sysrel5:
35948 0000C5A3 A2[D4030600]        <1>       mov    [u.t_lock], al ; 0 ; reset
35949 0000C5A8 A0[D5030600]        <1>       mov    al, [u.t_mode]
35950 0000C5AD 20C0                <1>       and    al, al
35951                              <1>       ;jnz   short sysrel2 ; 0FFh ; user mode
35952 0000C5AF 75E4                <1>       jnz    short sysrel4 ; 29/01/2017
35953 0000C5B1 FEC8                <1>       dec    al
35954 0000C5B3 A2[D5030600]        <1>       mov    [u.t_mode], al ; 0FFh ; not necessary !?
35955                              <1> sysrel6:
35956                              <1>       ; cpu will continue from the interrupted sytem call addr
35957 0000C5B8 61                  <1>       popad         ; edi, esi, ebp, esp, ebx, edx, ecx, eax
35958 0000C5B9 83C410              <1>       add    esp, 16      ; pass segment segisters: ds, es, fs, gs
35959 0000C5BC CF                  <1>       iretd         ; eip, cs, eflags
35960                              <1>
35961                              <1> sysrel7:
35962 0000C5BD 0FB61D[B3030600]    <1>       movzx  ebx, byte [u.uno] ; current process number
35963 0000C5C4 66C1E302            <1>       shl    bx, 2
35964                              <1>       ;cmp   [ebx+p.tcb-4], eax ; 0 ; is there callback address ?
35965                              <1>       ;jna   short sysrel0
35966                              <1>       ; yes, reset callback address then restore process registers
35967                              <1>       ;mov   [ebx+p.tcb-4], eax ; 0 ; reset
35968 0000C5C8 8B83[BC000600]      <1>       mov     eax, [ebx+p.upage-4] ; UPAGE address
35969 0000C5CE FA                  <1>       cli    ; disable interrupts till 'iretd'
35970 0000C5CF E98B210000          <1>       jmp    rswap ; restore process 'u' structure
35971                              <1>
35972                              <1> badsys:
35973                              <1>       ; 25/12/2016
35974                              <1>       ; 18/04/2016 (TRDOS 386 = TRDOS v2.0)
35975                              <1>       ; 17/04/2011 (TRDOS v1.0, 'IFC.ASM')
35976                              <1>       ; 03/02/2011 ('trdos_ifc_routine')
35977                              <1>       ;
35978                              <1>       ; 16/04/2015 (Retro UNIX 386 v1, 'badsys')
35979                              <1>       ; (EIP, EAX values will be shown on screen with error message)
35980                              <1>       ; (EIP = 'CD 40h' instruction address -INT 40h-)
35981                              <1>       ; (EAX = Function number)
35982                              <1>       ;
35983 0000C5D4 FE05[B2030600]      <1>       inc    byte [u.bsys]
35984                              <1>       ;
35985 0000C5DA 8B1D[5C030600]      <1>       mov    ebx, [u.sp] ; esp at the beginning of 'sysent'
35986 0000C5E0 8B03                <1>       mov    eax, [ebx] ; EIP (return address, not 'INT 30h' address)
35987 0000C5E2 83E802              <1>       sub    eax, 2 ; CDh, ##h
35988 0000C5E5 E81F6DFFFF          <1>       call   dwordtohex
35989 0000C5EA 8915[A5090100]      <1>       mov    [eip_str], edx
35990 0000C5F0 A3[A9090100]        <1>       mov    [eip_str+4], eax
35991 0000C5F5 A1[64030600]        <1>       mov    eax, [u.r0]
35992 0000C5FA E80A6DFFFF          <1>       call   dwordtohex
35993 0000C5FF 8915[94090100]      <1>       mov    [eax_str], edx
35994 0000C605 A3[98090100]        <1>       mov    [eax_str+4], eax
35995                              <1>
35996 0000C60A 66C705[89090100]34- <1>       mov    word [int_num_str], SYSCALL_INT_NUM ; 25/12/2016
35997 0000C612 30                  <1>
35998                              <1>
35999 0000C613 BE[5B090100]        <1>       mov    esi, ifc_msg ; "invalid funtion call !" msg (trdosk9.s)
36000 0000C618 E8409DFFFF          <1>       call   print_msg
```

```
36001                                  <1>
36002 0000C61D EB17                    <1>       jmp    sysexit
36003                                  <1>
36004                                  <1> intract: ; / interrupt action
36005                                  <1>       ; 14/10/2015
36006                                  <1>       ; 16/04/2015 (Retro UNIX 386 v1 - Beginning)
36007                                  <1>       ; 09/05/2013 - 07/12/2013 (Retro UNIX 8086 v1)
36008                                  <1>       ;
36009                                  <1>       ; Retro UNIX 8086 v1 modification !
36010                                  <1>       ; (Process/task switching and quit routine by using
36011                                  <1>       ; Retro UNIX 8086 v1 keyboard interrupt output.))
36012                                  <1>       ;
36013                                  <1>       ; input -> 'u.quit'  (also value of 'u.intr' > 0)
36014                                  <1>       ; output -> If value of 'u.quit' = FFFFh ('ctrl+brk' sign)
36015                                  <1>       ;            'intract' will jump to 'sysexit'.
36016                                  <1>       ;          Intract will return to the caller
36017                                  <1>       ;            if value of 'u.quit' <> FFFFh.
36018                                  <1>       ; 14/10/2015
36019 0000C61F FB                      <1>       sti
36020                                  <1>       ; 07/12/2013
36021 0000C620 66FF05[AC030600]        <1>       inc    word [u.quit]
36022 0000C627 7408                    <1>       jz     short intrct0 ; FFFFh -> 0
36023 0000C629 66FF0D[AC030600]        <1>       dec    word [u.quit]
36024                                  <1>       ; 16/04/2015
36025 0000C630 C3                      <1>       retn
36026                                  <1> intrct0:
36027 0000C631 58                      <1>       pop    eax ; call intract -> retn
36028                                  <1>       ;
36029 0000C632 31C0                    <1>       xor    eax, eax
36030 0000C634 FEC0                    <1>       inc    al  ; mov ax, 1
36031                                  <1> ;;;
36032                                  <1>       ; UNIX v1 original 'intract' routine...
36033                                  <1>       ; / interrupt action
36034                                  <1>             ;cmp *(sp),$rti / are you in a clock interrupt?
36035                                  <1>             ; bne 1f / no, 1f
36036                                  <1>             ; cmp (sp)+,(sp)+ / pop clock pointer
36037                                  <1>       ; 1: / now in user area
36038                                  <1>             ; mov r1,-(sp) / save r1
36039                                  <1>             ; mov u.ttyp,r1
36040                                  <1>                   ; / pointer to tty buffer in control-to r1
36041                                  <1>             ; cmpb 6(r1),$177
36042                                  <1>                   ; / is the interrupt char equal to "del"
36043                                  <1>             ; beq 1f / yes, 1f
36044                                  <1>             ; clrb 6(r1)
36045                                  <1>                    ; / no, clear the byte
36046                                  <1>                   ; / (must be a quit character)
36047                                  <1>             ; mov (sp)+,r1 / restore r1
36048                                  <1>             ; clr u.quit / clear quit flag
36049                                  <1>             ; bis $20,2(sp)
36050                                  <1>                   ; / set trace for quit (sets t bit of
36051                                  <1>                   ; / ps-trace trap)
36052                                  <1>             ; rti   ;  / return from interrupt
36053                                  <1>       ; 1: / interrupt char = del
36054                                  <1>             ; clrb 6(r1) / clear the interrupt byte
36055                                  <1>                     ; / in the buffer
36056                                  <1>             ; mov (sp)+,r1 / restore r1
36057                                  <1>             ; cmp u.intr,$core / should control be
36058                                  <1>                         ; / transferred to loc core?
36059                                  <1>             ; blo 1f
36060                                  <1>             ; jmp *u.intr / user to do rti yes,
36061                                  <1>                         ; / transfer to loc core
36062                                  <1>       ; 1:
36063                                  <1>             ; sys 1 / exit
36064                                  <1>
36065                                  <1> sysexit: ; <terminate process>
36066                                  <1>       ; 27/05/2017
36067                                  <1>       ; 10/04/2017
36068                                  <1>       ; 28/02/2017
36069                                  <1>       ; 26/02/2017
36070                                  <1>       ; 02/01/2017, 23/01/2017
36071                                  <1>       ; 06/06/2016, 10/06/2016
36072                                  <1>       ; 19/05/2016, 23/05/2016
36073                                  <1>       ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
36074                                  <1>       ; 16/04/2015 - 01/09/2015 (Retro UNIX 386 v1)
36075                                  <1>       ; 19/04/2013 - 14/02/2014 (Retro UNIX 8086 v1)
36076                                  <1>       ;
36077                                  <1>       ; 'sysexit' terminates a process. First each file that
36078                                  <1>       ; the process has opened is closed by 'flose'. The process
36079                                  <1>       ; status is then set to unused. The 'p.pid' table is then
36080                                  <1>       ; searched to find children of the dying process. If any of
36081                                  <1>       ; children are zombies (died by not waited for), they are
36082                                  <1>       ; set free. The 'p.pid' table is then searched to find the
36083                                  <1>       ; dying process's parent. When the parent is found, it is
36084                                  <1>       ; checked to see if it is free or it is a zombie. If it is
36085                                  <1>       ; one of these, the dying process just dies. If it is waiting
36086                                  <1>       ; for a child process to die, it notified that it doesn't
36087                                  <1>       ; have to wait anymore by setting it's status from 2 to 1
36088                                  <1>       ; (waiting to active). It is awakened and put on runq by
36089                                  <1>       ; 'putlu'. The dying process enters a zombie state in which
36090                                  <1>       ; it will never be run again but stays around until a 'wait'
36091                                  <1>       ; is completed by it's parent process. If the parent is not
36092                                  <1>       ; found, process just dies. This means 'swap' is called with
36093                                  <1>       ; 'u.uno=0'. What this does is the 'wswap' is not called
36094                                  <1>       ; to write out the process and 'rswap' reads the new process
36095                                  <1>       ; over the one that dies..i.e., the dying process is
36096                                  <1>       ; overwritten and destroyed.
36097                                  <1>       ;
36098                                  <1>       ; Calling sequence:
36099                                  <1>       ;     sysexit or conditional branch.
36100                                  <1>       ; Arguments:
36101                                  <1>       ;     -
36102                                  <1>       ; ..........................................................
```

```
36103                                  <1>       ;
36104                                  <1>       ; Retro UNIX 8086 v1 modification:
36105                                  <1>       ;       System call number (=1) is in EAX register.
36106                                  <1>       ;
36107                                  <1>       ;       Other parameters are in EDX, EBX, ECX, ESI, EDI, EBP
36108                                  <1>       ;       registers depending of function details.
36109                                  <1>       ;
36110                                  <1>       ; ('swap' procedure is mostly different than original UNIX v1.)
36111                                  <1>       ;
36112                                  <1> ; / terminate process
36113                                  <1>       ; AX = 1
36114 0000C636 6648                    <1>       dec    ax ; 0
36115 0000C638 66A3[AA030600]          <1>       mov    [u.intr], ax ; 0
36116                                  <1>              ; clr u.intr / clear interrupt control word
36117                                  <1>              ; clr r1 / clear r1
36118                                  <1> sysexit_0:
36119                                  <1>       ; 23/01/2017
36120                                  <1>       ; 02/01/2017
36121                                  <1>       ; 10/06/2016
36122                                  <1>       ; 06/06/2016
36123                                  <1>       ; 23/05/2016
36124                                  <1>       ; 19/05/2016 - TRDOS 386 (TRDOS v2.0)
36125                                  <1>       ; Check and stop/clear timer event(s) of this (dying) process
36126                                  <1>       ; if there is.
36127                                  <1>
36128                                  <1>       ; 02/01/2017
36129 0000C63E FA                      <1>       cli    ; disable interrupts
36130                                  <1>       ; 23/01/2017 - reset timer frequency (to 18.2Hz)
36131 0000C63F B036                    <1>       mov    al, 00110110b ; 36h
36132 0000C641 E643                    <1>       out    43h, al
36133 0000C643 28C0                    <1>       sub    al, al ; 0
36134 0000C645 E640                    <1>       out    40h, al ; LB
36135 0000C647 E640                    <1>       out    40h, al ; HB
36136                                  <1>       ;
36137 0000C649 0FB61D[B3030600]        <1>       movzx  ebx, byte [u.uno]
36138                                  <1>       ;mov   bl, [u.uno] ; process number of dying process
36139 0000C650 3883[FF000600]          <1>       cmp    byte [ebx+p.timer-1], al ; 0
36140 0000C656 763B                    <1>       jna    short sysexit_12 ; no timer events for this process
36141 0000C658 8883[FF000600]          <1>       mov    byte [ebx+p.timer-1], al ; 0 ; reset
36142 0000C65E A0[5F5B0100]            <1>       mov    al, [timer_events]
36143                                  <1>       ;or    al, al
36144                                  <1>       ;jz    short sysexit_12 ; no timer events
36145 0000C663 88C1                    <1>       mov    cl, al
36146                                  <1>       ;cli   ; disable interrupts
36147 0000C665 B410                    <1>       mov    ah, 16 ; number of available timer events
36148 0000C667 BE[60040600]            <1>       mov    esi, timer_set ; beginning address of timer events
36149                                  <1> sysexit_7:
36150 0000C66C 8A06                    <1>       mov    al, [esi] ; process number (of timer event)
36151 0000C66E 38D8                    <1>       cmp    al, bl ; process number comparison
36152 0000C670 7411                    <1>       je     short sysexit_10
36153 0000C672 20C0                    <1>       and    al, al
36154 0000C674 7404                    <1>       jz     short sysexit_9
36155                                  <1> sysexit_8:
36156 0000C676 FEC9                    <1>       dec    cl
36157 0000C678 7416                    <1>       jz     short sysexit_11
36158                                  <1> sysexit_9:
36159 0000C67A FECC                    <1>       dec    ah
36160 0000C67C 7415                    <1>       jz     short sysexit_12
36161 0000C67E 83C610                  <1>       add    esi, 16
36162 0000C681 EBE9                    <1>       jmp    short sysexit_7
36163                                  <1>
36164                                  <1> sysexit_10:
36165                                  <1>       ;mov   byte [esi], 0
36166 0000C683 66C7060000              <1>       mov    word [esi], 0
36167                                  <1>       ;mov   dword [esi+12], 0
36168                                  <1>       ;
36169 0000C688 FE0D[5F5B0100]          <1>       dec    byte [timer_events] ; 02/01/2017
36170                                  <1>       ;
36171 0000C68E EBE6                    <1>       jmp    short sysexit_8
36172                                  <1>
36173                                  <1> sysexit_11:
36174 0000C690 6629C0                  <1>       sub    ax, ax ; 0 ; 26/02/2017
36175                                  <1> sysexit_12:
36176                                  <1>       ; 26/02/2017 (Unlink IRQ callbacks belong to the user)
36177 0000C693 803D[D6030600]00        <1>       cmp    byte [u.irqc], 0 ; Count of IRQ callbacks
36178 0000C69A 7E2E                    <1>       jng    short sysexit_16 ; zero or invalid
36179                                  <1>       ; 28/02/2017
36180                                  <1>       ; clear IRQ callback flags (for 'sysrele' and 'sysret')
36181 0000C69C A2[D7030600]            <1>       mov    [u.irqwait], al ; 0 ; force to clear waiting flag
36182 0000C6A1 A2[D8030600]            <1>       mov    [u.r_lock], al ; 0 ; force to clear busy flag
36183 0000C6A6 BE[FE600100]            <1>       mov    esi, IRQ.owner
36184                                  <1> sysexit_13:
36185 0000C6AB AC                      <1>       lodsb
36186 0000C6AC 3A05[B3030600]          <1>       cmp    al, [u.uno] ; owner  = current user ?
36187 0000C6B2 750C                    <1>       jne    short sysexit_14
36188 0000C6B4 C646FF00                <1>       mov    byte [esi-1], 0 ; owner = 0 : Free
36189 0000C6B8 FE0D[D6030600]          <1>       dec    byte [u.irqc]
36190 0000C6BE 7408                    <1>       jz     short sysexit_15
36191                                  <1> sysexit_14:
36192 0000C6C0 81FE[06610100]          <1>       cmp    esi, IRQ.owner + 8 ; the last IRQ index number ?
36193 0000C6C6 76E3                    <1>       jna    short sysexit_13 ; no
36194                                  <1> sysexit_15:
36195 0000C6C8 30C0                    <1>       xor    al, al ; 0
36196                                  <1> sysexit_16: ; 2:
36197 0000C6CA FB                      <1>       sti    ; enable interrupts
36198                                  <1>       ;
36199                                  <1>       ; AX = 0
36200                                  <1> sysexit_1: ; 1:
36201                                  <1>       ; AX = File descriptor
36202                                  <1>              ; / r1 has file descriptor (index to u.fp list)
36203                                  <1>              ; / Search the whole list
36204 0000C6CB E8E5130000              <1>       call   fclose
```

```
36205                                  <1>             ; jsr r0,fclose / close all files the process opened
36206                                  <1>         ;; ignore error return
36207                                  <1>             ; br .+2 / ignore error return
36208                                  <1>         ;inc   ax
36209 0000C6D0 FEC0                   <1>         inc    al
36210                                  <1>             ; inc r1 / increment file descriptor
36211                                  <1>         ;cmp   ax, 10
36212 0000C6D2 3C0A                   <1>         cmp    al, 10
36213                                  <1>             ; cmp r1,$10. / end of u.fp list?
36214 0000C6D4 72F5                   <1>         jb     short sysexit_1
36215                                  <1>             ; blt 1b / no, go back
36216                                  <1>         ;movzx ebx, byte [u.uno]
36217 0000C6D6 8A1D[B3030600]         <1>         mov    bl, [u.uno] ; 02/01/2017
36218                                  <1>             ; movb u.uno,r1 / yes, move dying process's number to r1
36219 0000C6DC 88A3[AF000600]         <1>         mov    [ebx+p.stat-1], ah ; 0, SFREE
36220                                  <1>             ; clrb p.stat-1(r1) / free the process
36221                                  <1>         ; 10/04/2017
36222 0000C6E2 381D[75610100]         <1>         cmp    [audio_user], bl
36223 0000C6E8 7518                   <1>         jne    short sysexit_17
36224                                  <1>         ; reset audio device (current) owner and 'initializated' flag
36225 0000C6EA 883D[75610100]         <1>         mov    [audio_user], bh ; 0
36226                                  <1>         ; 27/05/2017
36227 0000C6F0 8B0D[60610100]         <1>         mov    ecx,  [audio_buffer]
36228 0000C6F6 09C9                   <1>         or     ecx, ecx
36229 0000C6F8 7408                   <1>         jz     short sysexit_17
36230                                  <1>         ; 'deallocate_user_pages' is not necessary in sysexit !!!
36231                                  <1>         ;push  ebx
36232                                  <1>         ;mov   ebx, ecx
36233                                  <1>         ;mov   ecx, [audio_buff_size]
36234                                  <1>         ;call  deallocate_user_pages
36235                                  <1>         ;; (Modified Registers -> EAX, EDX, ESI, EDI, EBX, ECX, EBP)
36236 0000C6FA 29C9                   <1>         sub    ecx, ecx
36237 0000C6FC 890D[60610100]         <1>         mov    [audio_buffer], ecx ; 0
36238                                  <1>         ;pop   ebx
36239                                  <1> sysexit_17:
36240                                  <1>         ;shl   bx, 1
36241 0000C702 D0E3                   <1>         shl    bl, 1
36242                                  <1>             ; asl r1 / use r1 for index into the below tables
36243 0000C704 668B8B[1E000600]       <1>         mov    cx, [ebx+p.pid-2]
36244                                  <1>             ; mov p.pid-2(r1),r3 / move dying process's name to r3
36245 0000C70B 668B93[3E000600]       <1>         mov    dx, [ebx+p.ppid-2]
36246                                  <1>             ; mov p.ppid-2(r1),r4 / move its parents name to r4
36247                                  <1>         ; xor  bx, bx ; 0
36248 0000C712 30DB                   <1>         xor    bl, bl ; 0
36249                                  <1>             ; clr r2
36250 0000C714 31F6                   <1>         xor    esi, esi ; 0
36251                                  <1>             ; clr r5 / initialize reg
36252                                  <1> sysexit_2: ; 1:
36253                                  <1>              ; / find children of this dying process,
36254                                  <1>             ; / if they are zombies, free them
36255                                  <1>         ;add   bx, 2
36256 0000C716 80C302                 <1>         add    bl, 2
36257                                  <1>             ; add $2,r2 / search parent process table
36258                                  <1>                       ; / for dying process's name
36259 0000C719 66398B[3E000600]       <1>         cmp    [ebx+p.ppid-2], cx
36260                                  <1>             ; cmp p.ppid-2(r2),r3 / found it?
36261 0000C720 7513                   <1>         jne    short sysexit_4
36262                                  <1>             ; bne 3f / no
36263                                  <1>         ;shr   bx, 1
36264 0000C722 D0EB                   <1>         shr    bl, 1
36265                                  <1>             ; asr r2 / yes, it is a parent
36266 0000C724 80BB[AF000600]03       <1>         cmp    byte [ebx+p.stat-1], 3 ; SZOMB
36267                                  <1>             ; cmpb p.stat-1(r2),$3 / is the child of this
36268                                  <1>                                    ; / dying process a zombie
36269 0000C72B 7506                   <1>         jne    short sysexit_3
36270                                  <1>             ; bne 2f / no
36271 0000C72D 88A3[AF000600]         <1>         mov    [ebx+p.stat-1], ah ; 0, SFREE
36272                                  <1>             ; clrb p.stat-1(r2) / yes, free the child process
36273                                  <1> sysexit_3: ; 2:
36274                                  <1>         ;shr   bx, 1
36275 0000C733 D0E3                   <1>         shl    bl, 1
36276                                  <1>             ; asl r2
36277                                  <1> sysexit_4: ; 3:
36278                                  <1>             ; / search the process name table
36279                                  <1>             ; / for the dying process's parent
36280 0000C735 663993[1E000600]       <1>         cmp    [ebx+p.pid-2], dx
36281                                  <1>             ; cmp p.pid-2(r2),r4 / found it?
36282 0000C73C 7502                   <1>         jne    short sysexit_5
36283                                  <1>             ; bne 3f / no
36284 0000C73E 89DE                   <1>         mov    esi, ebx
36285                                  <1>             ; mov r2,r5 / yes, put index to p.pid table (parents
36286                                  <1>                       ; / process # x2) in r5
36287                                  <1> sysexit_5: ; 3:
36288                                  <1>         ;cmp   bx, nproc + nproc
36289 0000C740 80FB20                 <1>         cmp    bl, nproc + nproc
36290                                  <1>             ; cmp r2,$nproc+nproc / has whole table been searched?
36291 0000C743 72D1                   <1>         jb     short sysexit_2
36292                                  <1>             ; blt 1b / no, go back
36293                                  <1>             ; mov r5,r1 / yes, r1 now has parents process # x2
36294 0000C745 21F6                   <1>         and    esi, esi ; r5=r1
36295 0000C747 7436                   <1>         jz     short sysexit_6
36296                                  <1>             ; beq 2f / no parent has been found.
36297                                  <1>                    ; / The process just dies
36298 0000C749 66D1EE                 <1>         shr    si, 1
36299                                  <1>             ; asr r1 / set up index to p.stat
36300 0000C74C 8A86[AF000600]         <1>         mov    al, [esi+p.stat-1]
36301                                  <1>             ; movb p.stat-1(r1),r2 / move status of parent to r2
36302 0000C752 20C0                   <1>         and    al, al
36303 0000C754 7429                   <1>         jz     short sysexit_6
36304                                  <1>             ; beq 2f / if its been freed, 2f
36305 0000C756 3C03                   <1>         cmp    al, 3
36306                                  <1>             ; cmp r2,$3 / is parent a zombie?
```

```
36307 0000C758 7425                <1>       je    short sysexit_6
36308                              <1>             ; beq 2f / yes, 2f
36309                              <1>       ; BH = 0
36310 0000C75A 8A1D[B3030600]      <1>       mov   bl, [u.uno]
36311                              <1>             ; movb u.uno,r3 / move dying process's number to r3
36312 0000C760 C683[AF000600]03    <1>       mov   byte [ebx+p.stat-1], 3  ; SZOMB
36313                              <1>             ; movb $3,p.stat-1(r3) / make the process a zombie
36314 0000C767 3C01                <1>       cmp   al, 1 ; SRUN
36315 0000C769 7414                <1>       je    short sysexit_6
36316                              <1>       ;cmp  al, 2
36317                              <1>             ; cmp r2,$2 / is the parent waiting for
36318                              <1>                     ; / this child to die
36319                              <1>       ;jne  short sysexit_6
36320                              <1>             ; bne 2f / yes, notify parent not to wait any more
36321                              <1>       ; p.stat = 2 --> waiting
36322                              <1>       ; p.stat = 4 --> sleeping
36323 0000C76B C686[AF000600]01    <1>       mov   byte [esi+p.stat-1], 1 ; SRUN
36324                              <1>       ;dec  byte [esi+p.stat-1]
36325                              <1>             ; decb p.stat-1(r1) / awaken it by putting it (parent)
36326 0000C772 6689F0              <1>       mov   ax, si ; r1  (process number in AL)
36327                              <1>       ;
36328                              <1>       ;mov  ebx, runq + 4
36329                              <1>             ; mov $runq+4,r2 / on the runq
36330 0000C775 BB[54030600]        <1>       mov   ebx, runq+2 ; normal run queue ; 02/01/2017
36331 0000C77A E818200000          <1>       call  putlu
36332                              <1>             ; jsr r0, putlu
36333                              <1> sysexit_6:
36334                              <1>             ; / the process dies
36335 0000C77F C605[B3030600]00    <1>       mov   byte [u.uno], 0
36336                              <1>             ; clrb u.uno / put zero as the process number,
36337                              <1>                 ; / so "swap" will
36338 0000C786 E80E1F0000          <1>       call  swap
36339                              <1>             ; jsr r0,swap / overwrite process with another process
36340                              <1>
36341                              <1> hlt_sys:
36342                              <1>       ;sti
36343                              <1> hlts0:
36344 0000C78B F4                  <1>       hlt
36345 0000C78C EBFD                <1>       jmp   short hlts0
36346                              <1>             ; 0 / and thereby kill it; halt?
36347                              <1>
36348                              <1> syswait: ; < wait for a processs to die >
36349                              <1>       ; 17/09/2015
36350                              <1>       ; 02/09/2015
36351                              <1>       ; 01/09/2015
36352                              <1>       ; 16/04/2015 (Retro UNIX 386 v1 - Beginning)
36353                              <1>       ; 24/05/2013 - 05/02/2014 (Retro UNIX 8086 v1)
36354                              <1>       ;
36355                              <1>       ; 'syswait' waits for a process die.
36356                              <1>       ; It works in following way:
36357                              <1>       ;    1) From the parent process number, the parent's
36358                              <1>       ;      process name is found. The p.ppid table of parent
36359                              <1>       ;      names is then searched for this process name.
36360                              <1>       ;      If a match occurs, r2 contains child's process
36361                              <1>       ;      number. The child status is checked to see if it is
36362                              <1>       ;      a zombie, i.e; dead but not waited for (p.stat=3)
36363                              <1>       ;      If it is, the child process is freed and it's name
36364                              <1>       ;      is put in (u.r0). A return is then made via 'sysret'.
36365                              <1>       ;      If the child is not a zombie, nothing happens and
36366                              <1>       ;      the search goes on through the p.ppid table until
36367                              <1>       ;      all processes are checked or a zombie is found.
36368                              <1>       ;    2) If no zombies are found, a check is made to see if
36369                              <1>       ;      there are any children at all. If there are none,
36370                              <1>       ;      an error return is made. If there are, the parent's
36371                              <1>       ;      status is set to 2 (waiting for child to die),
36372                              <1>       ;      the parent is swapped out, and a branch to 'syswait'
36373                              <1>       ;      is made to wait on the next process.
36374                              <1>       ;
36375                              <1>       ; Calling sequence:
36376                              <1>       ;     ?
36377                              <1>       ; Arguments:
36378                              <1>       ;     -
36379                              <1>       ; Inputs: -
36380                              <1>       ; Outputs: if zombie found, it's name put in u.r0.
36381                              <1>       ; ..........................................................
36382                              <1>       ;
36383                              <1>
36384                              <1> ; / wait for a process to die
36385                              <1>
36386                              <1> syswait_0:
36387 0000C78E 0FB61D[B3030600]    <1>       movzx  ebx, byte [u.uno] ; 01/09/2015
36388                              <1>              ; movb u.uno,r1 / put parents process number in r1
36389 0000C795 D0E3                <1>       shl   bl, 1
36390                              <1>       ;shl  bx, 1
36391                              <1>             ; asl r1 / x2 to get index into p.pid table
36392 0000C797 668B83[1E000600]    <1>       mov   ax, [ebx+p.pid-2]
36393                              <1>             ; mov p.pid-2(r1),r1 / get the name of this process
36394 0000C79E 31F6                <1>       xor   esi, esi
36395                              <1>             ; clr r2
36396 0000C7A0 31C9                <1>       xor   ecx, ecx ; 30/10/2013
36397                              <1>       ;xor  cl, cl
36398                              <1>             ; clr r3 / initialize reg 3
36399                              <1> syswait_1: ; 1:
36400 0000C7A2 6683C602            <1>       add   si, 2
36401                              <1>             ; add $2,r2 / use r2 for index into p.ppid table
36402                              <1>                     ; / search table of parent processes
36403                              <1>                     ; / for this process name
36404 0000C7A6 663B86[3E000600]    <1>       cmp   ax, [esi+p.ppid-2]
36405                              <1>             ; cmp p.ppid-2(r2),r1 / r2 will contain the childs
36406                              <1>                                  ; / process number
36407 0000C7AD 7535                <1>       jne   short syswait_3
36408                              <1>             ;bne 3f / branch if no match of parent process name
```

```
36409                                  <1>       ;inc   cx
36410 0000C7AF FEC1                    <1>       inc    cl
36411                                  <1>              ;inc r3 / yes, a match, r3 indicates number of children
36412 0000C7B1 66D1EE                  <1>       shr    si, 1
36413                                  <1>              ; asr r2 / r2/2 to get index to p.stat table
36414                                  <1>       ; The possible states ('p.stat' values) of a process are:
36415                                  <1>       ;      0 = free or unused
36416                                  <1>       ;      1 = active
36417                                  <1>       ;      2 = waiting for a child process to die
36418                                  <1>       ;      3 = terminated, but not yet waited for (zombie).
36419 0000C7B4 80BE[AF000600]03        <1>       cmp    byte [esi+p.stat-1], 3 ; SZOMB, 05/02/2014
36420                                  <1>              ; cmpb p.stat-1(r2),$3 / is the child process a zombie?
36421 0000C7BB 7524                    <1>       jne    short syswait_2
36422                                  <1>              ; bne 2f / no, skip it
36423 0000C7BD 88BE[AF000600]          <1>       mov    [esi+p.stat-1], bh ; 0
36424                                  <1>              ; clrb p.stat-1(r2) / yes, free it
36425 0000C7C3 66D1E6                  <1>       shl    si, 1
36426                                  <1>              ; asl r2 / r2x2 to get index into p.pid table
36427 0000C7C6 0FB786[1E000600]        <1>       movzx  eax, word [esi+p.pid-2]
36428 0000C7CD A3[64030600]            <1>       mov    [u.r0], eax
36429                                  <1>              ; mov p.pid-2(r2),*u.r0
36430                                  <1>                           ; / put childs process name in (u.r0)
36431                                  <1>       ;
36432                                  <1>       ; Retro UNIX 386 v1 modification ! (17/09/2015)
36433                                  <1>       ;
36434                                  <1>       ; Parent process ID -p.ppid- field (of the child process)
36435                                  <1>       ; must be cleared in order to prevent infinitive 'syswait'
36436                                  <1>       ; system call loop from the application/program if it calls
36437                                  <1>       ; 'syswait' again (mistakenly) while there is not a zombie
36438                                  <1>       ; or running child process to wait. ('forktest.s', 17/09/2015)
36439                                  <1>       ;
36440                                  <1>       ; Note: syswait will return with error if there is not a
36441                                  <1>       ;       zombie or running process to wait.
36442                                  <1>       ;
36443 0000C7D2 6629C0                  <1>       sub    ax, ax
36444 0000C7D5 668986[3E000600]        <1>       mov    [esi+p.ppid-2], ax ; 0 ; 17/09/2015
36445 0000C7DC E9D0FCFFFF              <1>       jmp    sysret0 ; ax = 0
36446                                  <1>       ;
36447                                  <1>       ;jmp   sysret
36448                                  <1>              ; br sysret1 / return cause child is dead
36449                                  <1> syswait_2: ; 2:
36450 0000C7E1 66D1E6                  <1>       shl    si, 1
36451                                  <1>              ; asl r2 / r2x2 to get index into p.ppid table
36452                                  <1> syswait_3: ; 3:
36453 0000C7E4 6683FE20                <1>       cmp    si, nproc+nproc
36454                                  <1>              ; cmp r2,$nproc+nproc / have all processes been checked?
36455 0000C7E8 72B8                    <1>       jb     short syswait_1
36456                                  <1>              ; blt 1b / no, continue search
36457                                  <1>       ;and   cx, cx
36458 0000C7EA 20C9                    <1>       and    cl, cl
36459                                  <1>              ; tst r3 / one gets here if there are no children
36460                                  <1>                     ; / or children that are still active
36461                                  <1>       ; 30/10/2013
36462 0000C7EC 750B                    <1>       jnz    short syswait_4
36463                                  <1>       ;jz    error
36464                                  <1>              ; beq error1 / there are no children, error
36465 0000C7EE 890D[64030600]          <1>       mov    [u.r0], ecx ; 0
36466 0000C7F4 E996FCFFFF              <1>       jmp    error
36467                                  <1> syswait_4:
36468 0000C7F9 8A1D[B3030600]          <1>       mov    bl, [u.uno]
36469                                  <1>              ; movb u.uno,r1 / there are children so put
36470                                  <1>                           ; / parent process number in r1
36471 0000C7FF FE83[AF000600]          <1>       inc    byte [ebx+p.stat-1] ; 2, SWAIT, 05/02/2014
36472                                  <1>              ; incb p.stat-1(r1) / it is waiting for
36473                                  <1>                             ; / other children to die
36474                                  <1>       ; 04/11/2013
36475 0000C805 E88F1E0000              <1>       call   swap
36476                                  <1>              ; jsr r0,swap / swap it out, because it's waiting
36477 0000C80A EB82                    <1>       jmp    syswait_0
36478                                  <1>              ; br syswait / wait on next process
36479                                  <1>
36480                                  <1> sysfork: ; < create a new process >
36481                                  <1>       ; 02/01/2017 (TRDOS 386 modification)
36482                                  <1>       ; 04/09/2015, 18/05/2015
36483                                  <1>       ; 28/08/2015, 01/09/2015, 02/09/2015
36484                                  <1>       ; 09/05/2015, 10/05/2015, 14/05/2015
36485                                  <1>       ; 06/05/2015 (Retro UNIX 386 v1 - Beginning)
36486                                  <1>       ; 24/05/2013 - 14/02/2014 (Retro UNIX 8086 v1)
36487                                  <1>       ;
36488                                  <1>       ; 'sysfork' creates a new process. This process is referred
36489                                  <1>       ; to as the child process. This new process core image is
36490                                  <1>       ; a copy of that of the caller of 'sysfork'. The only
36491                                  <1>       ; distinction is the return location and the fact that (u.r0)
36492                                  <1>       ; in the old process (parent) contains the process id (p.pid)
36493                                  <1>       ; of the new process (child). This id is used by 'syswait'.
36494                                  <1>       ; 'sysfork' works in the following manner:
36495                                  <1>       ;    1) The process status table (p.stat) is searched to find
36496                                  <1>       ;      a process number that is unused. If none are found
36497                                  <1>       ;      an error occurs.
36498                                  <1>       ;    2) when one is found, it becomes the child process number
36499                                  <1>       ;      and it's status (p.stat) is set to active.
36500                                  <1>       ;    3) If the parent had a control tty, the interrupt
36501                                  <1>       ;      character in that tty buffer is cleared.
36502                                  <1>       ;    4) The child process is put on the lowest priority run
36503                                  <1>       ;      queue via 'putlu'.
36504                                  <1>       ;    5) A new process name is gotten from 'mpid' (actually
36505                                  <1>       ;      it is a unique number) and is put in the child's unique
36506                                  <1>       ;      identifier; process id (p.pid).
36507                                  <1>       ;    6) The process name of the parent is then obtained and
36508                                  <1>       ;      placed in the unique identifier of the parent process
36509                                  <1>       ;      name is then put in 'u.r0'.
36510                                  <1>       ;    7) The child process is then written out on disk by
```

```
36511                                  <1>       ;     'wswap',i.e., the parent process is copied onto disk
36512                                  <1>       ;     and the child is born. (The child process is written
36513                                  <1>       ;     out on disk/drum with 'u.uno' being the child process
36514                                  <1>       ;     number.)
36515                                  <1>       ;    8) The parent process number is then restored to 'u.uno'.
36516                                  <1>       ;    9) The child process name is put in 'u.r0'.
36517                                  <1>       ;   10) The pc on the stack sp + 18 is incremented by 2 to
36518                                  <1>       ;     create the return address for the parent process.
36519                                  <1>       ;   11) The 'u.fp' list as then searched to see what files
36520                                  <1>       ;     the parent has opened. For each file the parent has
36521                                  <1>       ;     opened, the corresponding 'fsp' entry must be updated
36522                                  <1>       ;     to indicate that the child process also has opened
36523                                  <1>       ;     the file. A branch to 'sysret' is then made.
36524                                  <1>       ;
36525                                  <1>       ; Calling sequence:
36526                                  <1>       ;     from shell ?
36527                                  <1>       ; Arguments:
36528                                  <1>       ;     -
36529                                  <1>       ; Inputs: -
36530                                  <1>       ; Outputs: *u.r0 - child process name
36531                                  <1>       ; ............................................................
36532                                  <1>       ;
36533                                  <1>       ; Retro UNIX 8086 v1 modification:
36534                                  <1>       ;     AX = r0 = PID (>0) (at the return of 'sysfork')
36535                                  <1>       ;     = process id of child a parent process returns
36536                                  <1>       ;     = process id of parent when a child process returns
36537                                  <1>       ;
36538                                  <1>       ;       In original UNIX v1, sysfork is called and returns as
36539                                  <1>       ;     in following manner: (with an example: c library, fork)
36540                                  <1>       ;
36541                                  <1>       ;     1:
36542                                  <1>       ;             sys    fork
36543                                  <1>       ;                    br 1f  / child process returns here
36544                                  <1>       ;             bes    2f     / parent process returns here
36545                                  <1>       ;             / pid of new process in r0
36546                                  <1>       ;             rts    pc
36547                                  <1>       ;     2: / parent process condionally branches here
36548                                  <1>       ;             mov    $-1,r0 / pid = -1 means error return
36549                                  <1>       ;             rts    pc
36550                                  <1>       ;
36551                                  <1>       ;     1: / child process brances here
36552                                  <1>       ;             clr    r0   / pid = 0 in child process
36553                                  <1>       ;             rts    pc
36554                                  <1>       ;
36555                                  <1>       ;     In UNIX v7x86 (386) by Robert Nordier (1999)
36556                                  <1>       ;             // pid = fork();
36557                                  <1>       ;             //
36558                                  <1>       ;             // pid == 0 in child process;
36559                                  <1>       ;             // pid == -1 means error return
36560                                  <1>       ;             // in child,
36561                                  <1>       ;             //    parents id is in par_uid if needed
36562                                  <1>       ;
36563                                  <1>       ;             _fork:
36564                                  <1>       ;                    mov    $.fork,eax
36565                                  <1>       ;                    int    $0x30
36566                                  <1>       ;                    jmp    1f
36567                                  <1>       ;                    jnc    2f
36568                                  <1>       ;                    jmp    cerror
36569                                  <1>       ;             1:
36570                                  <1>       ;                    mov    eax,_par_uid
36571                                  <1>       ;                    xor    eax,eax
36572                                  <1>       ;             2:
36573                                  <1>       ;                    ret
36574                                  <1>       ;
36575                                  <1>       ;     In Retro UNIX 8086 v1,
36576                                  <1>       ;     'sysfork' returns in following manner:
36577                                  <1>       ;
36578                                  <1>       ;             mov    ax, sys_fork
36579                                  <1>       ;             mov    bx, offset @f ; routine for child
36580                                  <1>       ;             int    20h
36581                                  <1>       ;             jc     error
36582                                  <1>       ;
36583                                  <1>       ;     ; Routine for parent process here (just after 'jc')
36584                                  <1>       ;             mov    word ptr [pid_of_child], ax
36585                                  <1>       ;             jmp    next_routine_for_parent
36586                                  <1>       ;
36587                                  <1>       ;     @@: ; routine for child process here
36588                                  <1>       ;             ....
36589                                  <1>       ;     NOTE: 'sysfork' returns to specified offset
36590                                  <1>       ;            for child process by using BX input.
36591                                  <1>       ;           (at first, parent process will return then
36592                                  <1>       ;           child process will return -after swapped in-
36593                                  <1>       ;           'syswait' is needed in parent process
36594                                  <1>       ;           if return from child process will be waited for.)
36595                                  <1>       ;
36596                                  <1>
36597                                  <1> ; / create a new process
36598                                  <1>       ; EBX = return address for child process
36599                                  <1>           ; (Retro UNIX 8086 v1 modification !)
36600 0000C80C 31F6                   <1>       xor    esi, esi
36601                                  <1>             ; clr r1
36602                                  <1> sysfork_1: ; 1: / search p.stat table for unused process number
36603 0000C80E 46                     <1>       inc    esi
36604                                  <1>             ; inc r1
36605 0000C80F 80BE[AF000600]00       <1>       cmp    byte [esi+p.stat-1], 0 ; SFREE, 05/02/2014
36606                                  <1>             ; tstb p.stat-1(r1) / is process active, unused, dead
36607 0000C816 760B                   <1>       jna    short sysfork_2
36608                                  <1>             ; beq 1f / it's unused so branch
36609 0000C818 6683FE10               <1>       cmp    si, nproc
36610                                  <1>             ; cmp r1,$nproc / all processes checked
36611 0000C81C 72F0                   <1>       jb     short sysfork_1
```

```
36612                                  <1>             ; blt 1b / no, branch back
36613                                  <1>         ;
36614                                  <1>         ; Retro UNIX 8086 v1. modification:
36615                                  <1>         ;     Parent process returns from 'sysfork' to address
36616                                  <1>         ;     which is just after 'sysfork' system call in parent
36617                                  <1>         ;     process. Child process returns to address which is put
36618                                  <1>         ;     in BX register by parent process for 'sysfork'.
36619                                  <1>         ;
36620                                  <1>             ;add $2,18.(sp) / add 2 to pc when trap occured, points
36621                                  <1>                             ; / to old process return
36622                                  <1>             ; br error1 / no room for a new process
36623 0000C81E E96CFCFFFF              <1>         jmp    error
36624                                  <1> sysfork_2: ; 1:
36625 0000C823 E85183FFFF              <1>         call   allocate_page
36626 0000C828 0F8261FCFFFF            <1>         jc     error
36627 0000C82E 50                      <1>         push   eax   ; UPAGE (user structure page) address
36628                                  <1>         ; Retro UNIX 386 v1 modification!
36629 0000C82F E85485FFFF              <1>         call   duplicate_page_dir
36630                                  <1>             ; EAX = New page directory
36631 0000C834 730B                    <1>         jnc    short sysfork_3
36632 0000C836 58                      <1>         pop    eax   ; UPAGE (user structure page) address
36633 0000C837 E81B85FFFF              <1>         call   deallocate_page
36634 0000C83C E94EFCFFFF              <1>         jmp    error
36635                                  <1> sysfork_3:
36636                                  <1>         ; Retro UNIX 386 v1 modification !
36637 0000C841 56                      <1>         push   esi
36638 0000C842 E8E01E0000              <1>         call   wswap ; save current user (u) structure, user registers
36639                                  <1>                      ; and interrupt return components (for IRET)
36640 0000C847 8705[B8030600]          <1>         xchg   eax, [u.pgdir] ; page directory of the child process
36641 0000C84D A3[BC030600]            <1>         mov    [u.ppgdir], eax ; page directory of the parent process
36642 0000C852 5E                      <1>         pop    esi
36643 0000C853 58                      <1>         pop    eax   ; UPAGE (user structure page) address
36644                                  <1>             ; [u.usp] = esp
36645 0000C854 89F7                    <1>       mov    edi, esi
36646 0000C856 66C1E702                <1>       shl    di, 2
36647 0000C85A 8987[BC000600]          <1>       mov    [edi+p.upage-4], eax ; memory page for 'user' struct
36648 0000C860 A3[B4030600]            <1>       mov    [u.upage], eax ; memory page for 'user' struct (child)
36649                                  <1>       ; 28/08/2015
36650 0000C865 0FB605[B3030600]        <1>       movzx  eax, byte [u.uno] ; parent process number
36651                                  <1>             ; movb u.uno,-(sp) / save parent process number
36652 0000C86C 89C7                    <1>       mov    edi, eax
36653 0000C86E 50                      <1>         push eax ; **
36654 0000C86F 8A87[7F000600]          <1>       mov     al, [edi+p.ttyc-1] ; console tty (parent)
36655                                  <1>       ; 18/09/2015
36656                                  <1>       ;mov     [esi+p.ttyc-1], al ; set child's console tty
36657                                  <1>       ;mov     [esi+p.waitc-1], ah ; 0 ; reset child's wait channel
36658 0000C875 668986[7F000600]        <1>       mov     [esi+p.ttyc-1], ax ; al - set child's console tty
36659                                  <1>                               ; ah - reset child's wait channel
36660 0000C87C 89F0                    <1>       mov    eax, esi
36661 0000C87E A2[B3030600]            <1>       mov    [u.uno], al ; child process number
36662                                  <1>             ;movb r1,u.uno / set child process number to r1
36663 0000C883 FE86[AF000600]          <1>         inc     byte [esi+p.stat-1] ; 1, SRUN, 05/02/2014
36664                                  <1>             ; incb p.stat-1(r1) / set p.stat entry for child
36665                                  <1>                           ; / process to active status
36666                                  <1>             ; mov u.ttyp,r2 / put pointer to parent process'
36667                                  <1>                           ; / control tty buffer in r2
36668                                  <1>                ; beq 2f / branch, if no such tty assigned
36669                                  <1>             ; clrb 6(r2) / clear interrupt character in tty buffer
36670                                  <1>       ; 2:
36671 0000C889 53                      <1>       push   ebx  ; * return address for the child process
36672                                  <1>                    ; * Retro UNIX 8086 v1 feature only !
36673                                  <1>       ; (Retro UNIX 8086 v1 modification!)
36674                                  <1>             ; mov $runq+4,r2
36675 0000C88A BB[54030600]            <1>       mov    ebx, runq+2 ; normal run queue ; 02/01/2017
36676 0000C88F E8031F0000              <1>       call   putlu
36677                                  <1>             ; jsr r0,putlu / put child process on lowest priority
36678                                  <1>                            ; / run queue
36679 0000C894 66D1E6                  <1>       shl    si, 1
36680                                  <1>             ; asl r1 / multiply r1 by 2 to get index
36681                                  <1>                    ; / into p.pid table
36682 0000C897 66FF05[4E030600]        <1>       inc    word [mpid]
36683                                  <1>             ; inc mpid / increment m.pid; get a new process name
36684 0000C89E 66A1[4E030600]          <1>       mov    ax, [mpid]
36685 0000C8A4 668986[1E000600]        <1>       mov    [esi+p.pid-2], ax
36686                                  <1>             ;mov mpid,p.pid-2(r1) / put new process name
36687                                  <1>                                   ; / in child process' name slot
36688 0000C8AB 5A                      <1>       pop    edx  ; * return address for the child process
36689                                  <1>                    ; * Retro UNIX 8086 v1 feature only !
36690 0000C8AC 5B                      <1>       pop    ebx  ; **
36691                                  <1>       ;mov   ebx, [esp] ; ** parent process number
36692                                  <1>             ; movb (sp),r2 / put parent process number in r2
36693 0000C8AD 66D1E3                  <1>       shl    bx, 1
36694                                  <1>             ;asl r2 / multiply by 2 to get index into below tables
36695                                  <1>       ;movzx eax, word [ebx+p.pid-2]
36696 0000C8B0 668B83[1E000600]        <1>       mov    ax, [ebx+p.pid-2]
36697                                  <1>             ; mov p.pid-2(r2),r2 / get process name of parent
36698                                  <1>                                  ; / process
36699 0000C8B7 668986[3E000600]        <1>       mov    [esi+p.ppid-2], ax
36700                                  <1>             ; mov r2,p.ppid-2(r1) / put parent process name
36701                                  <1>                      ; / in parent process slot for child
36702 0000C8BE A3[64030600]            <1>       mov    [u.r0], eax
36703                                  <1>             ; mov r2,*u.r0 / put parent process name on stack
36704                                  <1>                           ; / at location where r0 was saved
36705 0000C8C3 8B2D[5C030600]          <1>       mov    ebp, [u.sp] ; points to return address (EIP for IRET)
36706 0000C8C9 895500                  <1>       mov    [ebp], edx ; *, CS:EIP -> EIP
36707                                  <1>                         ; * return address for the child process
36708                                  <1>             ; mov $sysret1,-(sp) /
36709                                  <1>             ; mov sp,u.usp / contents of sp at the time when
36710                                  <1>                           ; / user is swapped out
36711                                  <1>             ; mov $sstack,sp / point sp to swapping stack space
36712                                  <1>       ; 04/09/2015 - 01/09/2015
36713                                  <1>       ; [u.usp] = esp
```

```
36714 0000C8CC 68[AFC40000]        <1>       push   sysret ; ***
36715 0000C8D1 8925[60030600]      <1>       mov    [u.usp], esp ; points to 'sysret' address (***)
36716                              <1>                           ; (for child process)
36717 0000C8D7 31C0                <1>       xor    eax, eax
36718 0000C8D9 66A3[94030600]      <1>       mov    [u.ttyp], ax ; 0
36719                              <1>       ;
36720 0000C8DF E8431E0000          <1>       call   wswap ; Retro UNIX 8086 v1 modification !
36721                              <1>              ;jsr r0,wswap / put child process out on drum
36722                              <1>              ;jsr r0,unpack / unpack user stack
36723                              <1>              ;mov u.usp,sp / restore user stack pointer
36724                              <1>              ; tst (sp)+ / bump stack pointer
36725                              <1>       ; Retro UNIX 386 v1 modification !
36726 0000C8E4 58                 <1>       pop    eax ; ***
36727 0000C8E5 66D1E3              <1>       shl    bx, 1
36728 0000C8E8 8B83[BC000600]      <1>       mov     eax, [ebx+p.upage-4] ; UPAGE address ; 14/05/2015
36729 0000C8EE E86C1E0000          <1>       call   rswap ; restore parent process 'u' structure,
36730                              <1>                    ; registers and return address (for IRET)
36731                              <1>              ;movb (sp)+,u.uno / put parent process number in u.uno
36732 0000C8F3 0FB705[4E030600]    <1>         movzx   eax, word [mpid]
36733 0000C8FA A3[64030600]        <1>       mov    [u.r0], eax
36734                              <1>              ; mov mpid,*u.r0 / put child process name on stack
36735                              <1>                             ; / where r0 was saved
36736                              <1>              ; add $2,18.(sp) / add 2 to pc on stack; gives parent
36737                              <1>                              ; / process return
36738                              <1>       ;xor   ebx, ebx
36739 0000C8FF 31F6                <1>       xor     esi, esi
36740                              <1>              ;clr r1
36741                              <1> sysfork_4: ; 1: / search u.fp list to find the files
36742                              <1>                ; / opened by the parent process
36743                              <1>       ; 01/09/2015
36744                              <1>       ;xor   bh, bh
36745                              <1>       ;mov   bl, [esi+u.fp]
36746 0000C901 8A86[6A030600]      <1>       mov    al, [esi+u.fp]
36747                              <1>              ; movb u.fp(r1),r2 / get an open file for this process
36748                              <1>         ;or      bl, bl
36749 0000C907 08C0                <1>       or     al, al
36750 0000C909 740D                <1>       jz     short sysfork_5
36751                              <1>              ; beq 2f / file has not been opened by parent,
36752                              <1>                     ; / so branch
36753 0000C90B B40A                <1>       mov    ah, 10 ; Retro UNIX 386 v1 fsp structure size = 10 bytes
36754 0000C90D F6E4                <1>       mul    ah
36755                              <1>       ;movzx ebx, ax
36756 0000C90F 6689C3              <1>       mov    bx, ax
36757                              <1>       ;shl     bx, 3
36758                              <1>              ; asl r2 / multiply by 8
36759                              <1>                    ; asl r2 / to get index into fsp table
36760                              <1>                    ; asl r2
36761 0000C912 FE83[4E010600]      <1>       inc     byte [ebx+fsp-2]
36762                              <1>              ; incb fsp-2(r2) / increment number of processes
36763                              <1>                         ; / using file, because child will now be
36764                              <1>                         ; / using this file
36765                              <1> sysfork_5: ; 2:
36766 0000C918 46                  <1>         inc     esi
36767                              <1>              ; inc r1 / get next open file
36768 0000C919 6683FE0A            <1>         cmp     si, 10
36769                              <1>              ; cmp r1,$10. / 10. files is the maximum number which
36770                              <1>                      ; / can be opened
36771 0000C91D 72E2                <1>       jb     short sysfork_4
36772                              <1>              ; blt 1b / check next entry
36773 0000C91F E98BFBFFFF          <1>       jmp    sysret
36774                              <1>              ; br sysret1
36775                              <1>
36776                              <1> syscreat: ; < create file >
36777                              <1>       ; 27/10/2016
36778                              <1>       ; 25/10/2016, 26/10/2016
36779                              <1>       ; 15/10/2016, 16/10/2016, 17/10/2016
36780                              <1>       ; 10/10/2016 (TRDOS 386 = TRDOS v2.0)
36781                              <1>       ;          -derived from INT_21H.ASM-
36782                              <1>       ;            ("loc_INT21h_create_file")
36783                              <1>         ;    10/07/2011 (12/03/2011)
36784                              <1>         ;    INT 21h Function AH = 3Ch
36785                              <1>         ;    Create File
36786                              <1>         ;    INPUT
36787                              <1>         ;       CX = Attributes
36788                              <1>         ;           DS:DX= Address of zero terminaned path name
36789                              <1>         ;
36790                              <1>       ; 27/12/2015 (Retro UNIX 386 v1.1)
36791                              <1>       ; 14/05/2015 (Retro UNIX 386 v1 - Beginning)
36792                              <1>       ; 27/05/2013 (Retro UNIX 8086 v1)
36793                              <1>       ;
36794                              <1>       ; 'syscreat' called with two arguments; name and mode.
36795                              <1>       ; u.namep points to name of the file and mode is put
36796                              <1>       ; on the stack. 'namei' is called to get i-number of the file.
36797                              <1>       ; If the file aready exists, it's mode and owner remain
36798                              <1>       ; unchanged, but it is truncated to zero length. If the file
36799                              <1>       ; did not exist, an i-node is created with the new mode via
36800                              <1>       ; 'maknod' whether or not the file already existed, it is
36801                              <1>       ; open for writing. The fsp table is then searched for a free
36802                              <1>       ; entry. When a free entry is found, proper data is placed
36803                              <1>       ; in it and the number of this entry is put in the u.fp list.
36804                              <1>       ; The index to the u.fp (also know as the file descriptor)
36805                              <1>       ; is put in the user's r0.
36806                              <1>       ;
36807                              <1>       ; Calling sequence:
36808                              <1>       ;     syscreate; name; mode
36809                              <1>       ; Arguments:
36810                              <1>       ;     name - name of the file to be created
36811                              <1>       ;     mode - mode of the file to be created
36812                              <1>       ; Inputs: (arguments)
36813                              <1>       ; Outputs: *u.r0 - index to u.fp list
36814                              <1>       ;              (the file descriptor of new file)
36815                              <1>       ; ..........................................................
```

```
36816                                  <1>       ;
36817                                  <1>       ; Retro UNIX 8086 v1 modification:
36818                                  <1>       ;       'syscreate' system call has two arguments; so,
36819                                  <1>       ;     * 1st argument, name is pointed to by BX register
36820                                  <1>       ;     * 2nd argument, mode is in CX register
36821                                  <1>       ;
36822                                  <1>       ;     AX register (will be restored via 'u.r0') will return
36823                                  <1>       ;     to the user with the file descriptor/number
36824                                  <1>       ;     (index to u.fp list).
36825                                  <1>       ;
36826                                  <1>       ;call  arg2
36827                                  <1>       ; * name - 'u.namep' points to address of file/path name
36828                                  <1>       ;          in the user's program segment ('u.segmnt')
36829                                  <1>       ;          with offset in BX register (as sysopen argument 1).
36830                                  <1>       ; * mode - sysopen argument 2 is in CX register
36831                                  <1>       ;          which is on top of stack.
36832                                  <1>       ;
36833                                  <1>       ; TRDOS 386 (10/10/2016)
36834                                  <1>       ;
36835                                  <1>         ; INPUT ->
36836                                  <1>         ;       CL = File Attributes
36837                                  <1>       ;            bit 0 (1) - Read only file (R)
36838                                  <1>       ;              bit 1 (1) - Hidden file (H)
36839                                  <1>         ;              bit 2 (1) - System file (R)
36840                                  <1>       ;              bit 3 (1) - Volume label/name (V)
36841                                  <1>         ;              bit 4 (1) - Subdirectory (D)
36842                                  <1>       ;            bit 5 (1) - File has been archived (A)
36843                                  <1>         ;           EBX = Pointer to filename (ASCIIZ) -path-
36844                                  <1>       ;
36845                                  <1>       ; OUTPUT ->
36846                                  <1>       ;          eax = File/Device Handle/Number (index) (AL)
36847                                  <1>       ;          cf = 1 -> Error code in AL
36848                                  <1>       ;
36849                                  <1>       ; Modified Registers: EAX (at the return of system call)
36850                                  <1>       ;
36851                                  <1>       ; Note: If the file is existing and it has not any one
36852                                  <1>       ;       of S,H,R,V,D attributes, it will be truncated
36853                                  <1>       ;       to zero length; otherwise, access error will be
36854                                  <1>       ;       returned.
36855                                  <1>
36856                                  <1> sysmkdir_0:
36857 0000C924 F6C108                <1>       test   cl, 08h ; Volume name
36858 0000C927 740A                  <1>       jz     short syscreat_0
36859                                  <1>
36860                                  <1>       ; Volume name or long name creation
36861                                  <1>       ; is not permitted (in TRDOS 386)!
36862 0000C929 B80B000000            <1>       mov    eax, ERR_FILE_ACCESS  ; 11 ; 'permission denied !'
36863 0000C92E E927020000            <1>         jmp  sysopen_dev_err
36864                                  <1>
36865                                  <1> syscreat_0:
36866                                  <1>         ;mov [u.namep], ebx
36867 0000C933 51                    <1>       push   ecx
36868 0000C934 89DE                  <1>       mov    esi, ebx
36869                                  <1>       ; file name is forced, change directory as temporary
36870                                  <1>       ;mov   ax, 1
36871                                  <1>       ;mov   [FFF_Valid], ah ; 0 ; reset ; 17/10/2016
36872                                  <1>       ;call  set_working_path
36873 0000C936 E8682C0000            <1>       call   set_working_path_x ; 17/10/2016
36874 0000C93B 0F82D8000000          <1>       jc     syscreat_err
36875                                  <1>
36876                                  <1>       ; 16/10/2016
36877 0000C941 803D[835B0100]00      <1>       cmp    byte [SWP_inv_fname], 0
36878 0000C948 776D                  <1>       ja     short  syscreat_inv_fname ; invalid file name !
36879                                  <1>
36880                                  <1>       ; Here, we have a valid path and also a valid file name
36881                                  <1>       ; (Working dir has been changed if the path
36882                                  <1>       ;  -file name string- had contained a dir name.)
36883                                  <1>
36884 0000C94A 6631C0                <1>       xor    ax, ax
36885                                  <1>       ;mov   esi, FindFile_Name
36886 0000C94D E89DB6FFFF            <1>       call   find_first_file
36887 0000C952 59                    <1>       pop    ecx
36888                                  <1>              ; ESI = Directory Entry (FindFile_DirEntry) Location
36889                                  <1>              ; EDI = Directory Buffer Directory Entry Location
36890                                  <1>              ; EAX = File Size
36891                                  <1>              ;  BL = Attributes of The File/Directory
36892                                  <1>              ;  BH = Long Name Yes/No Status (>0 is YES)
36893                                  <1>              ;  DX > 0 : Ambiguous filename chars are used
36894 0000C953 726A                  <1>       jc     short syscreat_1 ; file not found (the good!)
36895                                  <1>                             ; or another error (the bad')
36896                                  <1>
36897                                  <1>       ; (& the uggly!) truncate file to zero length before open
36898                                  <1>
36899                                  <1>       ;'*' and '?' already checked at 'set_working_path' stage
36900                                  <1>       ;and   dx, dx
36901                                  <1>       ;jnz   short sysmkdir_err ; permission denied
36902                                  <1>                              ; invalid filename chars
36903                                  <1>
36904                                  <1>       ;test  cl, 10h ; subdirectory ?
36905                                  <1>       ;jnz   short sysmkdir_err
36906                                  <1>
36907                                  <1>       ; BL = File Attributes:
36908                                  <1>       ;            bit 0 (1) - Read only file (R)
36909                                  <1>       ;              bit 1 (1) - Hidden file (H)
36910                                  <1>         ;              bit 2 (1) - System file (R)
36911                                  <1>       ;              bit 3 (1) - Volume label/name (V)
36912                                  <1>         ;              bit 4 (1) - Subdirectory (D)
36913                                  <1>       ;            bit 5 (1) - File has been archived
36914                                  <1>
36915                                  <1>       ; * existing directory must not be truncated
36916                                  <1>       ;   (we don't know it is empty or not, at this stage)
36917                                  <1>       ; * existing volume name (or a long name) can not be
```

```
36918                                  <1>       ;   re-created or truncated by 'syscreat'
36919                                  <1>       ; * A file with S, H, R attributes must not be truncated
36920                                  <1>       ;   (change attributes to normal, if you need truncate it)
36921                                  <1>
36922 0000C955 F6C31F                 <1>       test   bl, 00011111b  ; check attributes of existing file
36923 0000C958 754F                   <1>       jnz    short sysmkdir_err
36924                                  <1>
36925                                  <1>       ;; normal file, OK to continue...
36926                                  <1>
36927                                  <1>       ; ESI = FindFile_DirEntry
36928 0000C95A 668B4614               <1>       mov    ax, [esi+DirEntry_FstClusHI] ; 20
36929 0000C95E 66C1E010               <1>       shl    ax, 16
36930 0000C962 668B461A               <1>       mov    ax, [esi+DirEntry_FstClusLO] ; 26
36931                                  <1>       ; EAX = First cluster to be truncated/unlinked
36932 0000C966 57                     <1>       push   edi
36933 0000C967 51                     <1>       push   ecx
36934 0000C968 BE00010900             <1>       mov    esi, Logical_DOSDisks
36935 0000C96D 29C9                   <1>       sub    ecx, ecx
36936 0000C96F 8A2D[8E4E0100]         <1>       mov    ch, [Current_Drv]
36937 0000C975 01CE                   <1>       add    esi, ecx
36938                                  <1>       ; ESI = Logical dos drive description table address
36939 0000C977 E8CBF7FFFF             <1>       call   truncate_cluster_chain
36940 0000C97C 59                     <1>       pop    ecx
36941 0000C97D 5F                     <1>       pop    edi
36942 0000C97E 7230                   <1>       jc     short syscreate_truncate_err
36943                                  <1>
36944                                  <1>       ; 26/10/2016
36945                                  <1>       ; EDI = Directory entry address in directory buffer
36946                                  <1>       ; Update directory entry
36947 0000C980 E847DCFFFF             <1>       call   convert_current_date_time
36948                                  <1>       ; OUTPUT -> DX = Date in dos dir entry format
36949                                  <1>         ;       AX = Time in dos dir entry format
36950 0000C985 66894716               <1>       mov    [edi+DirEntry_WrtTime], ax
36951 0000C989 66895718               <1>       mov    [edi+DirEntry_WrtDate], dx
36952 0000C98D 66895712               <1>       mov    [edi+DirEntry_LastAccDate], dx
36953 0000C991 31C0                   <1>       xor    eax, eax ; file size = 0
36954 0000C993 89471C                 <1>       mov    [edi+DirEntry_FileSize], eax ; 0
36955 0000C996 C605[B8560100]02       <1>       mov    byte [DirBuff_ValidData], 2 ; data changed sign
36956 0000C99D BE[84580100]           <1>       mov    esi, FindFile_DirEntry
36957 0000C9A2 B201                   <1>       mov    dl, 1 ; open file for writing
36958 0000C9A4 E9AA000000             <1>       jmp    sysopen_2
36959                                  <1>
36960                                  <1> sysmkdir_err:
36961                                  <1>       ; 1 = write, 2 = read & write, >2 = invalid
36962 0000C9A9 B80B000000             <1>         mov  eax, ERR_FILE_ACCESS  ; 11 ; 'permission denied !'
36963 0000C9AE EB73                   <1>         jmp  short sysopen_err
36964                                  <1>
36965                                  <1> syscreate_truncate_err:
36966 0000C9B0 B812000000             <1>       mov    eax, ERR_DRV_WRITE ; 18 ; 'disk write error !'
36967 0000C9B5 EB6C                   <1>         jmp  short sysopen_err
36968                                  <1>
36969                                  <1> syscreat_inv_fname:  ; invalid file name chars
36970                                  <1>       ; 16/10/2016
36971 0000C9B7 B81A000000             <1>       mov    eax, ERR_INV_FILE_NAME  ; 26 ; invalid file name chars
36972 0000C9BC 59                     <1>       pop    ecx
36973 0000C9BD EB64                   <1>       jmp    sysopen_err
36974                                  <1>
36975                                  <1> syscreat_1:
36976                                  <1>       ; Error code in EAX
36977 0000C9BF 3C02                   <1>         cmp  al, 02h ; 'File not found' error
36978 0000C9C1 7560                   <1>         jne  sysopen_err
36979                                  <1>
36980 0000C9C3 F6C110                 <1>       test   cl, 10h ; Directory
36981 0000C9C6 0F852C020000           <1>       jnz    sysmkdir_2
36982                                  <1>
36983                                  <1> syscreat_2:
36984 0000C9CC BE[74580100]           <1>       mov    esi, FindFile_Name
36985                                  <1>         ;xor edx, edx
36986 0000C9D1 31C0                   <1>         xor  eax, eax ; File Size  = 0
36987 0000C9D3 31DB                   <1>       xor    ebx, ebx
36988 0000C9D5 4B                     <1>       dec    ebx ; FFFFFFFFh -> create empty file
36989                                  <1>                   ;              (only for FAT fs)
36990                                  <1>       ; CL = File Attributes
36991 0000C9D6 E8FAEBFFFF             <1>       call   create_file
36992 0000C9DB 7246                   <1>       jc     sysopen_err
36993                                  <1>              ; EAX = New file's first cluster
36994                                  <1>              ; ESI = Logical Dos Drv Descr. Table Addr.
36995                                  <1>              ; EBX = offset CreateFile_Size
36996                                  <1>              ; ECX = Sectors per cluster (<256)
36997                                  <1>              ; EDX = Directory entry index/number (<65536)
36998                                  <1>       ; 26/10/2016
36999                                  <1>       ;mov   esi, Directory_Buffer
37000                                  <1>       ;shl   dx, 5 ; *32
37001                                  <1>       ;add   esi, edx
37002                                  <1>       ;; esi = directory entry address in directory buffer
37003                                  <1>
37004                                  <1>       ; Here, directory entry has been created but last
37005                                  <1>       ; modification date & time of the parent dir has not
37006                                  <1>       ; been updated, yet!
37007                                  <1>       ; (Note: Directory and FAT buffers have been updated...)
37008                                  <1>
37009 0000C9DD E823DDFFFF             <1>       call   update_parent_dir_lmdt ; now, it is OK too!
37010                                  <1>
37011                                  <1>       ; 25/10/2016
37012 0000C9E2 66B80018               <1>       mov    ax, 1800h
37013 0000C9E6 BE[74580100]           <1>       mov    esi, FindFile_Name
37014 0000C9EB E8FFB5FFFF             <1>       call   find_first_file
37015 0000C9F0 7231                   <1>       jc     short sysopen_err
37016                                  <1>
37017                                  <1>       ; Only possible error after here is
37018                                  <1>       ; "too many open files !" error.
37019                                  <1>       ;
```

```
37020                                  <1>       ; If "syscreat" will return with that error,
37021                                  <1>       ; (the file has been created but it could not be opened)
37022                                  <1>       ; the user must retry to open this file again
37023                                  <1>       ; or must close another file before using
37024                                  <1>       ; "sysopen" system call.
37025                                  <1>
37026 0000C9F2 B201                    <1>       mov    dl, 1 ; open file for writing
37027                                  <1>       ; ESI = Directory Entry (FindFile_DirEntry) Location
37028                                  <1>       ; EAX = File Size (= 0)
37029 0000C9F4 EB5D                    <1>       jmp    short sysopen_2
37030                                  <1>
37031                                  <1> sysopen: ;<open file>
37032                                  <1>       ; 26/10/2016
37033                                  <1>       ; 24/10/2016
37034                                  <1>       ; 17/10/2016
37035                                  <1>       ; 15/10/2016
37036                                  <1>       ; 06/10/2016, 07/10/2016, 08/10/2016
37037                                  <1>       ; 05/10/2016 (TRDOS 386 = TRDOS v2.0)
37038                                  <1>       ;          -derived from INT_21H.ASM-
37039                                  <1>       ;           ("loc_INT21h_open_file")
37040                                  <1>         ;    26/02/2011
37041                                  <1>         ;    INT 21h Function AH = 3Dh
37042                                  <1>         ;    Open File
37043                                  <1>         ;    INPUT
37044                                  <1>         ;       AL= File Access Value
37045                                  <1>       ;           0- Open for reading
37046                                  <1>       ;             1- Open for writing
37047                                  <1>         ;             2- Open for reading and writing
37048                                  <1>         ;          DS:DX= Pointer to filename (ASCIIZ)
37049                                  <1>         ;
37050                                  <1>       ; 14/05/2015 (Retro UNIX 386 v1 - Beginning)
37051                                  <1>       ; 22/05/2013 - 27/05/2013 (Retro UNIX 8086 v1)
37052                                  <1>       ;
37053                                  <1>       ; 'sysopen' opens a file in following manner:
37054                                  <1>       ;    1) The second argument in a sysopen says whether to
37055                                  <1>       ;      open the file ro read (0) or write (>0).
37056                                  <1>       ;    2) I-node of the particular file is obtained via 'namei'.
37057                                  <1>       ;    3) The file is opened by 'iopen'.
37058                                  <1>       ;    4) Next housekeeping is performed on the fsp table
37059                                  <1>       ;      and the user's open file list - u.fp.
37060                                  <1>       ;      a) u.fp and fsp are scanned for the next available slot.
37061                                  <1>       ;      b) An entry for the file is created in the fsp table.
37062                                  <1>       ;      c) The number of this entry is put on u.fp list.
37063                                  <1>       ;      d) The file descriptor index to u.fp list is pointed
37064                                  <1>       ;         to by u.r0.
37065                                  <1>       ;
37066                                  <1>       ; Calling sequence:
37067                                  <1>       ;      sysopen; name; mode
37068                                  <1>       ; Arguments:
37069                                  <1>       ;      name - file name or path name
37070                                  <1>       ;      mode - 0 to open for reading
37071                                  <1>       ;             1 to open for writing
37072                                  <1>       ; Inputs: (arguments)
37073                                  <1>       ; Outputs: *u.r0 - index to u.fp list (the file descriptor)
37074                                  <1>       ;              is put into r0's location on the stack.
37075                                  <1>       ; ..........................................................
37076                                  <1>       ;
37077                                  <1>       ; Retro UNIX 8086 v1 modification:
37078                                  <1>       ;       'sysopen' system call has two arguments; so,
37079                                  <1>       ;      * 1st argument, name is pointed to by BX register
37080                                  <1>       ;      * 2nd argument, mode is in CX register
37081                                  <1>       ;
37082                                  <1>       ;      AX register (will be restored via 'u.r0') will return
37083                                  <1>       ;      to the user with the file descriptor/number
37084                                  <1>       ;      (index to u.fp list).
37085                                  <1>       ;
37086                                  <1>       ;call  arg2
37087                                  <1>       ; * name - 'u.namep' points to address of file/path name
37088                                  <1>       ;          in the user's program segment ('u.segmnt')
37089                                  <1>       ;          with offset in BX register (as sysopen argument 1).
37090                                  <1>       ; * mode - sysopen argument 2 is in CX register
37091                                  <1>       ;          which is on top of stack.
37092                                  <1>       ;
37093                                  <1>       ; jsr r0,arg2 / get sys args into u.namep and on stack
37094                                  <1>       ;
37095                                  <1>            ; system call registers: ebx, ecx (through 'sysenter')
37096                                  <1>       ;
37097                                  <1>       ; TRDOS 386 (05/10/2016)
37098                                  <1>       ;
37099                                  <1>         ; INPUT ->
37100                                  <1>         ;       CL = File Access Value (Open Mode)
37101                                  <1>       ;            0 - Open file for reading
37102                                  <1>       ;              1 - Open file for writing
37103                                  <1>         ;              2 - Open device for reading
37104                                  <1>       ;            3 - Open device for writing
37105                                  <1>         ;           EBX = Pointer to filename/devicename (ASCIIZ)
37106                                  <1>       ; OUTPUT ->
37107                                  <1>       ;          eax = File/Device Handle/Number (index) (AL)
37108                                  <1>       ;          cf = 1 -> Error code in AL
37109                                  <1>       ;
37110                                  <1>       ; Modified Registers: EAX (at the return of system call)
37111                                  <1>       ;
37112                                  <1>
37113 0000C9F6 80F901                  <1>       cmp    cl, 1 ; read file (0), write file (1)
37114 0000C9F9 7614                    <1>       jna    short sysopen_0
37115                                  <1>
37116 0000C9FB 80F903                  <1>       cmp    cl, 3
37117 0000C9FE 0F8640010000            <1>       jna    sysopen_device
37118                                  <1>
37119                                  <1>       ; Invalid access code
37120 0000CA04 B817000000              <1>       mov    eax, ERR_INV_PARAMETER
37121 0000CA09 0F874B010000            <1>       ja     sysopen_dev_err
```

```
37122                                  <1>
37123                                  <1> sysopen_0:
37124                                  <1>       ;mov   [u.namep], ebx
37125 0000CA0F 51                      <1>       push   ecx
37126 0000CA10 89DE                    <1>       mov    esi, ebx
37127                                  <1>       ; file name is forced, change directory as temporary
37128                                  <1>       ;mov   ax, 1
37129                                  <1>       ;mov   [FFF_Valid], ah ; 0 ; reset ; 17/10/2016
37130                                  <1>       ;call  set_working_path
37131 0000CA12 E88C2B0000              <1>       call   set_working_path_x ; 17/10/2016
37132 0000CA17 731E                    <1>       jnc    short sysopen_1
37133                                  <1>
37134                                  <1> syscreat_err: ; ecx = file attributes (for 'syscreat')
37135 0000CA19 59                      <1>       pop    ecx ; open mode
37136 0000CA1A 21C0                    <1>       and    eax, eax  ; 0 -> Bad Path!
37137 0000CA1C 7505                    <1>       jnz    short sysopen_err
37138                                  <1>       ; eax = 0
37139 0000CA1E B80C000000              <1>       mov    eax, ERR_DIR_NOT_FOUND ; Directory not found !
37140                                  <1> sysopen_err:
37141 0000CA23 A3[64030600]            <1>       mov    [u.r0], eax
37142 0000CA28 A3[C8030600]            <1>       mov    [u.error], eax
37143 0000CA2D E8462C0000              <1>       call   reset_working_path
37144 0000CA32 E958FAFFFF              <1>       jmp    error
37145                                  <1>
37146                                  <1> sysopen_1:
37147                                  <1>       ;mov   esi, FindFile_Name
37148 0000CA37 66B80018                <1>         mov  ax, 1800h ; Only files
37149 0000CA3B E8AFB5FFFF              <1>       call   find_first_file
37150 0000CA40 5A                      <1>       pop    edx
37151 0000CA41 72E0                    <1>       jc     short sysopen_err ; eax = 2 (File not found !)
37152                                  <1>
37153                                  <1>       ; check_open_file_attr_access_code
37154                                  <1>
37155 0000CA43 F6C307                  <1>         test bl, 7  ; system, hidden, readonly
37156 0000CA46 740B                    <1>         jz   short sysopen_2
37157                                  <1>
37158 0000CA48 20D2                    <1>       and    dl, dl ; 0 = read mode
37159 0000CA4A 7407                    <1>       jz     short sysopen_2
37160                                  <1>
37161                                  <1>       ; 1 = write, 2 = read & write, >2 = invalid
37162 0000CA4C B80B000000              <1>         mov  eax, ERR_FILE_ACCESS  ; 11  = 'permission denied !'
37163 0000CA51 EBD0                    <1>         jmp  short sysopen_err
37164                                  <1>
37165                                  <1> sysopen_2:
37166                                  <1>       ; esi = Directory Entry (FindFile_DirEntry) Location
37167 0000CA53 89F3                    <1>       mov    ebx, esi
37168 0000CA55 31F6                    <1>         xor     esi, esi ; 0
37169 0000CA57 31FF                    <1>         xor     edi, edi ; 0
37170                                  <1> sysopen_3: ; scan the list of entries in fsp table
37171 0000CA59 80BE[6A030600]00        <1>         cmp     byte [esi+u.fp], 0
37172 0000CA60 760F                    <1>         jna     short sysopen_4 ; empty slot
37173 0000CA62 6646                    <1>         inc     si
37174 0000CA64 6683FE0A                <1>         cmp     si, 10
37175 0000CA68 72EF                    <1>       jb     short sysopen_3
37176                                  <1> toomanyf:
37177 0000CA6A B80D000000              <1>       mov    eax, ERR_TOO_MANY_FILES ; too many open files !
37178 0000CA6F EBB2                    <1>       jmp    short sysopen_err
37179                                  <1>
37180                                  <1> sysopen_4:
37181 0000CA71 80BF[F25E0100]00        <1>         cmp     byte [edi+OF_MODE], 0 ; Scan open files table
37182 0000CA78 760A                    <1>       jna     short sysopen_5
37183 0000CA7A 6647                    <1>       inc    di
37184 0000CA7C 6683FF0A                <1>       cmp     di, OPENFILES ; max. number of open files (=10)
37185 0000CA80 72EF                    <1>       jb     short sysopen_4
37186 0000CA82 EBE6                    <1>       jmp    short toomanyf
37187                                  <1>
37188                                  <1> sysopen_5:
37189 0000CA84 FEC2                    <1>       inc    dl
37190 0000CA86 8897[F25E0100]          <1>         mov     [edi+OF_MODE], dl
37191 0000CA8C 8A15[32580100]          <1>       mov    dl, [FindFile_Drv]
37192 0000CA92 8897[E85E0100]          <1>         mov     [edi+OF_DRIVE], dl ; Logical DOS drive number
37193 0000CA98 66C1E702                <1>       shl    di, 2 ; *4 (dword offset)
37194                                  <1>
37195 0000CA9C 8987[385F0100]          <1>       mov    [edi+OF_SIZE], eax ; File size in bytes
37196                                  <1>
37197 0000CAA2 668B4314                <1>         mov  ax, [ebx+DirEntry_FstClusHI]
37198 0000CAA6 C1E010                  <1>       shl    eax, 16
37199 0000CAA9 668B431A                <1>       mov    ax, [ebx+DirEntry_FstClusLO]
37200 0000CAAD 8987[C05E0100]          <1>       mov     [edi+OF_FCLUSTER], eax ; First cluster
37201 0000CAB3 8987[D85F0100]          <1>       mov     [edi+OF_CCLUSTER], eax ; Current cluster
37202                                  <1>
37203 0000CAB9 31DB                    <1>         xor  ebx, ebx
37204 0000CABB 899F[105F0100]          <1>         mov     [edi+OF_POINTER], ebx ; offset pointer (0)
37205 0000CAC1 899F[00600100]          <1>         mov     [edi+OF_CCINDEX], ebx ; cluster index (0)
37206                                  <1>
37207 0000CAC7 A1[A4580100]            <1>       mov    eax, [FindFile_DirFirstCluster]
37208 0000CACC 8987[605F0100]          <1>       mov    [edi+OF_DIRFCLUSTER], eax
37209                                  <1>
37210 0000CAD2 A1[A8580100]            <1>       mov    eax, [FindFile_DirCluster]
37211 0000CAD7 8987[885F0100]          <1>       mov    [edi+OF_DIRCLUSTER], eax
37212                                  <1>
37213                                  <1>       ; Get (& Save) Volume ID
37214                                  <1>       ; Important for files of removable drives
37215                                  <1>       ; (In order to check the drive has same volume/disk)
37216 0000CADD 88D7                    <1>       mov    bh, dl
37217 0000CADF 81C300010900            <1>         add  ebx, Logical_DOSDisks
37218 0000CAE5 8A4303                  <1>         mov  al, [ebx+LD_FATType]
37219 0000CAE8 3C01                    <1>         cmp  al, 1
37220 0000CAEA 7209                    <1>         jb   short sysopen_6_fs
37221 0000CAEC 3C02                    <1>         cmp  al, 2
37222 0000CAEE 770A                    <1>         ja   short sysopen_6_fat32
37223                                  <1> sysopen_6_fat:
```

```
37224 0000CAF0 8B432D              <1>         mov  eax, [ebx+LD_BPB+VolumeID]
37225 0000CAF3 EB08                <1>         jmp  short sysopen_7
37226                              <1> sysopen_6_fs:
37227 0000CAF5 8B4328              <1>         mov  eax, [ebx+LD_FS_VolumeSerial]
37228 0000CAF8 EB03                <1>         jmp  short sysopen_7
37229                              <1> sysopen_6_fat32:
37230 0000CAFA 8B4349              <1>         mov  eax, [ebx+LD_BPB+FAT32_VolID]
37231                              <1> sysopen_7:
37232 0000CAFD A3[844E0100]        <1>         mov  [Current_VolSerial], eax
37233                              <1>
37234 0000CB02 8987[B05F0100]      <1>       mov    [edi+OF_VOLUMEID], eax
37235                              <1>
37236                              <1>       ; 24/10/2016
37237 0000CB08 66D1EF              <1>       shr    di, 1 ; 4/2, word offset
37238 0000CB0B 668B1D[AC580100]    <1>       mov    bx, [FindFile_DirEntryNumber]
37239 0000CB12 66899F[28600100]    <1>       mov    [edi+OF_DIRENTRY], bx
37240                              <1>
37241 0000CB19 31D2                <1>       xor    edx, edx
37242                              <1>       ;shr   di, 2 ; /4 (byte offset)
37243 0000CB1B 66D1EF              <1>       shr    di, 1 ; 2/2, byte offset
37244 0000CB1E 8897[065F0100]      <1>       mov    byte [edi+OF_OPENCOUNT], dl ; 0
37245 0000CB24 8897[FC5E0100]      <1>       mov    byte [edi+OF_STATUS], dl ; 0
37246                              <1>
37247 0000CB2A 89FB                <1>       mov    ebx, edi
37248 0000CB2C FEC3                <1>       inc    bl
37249                              <1>
37250 0000CB2E 889E[6A030600]      <1>         mov     [esi+u.fp], bl ; Open File Entry Number
37251 0000CB34 8935[64030600]      <1>       mov     [u.r0], esi ; move index to u.fp list
37252                              <1>                         ; into eax on stack
37253                              <1>
37254 0000CB3A E8392B0000          <1>         call       reset_working_path
37255                              <1>
37256 0000CB3F E96BF9FFFF          <1>       jmp    sysret
37257                              <1>
37258                              <1>       ; (Retro UNIX 386 v1.0)
37259                              <1>       ; 'fsp' table (10 bytes/entry)
37260                              <1>       ; bit 15                            bit 0
37261                              <1>       ; ---|-----------------------------------------
37262                              <1>       ; r/w|       i-number of open file
37263                              <1>       ; ---|-----------------------------------------
37264                              <1>       ;               device number
37265                              <1>       ; ----------------------------------------------
37266                              <1>       ; offset pointer, r/w pointer to file (bit 0-15)
37267                              <1>       ; ----------------------------------------------
37268                              <1>       ; offset pointer, r/w pointer to file (bit 16-31)
37269                              <1>       ; ---------------------|------------------------
37270                              <1>       ;  flag that says file    | number of processes
37271                              <1>       ;   has been deleted      | that have file open
37272                              <1>       ; ---------------------|------------------------
37273                              <1>
37274                              <1> sysopen_device:
37275                              <1>       ; 15/10/2016
37276                              <1>       ; 08/10/2016
37277                              <1>       ; 07/10/2016 (TRDOS 386 = TRDOS v2.0)
37278 0000CB44 51                  <1>       push   ecx ; open mode
37279 0000CB45 89E5                <1>       mov    ebp, esp
37280 0000CB47 B910000000          <1>       mov    ecx, 16 ; transfer length = 16 bytes
37281 0000CB4C 29CC                <1>       sub    esp, ecx
37282 0000CB4E 89E7                <1>       mov    edi, esp ; destination address
37283 0000CB50 89DE                <1>       mov    esi, ebx ; dev name in user's memory space
37284 0000CB52 E8751D0000          <1>       call   transfer_from_user_buffer
37285 0000CB57 7310                <1>       jnc    short sysopen_dev_0
37286                              <1>       ; eax = ERR_OUT_OF_MEMORY = 4 = ERR_MINOR_IM
37287 0000CB59 59                  <1>       pop    ecx
37288                              <1> sysopen_dev_err:
37289 0000CB5A A3[64030600]        <1>       mov    [u.r0], eax
37290 0000CB5F A3[C8030600]        <1>       mov    [u.error], eax
37291 0000CB64 E926F9FFFF          <1>       jmp    error
37292                              <1> sysopen_dev_0:
37293 0000CB69 89FE                <1>       mov    esi, edi ; Device name addr (max. 16 bytes, ASCIIZ)
37294                              <1>                       ; for example: "tty, TTY, /dev/tty"
37295 0000CB6B E8B02D0000          <1>       call   get_device_number
37296 0000CB70 89EC                <1>       mov    esp, ebp
37297 0000CB72 59                  <1>       pop    ecx
37298 0000CB73 7307                <1>       jnc    short sysopen_dev_1
37299 0000CB75 B818000000          <1>       mov    eax, ERR_INV_DEV_NAME ; 24 ; 'invalid device name !'
37300 0000CB7A EBDE                <1>       jmp    short sysopen_dev_err
37301                              <1> sysopen_dev_1:
37302                              <1>       ; eax = Device Number (AL)
37303                              <1>       ;  cl = Open mode (2 = device read, 3 = device write)
37304 0000CB7C 31DB                <1>         xor     ebx, ebx ; 0
37305                              <1> sysopen_dev_2: ; scan the list of entries
37306 0000CB7E 389B[6A030600]      <1>         cmp     [ebx+u.fp], bl ; 0
37307 0000CB84 760E                <1>         jna     short sysopen_dev_3 ; empty slot
37308 0000CB86 FEC3                <1>         inc     bl
37309 0000CB88 80FB0A              <1>         cmp     bl, 10
37310 0000CB8B 72F1                <1>       jb     short sysopen_dev_2
37311                              <1>       ;
37312 0000CB8D B80D000000          <1>       mov    eax, ERR_TOO_MANY_FILES ; too many open files !
37313 0000CB92 EBC6                <1>       jmp    short sysopen_dev_err
37314                              <1> sysopen_dev_3:
37315 0000CB94 891D[64030600]      <1>       mov    [u.r0], ebx ; File/Device index/handle/descriptor
37316                              <1>       ; eax = device number (entry offset)
37317 0000CB9A 8AA8[845C0100]      <1>       mov    ch, [eax+DEV_ACCESS] ; bit 0 = accessable by users
37318                              <1>                                   ; bit 1 = read access perm
37319                              <1>                                   ; bit 2 = write access perm
37320                              <1>                                   ; bit 3 = IOCTL permit to users
37321                              <1>                                   ; bit 4 = block device if set
37322                              <1>                                   ; bit 5 = 16 bit or 1024 byte
37323                              <1>                                   ; bit 6 = 32 bit or 2048 byte
37324                              <1>                                   ; bit 7 = installable device drv
37325 0000CBA0 F6C501              <1>       test   ch, 1 ; accessable by normal users (except root)
```

```
37326 0000CBA3 7510                <1>       jnz    short sysopen_dev_4 ; yes, permission has been given
37327 0000CBA5 803D[B0030600]00    <1>       cmp    byte [u.uid], 0 ; root?
37328 0000CBAC 7607                <1>       jna    short sysopen_dev_4 ; superuser can open all devices
37329                              <1> sysopen_dev_perm_err:
37330 0000CBAE B80B000000          <1>       mov    eax, ERR_DEV_ACCESS  ; 11  = 'permission denied !'
37331 0000CBB3 EBA5                <1>       jmp    short sysopen_dev_err
37332                              <1> sysopen_dev_4:
37333 0000CBB5 D0ED                <1>       shr    ch, 1 ; result: 1 = read, 2 = write, 3 = r & w
37334 0000CBB7 FEC9                <1>       dec    cl  ; result: 1 = read, 2 = write
37335 0000CBB9 84E9                <1>       test   cl, ch
37336 0000CBBB 74F1                <1>       jz     short sysopen_dev_perm_err
37337                              <1>
37338 0000CBBD D0E5                <1>       shl    ch, 1 ; bit 0 = 0
37339                              <1>       ; eax = device number (entry offset)
37340 0000CBBF E8782E0000          <1>       call   device_open
37341 0000CBC4 72E8                <1>       jc     short sysopen_dev_perm_err
37342                              <1>
37343                              <1>       ; eax = device number (entry offset)
37344 0000CBC6 0C80                <1>       or     al, 80h ; set device bit (set bit 7 to 1)
37345 0000CBC8 8B1D[64030600]      <1>       mov    ebx, [u.r0]
37346 0000CBCE 8883[6A030600]      <1>       mov    [ebx+u.fp], al      ; bit 7 (=1) points to device
37347                              <1>
37348 0000CBD4 E9D6F8FFFF          <1>       jmp    sysret
37349                              <1>
37350                              <1> sysmkdir: ; < make directory >
37351                              <1>       ; 15/10/2016
37352                              <1>       ; 10/10/2016 (TRDOS 386 = TRDOS v2.0)
37353                              <1>       ;           -derived from INT_21H.ASM-
37354                              <1>       ;             ("loc_INT21h_create_file")
37355                              <1>         ;   10/07/2011 (12/03/2011)
37356                              <1>         ;   INT 21h Function AH = 3Ch
37357                              <1>         ;   Create File
37358                              <1>         ;   INPUT
37359                              <1>         ;      CX = Attributes
37360                              <1>         ;          DS:DX= Address of zero terminaned path name
37361                              <1>         ;
37362                              <1>         ;
37363                              <1>       ; 14/05/2015 (Retro UNIX 386 v1 - Beginning)
37364                              <1>       ; 27/05/2013 - 02/08/2013 (Retro UNIX 8086 v1)
37365                              <1>       ;
37366                              <1>       ; 'sysmkdir' creates an empty directory whose name is
37367                              <1>       ; pointed to by arg 1. The mode of the directory is arg 2.
37368                              <1>       ; The special entries '.' and '..' are not present.
37369                              <1>       ; Errors are indicated if the directory already exists or
37370                              <1>       ; user is not the super user.
37371                              <1>       ;
37372                              <1>       ; Calling sequence:
37373                              <1>       ;     sysmkdir; name; mode
37374                              <1>       ; Arguments:
37375                              <1>       ;     name - points to the name of the directory
37376                              <1>       ;     mode - mode of the directory
37377                              <1>       ; Inputs: (arguments)
37378                              <1>       ; Outputs: -
37379                              <1>       ;    (sets 'directory' flag to 1;
37380                              <1>       ;    'set user id on execution' and 'executable' flags to 0)
37381                              <1>       ; ..........................................................
37382                              <1>       ;
37383                              <1>       ; Retro UNIX 8086 v1 modification:
37384                              <1>       ;      'sysmkdir' system call has two arguments; so,
37385                              <1>       ;     * 1st argument, name is pointed to by BX register
37386                              <1>       ;     * 2nd argument, mode is in CX register
37387                              <1>       ;
37388                              <1>       ; TRDOS 386 (10/10/2016)
37389                              <1>       ;
37390                              <1>         ; INPUT ->
37391                              <1>         ;      CL = Directory Attributes
37392                              <1>       ;           bit 0 (1) - Read only file/dir (R)
37393                              <1>       ;            bit 1 (1) - Hidden file/dir (H)
37394                              <1>         ;            bit 2 (1) - System file/dir (R)
37395                              <1>       ;            bit 3 (1) - Volume label/name (V)
37396                              <1>         ;            bit 4 (1) - Subdirectory (D)
37397                              <1>       ;            bit 5 (1) - File/Dir has been archived (A)
37398                              <1>       ;        CX = 0 -> create normal directory
37399                              <1>         ;          EBX = Pointer to directory name (ASCIIZ) -path-
37400                              <1>       ;
37401                              <1>       ; OUTPUT ->
37402                              <1>       ;         eax = First cluster of the new directory
37403                              <1>       ;         cf = 1 -> Error code in AL
37404                              <1>       ;
37405                              <1>       ; Modified Registers: EAX (at the return of system call)
37406                              <1>       ;
37407                              <1>       ; Note: If the file or directory is existing
37408                              <1>       ;     an access error will be returned.
37409                              <1>
37410 0000CBD9 6621C9              <1>       and    cx, cx ; if cx = 0 -> create a normal subdir
37411 0000CBDC 7413                <1>       jz     short sysmkdir_1
37412                              <1>
37413 0000CBDE F6C110              <1>       test   cl, 10h ; if dir flags set, also use other flags
37414 0000CBE1 0F853DFDFFFF        <1>       jnz    sysmkdir_0 ; jump to head of 'syscreat'
37415                              <1>
37416                              <1>       ; CX has wrong flags
37417 0000CBE7 B817000000          <1>       mov    eax, ERR_INV_FLAGS
37418 0000CBEC E969FFFFFF          <1>       jmp    sysopen_dev_err
37419                              <1>
37420                              <1> sysmkdir_1:
37421 0000CBF1 B110                <1>       mov    cl, 10h ; set subdir flag and reset other flags
37422 0000CBF3 E92CFDFFFF          <1>       jmp    sysmkdir_0 ; jump to head of 'syscreat'
37423                              <1> sysmkdir_2:
37424                              <1>       ; jump from 'syscreat' ; from 'syscreat_1'
37425                              <1>       ;  CL = Directory attributes/flags
37426 0000CBF8 BE[74580100]        <1>       mov    esi, FindFile_Name
37427 0000CBFD E803D7FFFF          <1>       call   make_sub_directory
```

```
37428 0000CC02 0F821BFEFFFF        <1>       jc    sysopen_err       ; NOTE: Old type (TRDOS 8086)
37429                              <1>                               ; error codes must be modified
37430                              <1>                               ; for next TRDOS 386 versions
37431                              <1>                               ; (10/10/2016)
37432                              <1>                               ; Old (MSDOS type)
37433                              <1>                               ; error codes (2011):
37434                              <1>                               ;  2 = file not found
37435                              <1>                               ;  3 = directory not found
37436                              <1>                               ;  5 = access denied
37437                              <1>                               ; 12 = no more files
37438                              <1>                                ; 19 = disk write protected
37439                              <1>                               ; 39 = insufficient disk space
37440                              <1>                               ; 'sysdefs.s' ; 10/10/2016
37441                              <1>
37442 0000CC08 A3[64030600]        <1>       mov    [u.r0], eax ; New sub dir's first cluster
37443                              <1>
37444 0000CC0D E8662A0000          <1>         call       reset_working_path
37445                              <1>
37446 0000CC12 E998F8FFFF          <1>       jmp    sysret
37447                              <1>
37448                              <1> sysclose: ;<close file>
37449                              <1>       ; 06/10/2016 (TRDOS 386 = TRDOS v2.0)
37450                              <1>       ;
37451                              <1>       ; 14/05/2015 (Retro UNIX 386 v1 - Beginning)
37452                              <1>       ; 22/05/2013 - 26/05/2013 (Retro UNIX 8086 v1)
37453                              <1>       ;
37454                              <1>       ; 'sysclose', given a file descriptor in 'u.r0', closes the
37455                              <1>       ; associated file. The file descriptor (index to 'u.fp' list)
37456                              <1>       ; is put in r1 and 'fclose' is called.
37457                              <1>       ;
37458                              <1>       ; Calling sequence:
37459                              <1>       ;     sysclose
37460                              <1>       ; Arguments:
37461                              <1>       ;     -
37462                              <1>       ; Inputs: *u.r0 - file descriptor
37463                              <1>       ; Outputs: -
37464                              <1>       ; ..........................................................
37465                              <1>       ;
37466                              <1>       ; Retro UNIX 8086 v1 modification:
37467                              <1>       ;      The user/application program puts file descriptor
37468                              <1>       ;       in BX register as 'sysclose' system call argument.
37469                              <1>       ;      (argument transfer method 1)
37470                              <1>
37471                              <1>       ; TRDOS 386 (06/10/2016)
37472                              <1>       ;
37473                              <1>         ; INPUT ->
37474                              <1>         ;       EBX = File Handle/Number (file index) (AL)
37475                              <1>       ; OUTPUT ->
37476                              <1>       ;         cf = 0 -> EAX = 0
37477                              <1>       ;         cf = 1 -> Error code in EAX (ERR_FILE_NOT_OPEN)
37478                              <1>       ;
37479                              <1>       ; Modified Registers: EAX (at the return of system call)
37480                              <1>       ;
37481                              <1>
37482 0000CC17 89D8                <1>       mov    eax, ebx
37483 0000CC19 31DB                <1>       xor    ebx, ebx
37484 0000CC1B 891D[64030600]      <1>       mov    [u.r0], ebx ; 0  ; return value of EAX
37485 0000CC21 E88F0E0000          <1>       call   fclose
37486 0000CC26 0F8383F8FFFF        <1>       jnc    sysret
37487 0000CC2C B80A000000          <1>       mov    eax, ERR_FILE_NOT_OPEN ; file not open !
37488 0000CC31 A3[C8030600]        <1>       mov    [u.error], eax ;
37489 0000CC36 A3[64030600]        <1>       mov    [u.r0], eax ; ! invalid handle !
37490 0000CC3B E94FF8FFFF          <1>       jmp    error
37491                              <1>
37492                              <1> sysread: ; < read from file >
37493                              <1>       ; 11/10/2016 (TRDOS 386 = TRDOS v2.0)
37494                              <1>       ;          -derived from INT_21H.ASM-
37495                              <1>       ;           ("loc_INT21h_read_file")
37496                              <1>         ;   13/03/2011 (05/03/2011)
37497                              <1>         ;   INT 21h Function AH = 3Fh
37498                              <1>         ;   Read from a File
37499                              <1>         ;   INPUT
37500                              <1>       ;        BX = File Handle
37501                              <1>         ;      CX = Number of bytes to read
37502                              <1>         ;          DS:DX= Buffer address
37503                              <1>         ;
37504                              <1>       ; Note: TRDOS 386 'sysread' has been derived from
37505                              <1>       ;     Retro UNIX 386 v1 'sysread', except a few
37506                              <1>       ;     code modifications.
37507                              <1>       ;
37508                              <1>       ; 13/05/2015 (Retro UNIX 386 v1)
37509                              <1>       ; 11/05/2015 (Retro UNIX 386 v1 - Beginning)
37510                              <1>       ; 23/05/2013 (Retro UNIX 8086 v1)
37511                              <1>       ;
37512                              <1>       ; 'sysread' is given a buffer to read into and the number of
37513                              <1>       ; characters to be read. If finds the file from the file
37514                              <1>       ; descriptor located in *u.r0 (r0). This file descriptor
37515                              <1>       ; is returned from a successful open call (sysopen).
37516                              <1>       ; The i-number of file is obtained via 'rw1' and the data
37517                              <1>       ; is read into core via 'readi'.
37518                              <1>       ;
37519                              <1>       ; Calling sequence:
37520                              <1>       ;     sysread; buffer; nchars
37521                              <1>       ; Arguments:
37522                              <1>       ;     buffer - location of contiguous bytes where
37523                              <1>       ;             input will be placed.
37524                              <1>       ;     nchars - number of bytes or characters to be read.
37525                              <1>       ; Inputs: *u.r0 - file descriptor (& arguments)
37526                              <1>       ; Outputs: *u.r0 - number of bytes read.
37527                              <1>       ; ..........................................................
37528                              <1>       ;
37529                              <1>       ; Retro UNIX 8086 v1 modification:
```

```
37530                                  <1>       ;       'sysread' system call has three arguments; so,
37531                                  <1>       ;     * 1st argument, file descriptor is in BX register
37532                                  <1>       ;     * 2nd argument, buffer address/offset in CX register
37533                                  <1>       ;     * 3rd argument, number of bytes is in DX register
37534                                  <1>       ;
37535                                  <1>       ;     AX register (will be restored via 'u.r0') will return
37536                                  <1>       ;     to the user with number of bytes read.
37537                                  <1>       ;
37538                                  <1>       ; TRDOS 386 (05/10/2016)
37539                                  <1>       ;
37540                                  <1>         ; INPUT ->
37541                                  <1>         ;       EBX = File handle (descriptor/index)
37542                                  <1>       ;         ECX = Buffer address
37543                                  <1>         ;           EDX = Number of bytes
37544                                  <1>       ; OUTPUT ->
37545                                  <1>       ;          EAX = Number of bytes have been read
37546                                  <1>       ;          cf = 1 -> Error code in AL
37547                                  <1>       ;
37548                                  <1>       ; Modified Registers: EAX (at the return of system call)
37549                                  <1>       ;
37550                                  <1>
37551                                  <1>       ; EBX = File descriptor
37552 0000CC40 E8BE0E0000               <1>       call   getf1
37553 0000CC45 7277                     <1>       jc     short device_read ; read data from device
37554                                  <1>       ; EAX = First cluster of the file
37555                                  <1>
37556 0000CC47 E83F000000               <1>       call   rw1
37557 0000CC4C 730A                     <1>       jnc    short sysread_0
37558                                  <1>
37559 0000CC4E A3[64030600]             <1>       mov    [u.r0], eax ; error code
37560 0000CC53 E937F8FFFF               <1>       jmp    error
37561                                  <1>
37562                                  <1> sysread_0:
37563 0000CC58 E84C170000               <1>       call   readi
37564 0000CC5D EB1D                     <1>       jmp    short rw0
37565                                  <1>
37566                                  <1> syswrite: ; < write to file >
37567                                  <1>       ; 23/10/2016
37568                                  <1>       ; 11/10/2016 (TRDOS 386 = TRDOS v2.0)
37569                                  <1>       ;          -derived from INT_21H.ASM-
37570                                  <1>       ;           ("loc_INT21h_write_file")
37571                                  <1>         ;   13/03/2011 (05/03/2011)
37572                                  <1>         ;   INT 21h Function AH = 40h
37573                                  <1>         ;   Write to a File
37574                                  <1>         ;   INPUT
37575                                  <1>       ;        BX = File Handle
37576                                  <1>         ;      CX = Number of bytes to write
37577                                  <1>         ;          DS:DX= Buffer address
37578                                  <1>         ;
37579                                  <1>       ; Note: TRDOS 386 'sysrwrite' has been derived from
37580                                  <1>       ;     Retro UNIX 386 v1 'syswrite', except a few
37581                                  <1>       ;     code modifications.
37582                                  <1>       ;
37583                                  <1>
37584                                  <1>       ; 13/05/2015 (Retro UNIX 386 v1)
37585                                  <1>       ; 11/05/2015 (Retro UNIX 386 v1 - Beginning)
37586                                  <1>       ; 23/05/2013 (Retro UNIX 8086 v1)
37587                                  <1>       ;
37588                                  <1>       ; 'syswrite' is given a buffer to write onto an output file
37589                                  <1>       ; and the number of characters to write. If finds the file
37590                                  <1>       ; from the file descriptor located in *u.r0 (r0). This file
37591                                  <1>       ; descriptor is returned from a successful open or create call
37592                                  <1>       ; (sysopen or syscreat). The i-number of file is obtained via
37593                                  <1>       ; 'rw1' and buffer is written on the output file via 'write'.
37594                                  <1>       ;
37595                                  <1>       ; Calling sequence:
37596                                  <1>       ;     syswrite; buffer; nchars
37597                                  <1>       ; Arguments:
37598                                  <1>       ;     buffer - location of contiguous bytes to be writtten.
37599                                  <1>       ;     nchars - number of characters to be written.
37600                                  <1>       ; Inputs: *u.r0 - file descriptor (& arguments)
37601                                  <1>       ; Outputs: *u.r0 - number of bytes written.
37602                                  <1>       ; ..........................................................
37603                                  <1>       ;
37604                                  <1>       ; Retro UNIX 8086 v1 modification:
37605                                  <1>       ;       'syswrite' system call has three arguments; so,
37606                                  <1>       ;     * 1st argument, file descriptor is in BX register
37607                                  <1>       ;     * 2nd argument, buffer address/offset in CX register
37608                                  <1>       ;     * 3rd argument, number of bytes is in DX register
37609                                  <1>       ;
37610                                  <1>       ;     AX register (will be restored via 'u.r0') will return
37611                                  <1>       ;     to the user with number of bytes written.
37612                                  <1>       ;
37613                                  <1>       ; INPUT ->
37614                                  <1>         ;       EBX = File handle (descriptor/index)
37615                                  <1>       ;         ECX = Buffer address
37616                                  <1>         ;           EDX = Number of bytes
37617                                  <1>       ; OUTPUT ->
37618                                  <1>       ;          EAX = Number of bytes have been written
37619                                  <1>       ;          cf = 1 -> Error code in AL
37620                                  <1>       ;
37621                                  <1>       ; Modified Registers: EAX (at the return of system call)
37622                                  <1>       ;
37623                                  <1>
37624                                  <1>       ; EBX = File descriptor
37625 0000CC5F E89F0E0000               <1>       call   getf1
37626 0000CC64 7274                     <1>       jc     short device_write ; write data to device
37627                                  <1>       ; EAX = First cluster of the file
37628                                  <1>       ; EBX = File number  (Open file number) ; 23/10/2016
37629                                  <1>
37630 0000CC66 E820000000               <1>       call   rw1
37631 0000CC6B 730A                     <1>       jnc    short syswrite_0
```

```
37632 0000CC6D A3[64030600]        <1>       mov    [u.r0], eax ; error code
37633 0000CC72 E918F8FFFF          <1>       jmp    error
37634                              <1>
37635                              <1> syswrite_0:
37636 0000CC77 E8681E0000          <1>       call   writei
37637                              <1> rw0: ; 1:
37638 0000CC7C A1[8C030600]        <1>         mov  eax, [u.nread]
37639 0000CC81 A3[64030600]        <1>       mov    [u.r0], eax
37640 0000CC86 E924F8FFFF          <1>       jmp    sysret
37641                              <1>
37642                              <1> rw1:
37643                              <1>       ; 11/10/2016 (TRDOS 386 = TRDOS v2.0)
37644                              <1>       ; 14/05/2015 (Retro UNIX 386 v1)
37645                              <1>       ; 11/05/2015 (Retro UNIX 386 v1 - Beginning)
37646                              <1>       ; 23/05/2013 - 24/05/2013 (Retro UNIX 8086 v1)
37647                              <1>       ; System call registers: ebx, ecx, edx (through 'sysenter')
37648                              <1>       ;
37649                              <1>       ; EBX = File descriptor
37650                              <1>       ;call  getf1 ; calling point in 'getf' from 'rw1'
37651                              <1>       ;jc    short device_rw ; read/write data from/to device
37652                              <1>       ; EAX = First cluster of the file
37653                              <1>
37654 0000CC8B 83F802              <1>       cmp    eax, 2
37655 0000CC8E 7217                <1>       jb     short rw2
37656                              <1>       ;
37657 0000CC90 890D[84030600]      <1>       mov    [u.base], ecx       ; buffer address/offset
37658                              <1>                                  ;(in the user's virtual memory space)
37659 0000CC96 8915[88030600]      <1>       mov    [u.count], edx
37660                              <1>
37661 0000CC9C C705[C8030600]0000- <1>         mov     dword [u.error], 0 ; reset the last error code
37662 0000CCA4 0000                <1>
37663 0000CCA6 C3                  <1>       retn
37664                              <1>
37665                              <1> rw2:
37666 0000CCA7 B80A000000          <1>       mov    eax, ERR_FILE_NOT_OPEN ; file not open !
37667 0000CCAC A3[C8030600]        <1>       mov    dword [u.error], eax
37668 0000CCB1 C3                  <1>       retn
37669                              <1> rw3:
37670 0000CCB2 B80B000000          <1>       mov    eax, ERR_FILE_ACCESS ; permission denied !
37671 0000CCB7 A3[C8030600]        <1>       mov    dword [u.error], eax
37672 0000CCBC F9                  <1>       stc
37673 0000CCBD C3                  <1>       retn
37674                              <1>
37675                              <1> device_read:
37676                              <1>       ; 11/10/2016 (TRDOS 386 = TRDOS v2.0)
37677                              <1>       ; cl = DEV_OPENMODE ; open mode
37678                              <1>       ; ch = DEV_ACCESS   ; access flags
37679                              <1>       ; al = DEV_DRIVER   ; device number (eax)
37680                              <1>
37681 0000CCBE F6C101              <1>       test   cl, 1 ; 1 = read, 2 = write, 3 = read&write
37682 0000CCC1 74EF                <1>       jz     short rw3
37683                              <1>
37684 0000CCC3 89C3                <1>       mov    ebx, eax
37685 0000CCC5 66C1E302            <1>       shl    bx, 2 ; *4
37686                              <1>
37687 0000CCC9 F6C580              <1>       test   ch, 80h ; bit 7, installable device driver flag
37688 0000CCCC 7406                <1>       jz     short d_read_2 ; Kernel device
37689                              <1>       ; installable device
37690                              <1> d_read_1:
37691 0000CCCE FFA3[405C0100]      <1>         jmp  dword [ebx+IDEV_RADDR-4]
37692                              <1> d_read_2:
37693 0000CCD4 FFA3[140B0100]      <1>       jmp    dword [ebx+KDEV_RADDR-4]
37694                              <1>
37695                              <1> device_write:
37696                              <1>       ; 11/10/2016 (TRDOS 386 = TRDOS v2.0)
37697                              <1>       ; cl = DEV_OPENMODE ; open mode
37698                              <1>       ; ch = DEV_ACCESS   ; access flags
37699                              <1>       ; al = DEV_DRIVER   ; device number (eax)
37700                              <1>
37701 0000CCDA F6C102              <1>       test   cl, 2 ; 1 = read, 2 = write, 3 = read&write
37702 0000CCDD 74D3                <1>       jz     short rw3
37703                              <1>
37704 0000CCDF 89C3                <1>       mov    ebx, eax
37705 0000CCE1 66C1E302            <1>       shl    bx, 2 ; *4
37706                              <1>
37707 0000CCE5 F6C580              <1>       test   ch, 80h ; bit 7, installable device driver flag
37708 0000CCE8 7406                <1>       jz     short d_write_2 ; Kernel device
37709                              <1>       ; installable device
37710                              <1> d_write_1:
37711 0000CCEA FFA3[605C0100]      <1>         jmp  dword [ebx+IDEV_WADDR-4]
37712                              <1> d_write_2:
37713 0000CCF0 FFA3[640B0100]      <1>       jmp    dword [ebx+KDEV_WADDR-4]
37714                              <1>
37715                              <1>
37716                              <1> sysemt: ; enable (or disable) multi tasking -time sharing-
37717                              <1>       ;
37718                              <1>       ; 23/05/2016 - TRDOS 386 (TRDOS v2.0)
37719                              <1>       ; 14/05/2015 (Retro UNIX 386 v1)
37720                              <1>       ; 10/12/2013 - 20/04/2014 (Retro UNIX 8086 v1)
37721                              <1>       ;
37722                              <1>       ; Retro UNIX 8086 v1 modification:
37723                              <1>       ;     'Enable Multi Tasking'  system call instead
37724                              <1>       ;     of 'Emulator Trap' in original UNIX v1 for PDP-11.
37725                              <1>       ;
37726                              <1>       ; Retro UNIX 8086 v1 feature only!
37727                              <1>       ;     Using purpose: Kernel will start without time-out
37728                              <1>       ;     (internal clock/timer) functionality.
37729                              <1>       ;     Then etc/init will enable clock/timer for
37730                              <1>       ;     multi tasking.
37731                              <1>       ;
37732                              <1>       ; INPUT ->
37733                              <1>       ;     BL = 0 -> disable multi tasking
```

```
37734                                  <1>       ;      BL > 1 -> enable multi tasking (time sharing)
37735                                  <1>       ; OUTPUT ->
37736                                  <1>       ;      none
37737                                  <1>       ;
37738                                  <1>       ;  Note: Multi tasking is disabled during system
37739                                  <1>       ;       initialization, it must be enabled by using
37740                                  <1>       ;       this system call. (Otherwise, running proces
37741                                  <1>       ;       will not be changed by another process within
37742                                  <1>       ;       run time sequence/schedule, if running process
37743                                  <1>       ;       will not 'release' itself. Only 'wakeup' procedure
37744                                  <1>       ;       for waiting processes and programmed timer events
37745                                  <1>       ;       for other processes can change running process
37746                                  <1>       ;       while multi tasking is disabled.) ** 23/05/2016 **
37747                                  <1>
37748 0000CCF6 803D[B0030600]00       <1>       cmp    byte [u.uid], 0 ; root ?
37749                                  <1>       ;ja    error
37750 0000CCFD 0F87D1F8FFFF           <1>       ja     badsys ; 14/05/2015
37751                                  <1>       ;
37752 0000CD03 FA                     <1>       cli
37753 0000CD04 881D[5E5B0100]         <1>       mov    [multi_tasking], bl ; 0 to disable, >0 to enable
37754 0000CD0A E9A0F7FFFF             <1>       jmp    sysret
37755                                  <1>
37756                                  <1> systimer:
37757                                  <1>       ; 02/01/2017
37758                                  <1>       ; 21/12/2016
37759                                  <1>       ; 19/12/2016
37760                                  <1>       ; 10/12/2016 (callback)
37761                                  <1>       ; 10/06/2016
37762                                  <1>       ; 07/06/2016
37763                                  <1>       ; 06/06/2016
37764                                  <1>       ; 21/05/2016
37765                                  <1>       ; 19/05/2016
37766                                  <1>       ; 18/05/2016 - TRDOS 386 (TRDOS v2.0)
37767                                  <1>       ; (TRDOS 386 feature only!)
37768                                  <1>       ;
37769                                  <1>       ; (start or stop timer event(s))
37770                                  <1>       ;
37771                                  <1>       ; INPUT ->
37772                                  <1>       ;     BL = Signal return byte (response byte)
37773                                  <1>       ;          (Any requested value between 0 and 255)
37774                                  <1>       ;          (Kernel will put it at the requested address)
37775                                  <1>       ;     BH = Time count unit
37776                                  <1>       ;          0 = Stop timer event
37777                                  <1>       ;          1 = 18.2 ticks per second
37778                                  <1>       ;          2 = 10 milliseconds
37779                                  <1>       ;          3 = 1 second (for real time clock interrupt)
37780                                  <1>       ;          4 = time/tick count in current time count unit
37781                                  <1>       ;         // 10/12/2016
37782                                  <1>       ;         80h = Stop timer event (callback method)
37783                                  <1>       ;         81h = 18.2 ticks per second, callback method
37784                                  <1>       ;         82h = 10 milliseconds, callback method
37785                                  <1>       ;         83h = 1 second (for RTC int), callback method
37786                                  <1>       ;         84h = current time count unit, callback method
37787                                  <1>       ;
37788                                  <1>       ;         Note: Only 03h or 83h will set real time clock
37789                                  <1>       ;             (RTC) events (Others are for PIT events)!
37790                                  <1>       ;
37791                                  <1>       ;     NOTE: If callback (user service) method is used,
37792                                  <1>       ;         EDX will point to the return address (of service
37793                                  <1>       ;         procedure) in user's space instead of signal
37794                                  <1>       ;         response byte address. (TRDOS 386 kernel will
37795                                  <1>       ;         direct the cpu to that address -in user's space-
37796                                  <1>       ;         at the return of system call or interrupt
37797                                  <1>       ;         just after the adjusted count/time is elapsed.)
37798                                  <1>       ;         User's sevice routine must be ended with a
37799                                  <1>       ;         'iret'. Normal return addresses from system
37800                                  <1>       ;         calls or and interrupts will be kept same except
37801                                  <1>       ;         the timer returns.
37802                                  <1>       ;
37803                                  <1>       ;     BH = 0 -> Stop timer event
37804                                  <1>       ;     BL = Timer event number (1 to 255) if BH = 0
37805                                  <1>       ;          If BL = 0, all timer events (which are belongs
37806                                  <1>       ;           to running process) will be stopped
37807                                  <1>       ;     ECX = Time/Tick count (depending on time count unit)
37808                                  <1>       ;     EDX = Signal return (Response) byte address
37809                                  <1>       ;           (virtual address in user's memory space)
37810                                  <1>       ; OUTPUT ->
37811                                  <1>       ;     AL = Timer event number   (1 to 255) (max. value = 16)
37812                                  <1>       ;     IF BH Input = 0 & CF = 0 & AL = 0 ->
37813                                  <1>       ;          timer event(s) has/have been stopped/finished
37814                                  <1>       ;     CF = 1 & AL = 0 -> no timer setting space to set
37815                                  <1>       ;     CF = 1 & AL > 0 -> timer count unit is not usable
37816                                  <1>       ;
37817                                  <1>       ;     NOTE: To modify a time count for a user function,
37818                                  <1>       ;           at first, current timer event must be stopped
37819                                  <1>       ;           then a new timer event (which is related with
37820                                  <1>       ;           same user function) must be started.
37821                                  <1>       ;
37822                                  <1>       ;           Signal return (response) byte may be used for
37823                                  <1>       ;           several purposes. Kernel will put this value
37824                                  <1>       ;           to requested address during timer interrupt,
37825                                  <1>       ;           program/user can check this value to understand
37826                                  <1>       ;           which event has been occurred and what is changed.
37827                                  <1>       ;           (Multi timer events can share same signal address)
37828                                  <1>       ;
37829                                  <1>       ;     NOTE: If the process is running while the time count
37830                                  <1>       ;           is reached, kernel will put signal return (response)
37831                                  <1>       ;           byte value at requested address during timer
37832                                  <1>       ;           interrupt and the process will continue to run.
37833                                  <1>       ;           Program/process must call (jump to) it's timer event
37834                                  <1>       ;           function as required, for checking the timer event
37835                                  <1>       ;           status via signal return (response) byte address.
```

```
37836                                  <1>       ;
37837                                  <1>       ;           If the process is not running (waiting or sleeping
37838                                  <1>       ;           or released) while the time count is reached,
37839                                  <1>       ;           it is restarted from where it left, to ensure
37840                                  <1>       ;           proper multi media (video, audio, clock, timer)
37841                                  <1>       ;           functionality.
37842                                  <1>       ;
37843                                  <1>       ;           (It is better to use 'syswait' or 'syssleep',
37844                                  <1>       ;           or 'sysrele' system call just after the timer
37845                                  <1>       ;           function. Otherwise, timer events may block other
37846                                  <1>       ;           processes which are not using timer events.)
37847                                  <1>       ;
37848                                  <1>       ; Timer Event Structure: (max. 16 timer events, 16*16 bytes)
37849                                  <1>       ;       Owner:              resb 1 ; 0 = free
37850                                  <1>       ;                          ;>0 = process number (u.uno)
37851                                  <1>       ;     Calback:     resb 1 ; 1 = callback, 0 = response byte
37852                                  <1>       ;     Interrupt:      resb 1 ; 0 = Timer interrupt (or none)
37853                                  <1>       ;                          ; 1 = Real Time Clock interrupt
37854                                  <1>       ;     Response:       resb 1 ; 0 to 255, signal return value
37855                                  <1>       ;     Count Limit: resd 1 ; count of ticks (total/set)
37856                                  <1>       ;     Current Count:      resd 1 ; count of ticks (current)
37857                                  <1>       ;     Response Addr:  resd 1 ; response byte (pointer) address
37858                                  <1>       ;
37859                                  <1>
37860                                  <1>       ; 19/12/2016 (timer callback)
37861 0000CD0F C605[9C600100]00        <1>       mov    byte [tcallback], 0
37862 0000CD16 C605[9D600100]00        <1>       mov    byte [trtc], 0
37863 0000CD1D C705[D0030600]0000-     <1>       mov    dword [u.tcb], 0 ; this is not necessary...
37864 0000CD25 0000                    <1>
37865                                  <1>
37866 0000CD27 80FF80                  <1>       cmp    bh, 80h
37867 0000CD2A 7225                    <1>       jb     short systimer_cb2
37868 0000CD2C 7704                    <1>       ja     short systimer_cb0
37869                                  <1>
37870 0000CD2E 31D2                    <1>       xor    edx, edx ; 0, reset callback address
37871 0000CD30 EB0B                    <1>       jmp    short systimer_cb1
37872                                  <1>
37873                                  <1> systimer_cb0:
37874 0000CD32 80FF84                  <1>       cmp    bh, 84h
37875 0000CD35 7764                    <1>       ja     short systimer_5 ;  undefined, error
37876                                  <1>
37877                                  <1>       ;mov   byte [tcallback], 1 ; 19/12/2016
37878 0000CD37 FE05[9C600100]          <1>       inc    byte [tcallback]
37879                                  <1>
37880                                  <1> systimer_cb1:
37881 0000CD3D 0FB635[B3030600]        <1>       movzx  esi, byte [u.uno] ; process number
37882 0000CD44 66C1E602                <1>       shl    si, 2
37883 0000CD48 8996[0C010600]          <1>       mov    [esi+p.tcb-4], edx ; set process timer callback address
37884                                  <1>                               ; (overwrite prev value if it is set!)
37885 0000CD4E 80E77F                  <1>       and    bh, 7Fh
37886                                  <1>
37887                                  <1> systimer_cb2:
37888 0000CD51 80FF02                  <1>       cmp    bh, 2
37889 0000CD54 7445                    <1>         je     short systimer_5  ; only 18.2 ticks per second is usable
37890                                  <1>                               ; 10 milliseconds (100 Hertz) timer
37891                                  <1>                               ; will be set later (18/05/2016)
37892 0000CD56 774B                    <1>         ja     short systimer_6
37893                                  <1>
37894 0000CD58 20FF                    <1>       and    bh, bh
37895 0000CD5A 0F84BA000000            <1>         jz     systimer_9        ; stop timer event(s)
37896                                  <1>
37897                                  <1>       ; bh = 1 (timer interrupt, 18.2 Hz, IBM PC/AT ROMBIOS default)
37898                                  <1>
37899                                  <1> systimer_19:
37900 0000CD60 B00A                    <1>       mov    al, 10 ; (*)
37901                                  <1>
37902                                  <1> systimer_0:
37903 0000CD62 B710                    <1>       mov    bh, 16
37904                                  <1>       ;
37905 0000CD64 383D[5F5B0100]          <1>       cmp    [timer_events], bh ; 16 ; 07/06/2016
37906 0000CD6A 7319                    <1>       jnb    short systimer_3  ; max. 16 timer events
37907                                  <1>       ;
37908 0000CD6C 50                      <1>       push   eax ; (*)
37909                                  <1>
37910 0000CD6D BF[60040600]            <1>       mov    edi, timer_set  ; beginning address of timer events
37911                                  <1>                               ; setting space
37912 0000CD72 30C0                    <1>       xor    al, al ; 0
37913                                  <1> systimer_1:
37914 0000CD74 FEC0                    <1>       inc    al
37915 0000CD76 803F00                  <1>       cmp    byte [edi], 0       ; is it free space ?
37916 0000CD79 7639                    <1>       jna    short systimer_7 ; yes
37917 0000CD7B FECF                    <1>       dec    bh
37918 0000CD7D 7405                    <1>       jz     short systimer_2
37919 0000CD7F 83C710                  <1>       add    edi, 16
37920 0000CD82 EBF0                    <1>       jmp    short  systimer_1 ; next event space
37921                                  <1>
37922                                  <1> systimer_2:
37923 0000CD84 58                      <1>       pop    eax ; (*) discard
37924                                  <1> systimer_3:
37925 0000CD85 C605[64030600]00        <1>       mov    byte [u.r0], 0
37926                                  <1> systimer_4:
37927 0000CD8C C705[C8030600]1B00-     <1>         mov     dword [u.error], ERR_MISC
37928 0000CD94 0000                    <1>
37929                                  <1>                                   ; one of miscellaneous/other errors
37930 0000CD96 E9F4F6FFFF              <1>       jmp    error ; cf -> 1
37931                                  <1>
37932                                  <1> systimer_5:
37933 0000CD9B 883D[64030600]          <1>       mov    [u.r0], bh ; Time count unit (=2 or >3)
37934 0000CDA1 EBE9                    <1>       jmp    short systimer_4 ; 07/06/2016
37935                                  <1>
37936                                  <1> systimer_6:
```

```
37937 0000CDA3 80FF04              <1>       cmp    bh, 4
37938 0000CDA6 77F3                <1>         ja      short systimer_5  ; undefined time count unit
37939                              <1>       ;jb    short systimer_16
37940                              <1>
37941                              <1>       ;mov   al, 1  ; default (use current timer unit)
37942                              <1>                     ; countdown value is in ECX !
37943                              <1>                     ; max. value of ecx = 4294967296/10
37944                              <1>         ;jmp    short systimer_0
37945                              <1>       ;jmp   short systimer_19
37946 0000CDA8 74B6                <1>       je     short systimer_19
37947                              <1>
37948                              <1> systimer_16:
37949                              <1>       ; bh = 3
37950                              <1>       ; timer event via real time clock interrupt
37951                              <1>       ; interrupt/update frequency: 1 Hz (1 tick per second)
37952                              <1>
37953 0000CDAA B0B6                <1>       mov    al, 182 ; (*) ; 18.2 * 10
37954 0000CDAC FE05[9D600100]      <1>       inc    byte [trtc] ; timer event via real time clock
37955 0000CDB2 EBAE                <1>         jmp     short systimer_0
37956                              <1>
37957                              <1> systimer_7:
37958 0000CDB4 A2[64030600]        <1>       mov    [u.r0], al ; timer event number
37959                              <1>       ;
37960                              <1>       ; edi = address of empty timer event area
37961 0000CDB9 A0[B3030600]        <1>       mov    al, [u.uno]
37962 0000CDBE FA                  <1>       cli    ; disable interrupts
37963 0000CDBF AA                  <1>       stosb  ; process number
37964 0000CDC0 A0[9C600100]        <1>       mov    al, [tcallback] ; timer callback flag
37965 0000CDC5 AA                  <1>       stosb  ; 1= callback method, 0= signal response byte method
37966 0000CDC6 A0[9D600100]        <1>       mov    al, [trtc] ; timer interrupt type
37967 0000CDCB AA                  <1>       stosb  ; 1= real time clock, 0= programmable interval timer
37968 0000CDCC 88D8                <1>       mov    al, bl ; Signal return (Response) value
37969 0000CDCE AA                  <1>       stosb   ; response byte
37970 0000CDCF 58                  <1>       pop    eax ; (*) ; 10 or 182
37971 0000CDD0 89D3                <1>       mov    ebx, edx ; virtual address for response/signal byte
37972 0000CDD2 F7E1                <1>       mul    ecx
37973                              <1>       ; (eax = 10 * count of 18.2 Hz timer ticks)
37974                              <1>       ; (count down step = 10)
37975 0000CDD4 AB                  <1>       stosd  ; count limit (reset value)
37976 0000CDD5 AB                  <1>       stosd  ; current count value
37977                              <1>
37978                              <1>       ; 19/12/2016
37979 0000CDD6 803D[9C600100]00    <1>       cmp    byte [tcallback], 0 ; timer callback method ?
37980 0000CDDD 7604                <1>       jna    short systimer_17 ; no
37981 0000CDDF 89D8                <1>       mov    eax, ebx ; virtual address for callback routine
37982 0000CDE1 EB0D                <1>       jmp    short systimer_18
37983                              <1>
37984                              <1> systimer_17: ; signal response byte method
37985                              <1>       ; ebx = virtual address
37986                              <1>       ; [u.pgdir] = page directory's physical address
37987                              <1>       ; 20/02/2017
37988 0000CDE3 FE05[9E600100]      <1>       inc     byte [no_page_swap] ; 1
37989                              <1>                     ; Do not add this page to swap queue
37990                              <1>                     ; and remove it from swap queue if it is
37991                              <1>                     ; on the queue.
37992 0000CDE9 E8A084FFFF          <1>       call   get_physical_addr
37993 0000CDEE 721A                <1>       jc     short systimer_8 ; 07/06/2016
37994                              <1>       ; eax = physical address of the virtual address in user's space
37995                              <1> systimer_18:
37996 0000CDF0 AB                  <1>       stosd  ; response addr (physical) or callback addr (virtual)
37997 0000CDF1 FE05[5F5B0100]      <1>       inc    byte [timer_events] ; 07/06/201
37998                              <1>       ; 02/01/2017
37999 0000CDF7 0FB605[B3030600]    <1>       movzx  eax, byte [u.uno]
38000 0000CDFE FE80[FF000600]      <1>       inc    byte [eax+p.timer-1]
38001                              <1>       ;
38002 0000CE04 FB                  <1>       sti    ; enable interrupts
38003 0000CE05 E9A5F6FFFF          <1>       jmp    sysret
38004                              <1>
38005                              <1> systimer_8:
38006                              <1>       ; 10/06/2016
38007                              <1>       ; 07/06/2016
38008 0000CE0A 28C0                <1>       sub    al, al ; 0
38009 0000CE0C 8847F4              <1>       mov    [edi-12], al ; clear process number (free timer event)
38010                              <1>       ;mov   dword [edi], eax ; 0
38011 0000CE0F FB                  <1>       sti
38012 0000CE10 A2[64030600]        <1>       mov    [u.r0], al ; 0
38013 0000CE15 E975F6FFFF          <1>       jmp    error
38014                              <1>
38015                              <1> systimer_9:
38016                              <1>       ; 10/06/2016
38017                              <1>       ; 07/06/2016
38018 0000CE1A 28C0                <1>       sub    al, al
38019 0000CE1C A2[64030600]        <1>       mov    byte [u.r0], al ; 0
38020 0000CE21 3805[5F5B0100]      <1>       cmp     byte [timer_events], al ;  0
38021 0000CE27 7631                <1>       jna    short systimer_12
38022                              <1>
38023                              <1>       ; Note: ecx and edx are undefined here
38024                              <1>       ;      (for stop timer function)
38025                              <1>
38026 0000CE29 BE[60040600]        <1>       mov    esi, timer_set  ; beginning address of timer events
38027                              <1>                               ; setting space
38028 0000CE2E A0[B3030600]        <1>       mov    al, [u.uno]
38029                              <1>
38030 0000CE33 B710                <1>       mov    bh, 16
38031                              <1>
38032 0000CE35 08DB                <1>       or     bl, bl
38033 0000CE37 7544                <1>       jnz    short systimer_15
38034                              <1>
38035                              <1>       ; clear timer event areas belong to current process
38036                              <1>       ; (for stopping all timer events belong to current process)
38037 0000CE39 FA                  <1>       cli    ; disable interrupts
38038                              <1> systimer_10:
```

```
38039                                  <1>       ; 10/06/2016
38040                                  <1>       ; 07/06/2016
38041 0000CE3A 8A26                    <1>       mov    ah, [esi]
38042 0000CE3C 08E4                    <1>       or     ah, ah ; 0 ?
38043 0000CE3E 7411                    <1>       jz     short systimer_11
38044 0000CE40 38C4                    <1>       cmp    ah, al ; is the process number (owner) same ?
38045 0000CE42 750D                    <1>         jne     short systimer_11 ; no
38046                                  <1>
38047                                  <1>       ;mov   byte [esi], 0
38048 0000CE44 66C7060000              <1>       mov    word [esi], 0 ; clear
38049                                  <1>       ;mov   dword [esi+12], 0 ; clear
38050                                  <1>
38051 0000CE49 FE0D[5F5B0100]          <1>       dec    byte [timer_events]
38052 0000CE4F 7409                    <1>       jz     short systimer_12
38053                                  <1>
38054                                  <1> systimer_11:
38055 0000CE51 FECF                    <1>       dec    bh
38056 0000CE53 7405                    <1>       jz     short systimer_12
38057 0000CE55 83C610                  <1>       add    esi, 16
38058 0000CE58 EBE0                    <1>       jmp    short systimer_10
38059                                  <1>
38060                                  <1> systimer_12:
38061 0000CE5A 0FB635[B3030600]        <1>       movzx  esi, byte [u.uno]
38062 0000CE61 08DB                    <1>       or     bl, bl ; all timer events or one timer event ?
38063 0000CE63 740C                    <1>       jz     short systimer_13
38064 0000CE65 8A9E[FF000600]          <1>       mov    bl, [esi+p.timer-1]
38065 0000CE6B 20DB                    <1>       and    bl, bl ; previous number of timer events for the process
38066 0000CE6D 7408                    <1>       jz     short systimer_14
38067 0000CE6F FECB                    <1>       dec    bl  ; previous number of timer events for the process - 1
38068                                  <1> systimer_13:
38069 0000CE71 889E[FF000600]          <1>       mov    [esi+p.timer-1], bl ; 0 ; no timer events for process
38070                                  <1> systimer_14:
38071 0000CE77 FB                      <1>       sti    ; enable interrupts
38072 0000CE78 E932F6FFFF              <1>       jmp    sysret
38073                                  <1>
38074                                  <1> systimer_15:
38075 0000CE7D 38FB                    <1>       cmp    bl, bh ; 16
38076 0000CE7F 0F8707FFFFFF            <1>         ja      systimer_4      ; max. 16 timer events !
38077                                  <1>       ;
38078 0000CE85 88DA                    <1>       mov    dl, bl
38079 0000CE87 FECA                    <1>       dec    dl  ; 16 -> 15 ... 1 -> 0
38080 0000CE89 C0E204                  <1>       shl    dl, 4 ; * 16
38081 0000CE8C 0FB6FA                  <1>       movzx  edi, dl
38082 0000CE8F 01F7                    <1>       add    edi, esi ; timer_set
38083                                  <1>
38084 0000CE91 3A07                    <1>       cmp    al, [edi] ; process number
38085 0000CE93 0F85F3FEFFFF            <1>         jne     systimer_4
38086                                  <1>
38087                                  <1>       ; same process ID
38088 0000CE99 FA                      <1>       cli    ; disable interrupts
38089                                  <1>       ; 10/06/2016 ; 02/01/2017
38090                                  <1>       ;mov   byte [edi], 0
38091 0000CE9A 66C7070000              <1>       mov    word [edi], 0 ; clear
38092                                  <1>       ;mov   dword [edi+12], 0 ; clear
38093 0000CE9F FE0D[5F5B0100]          <1>       dec    byte [timer_events]
38094 0000CEA5 EBB3                    <1>       jmp    short systimer_12
38095                                  <1>
38096                                  <1> sysmdate: ; < change the modification time of a file >
38097                                  <1>       ; 13/01/2017 - TRDOS 386 (TRDOS v2.0)
38098                                  <1>       ; temporary !
38099 0000CEA7 B801000000              <1>       mov    eax, ERR_INV_FNUMBER ; 'invalid function number !'
38100 0000CEAC A3[C8030600]            <1>         mov     [u.error], eax
38101 0000CEB1 A3[64030600]            <1>         mov     [u.r0], eax
38102 0000CEB6 E9D4F5FFFF              <1>       jmp    error
38103                                  <1>
38104                                  <1> sysvideo: ; VIDEO DATA TRANSFER FUNCTIONS
38105                                  <1>       ; 12/05/2017
38106                                  <1>       ; 11/07/2016
38107                                  <1>       ; 13/06/2016
38108                                  <1>       ; 16/05/2016 - TRDOS 386 (TRDOS v2.0)
38109                                  <1>       ;
38110                                  <1>       ;
38111                                  <1>       ; VIDEO DATA TRANSFER FUNCTIONS:
38112                                  <1>       ;
38113                                  <1>       ; Inputs:
38114                                  <1>       ;     BH = 0 = VIDEO BIOS Mode 3, tty/text mode data transfers
38115                                  <1>       ;          BL =
38116                                  <1>       ;            Bits 0&1, Transfer direction
38117                                  <1>       ;                  0 - System to system
38118                                  <1>       ;                  1 - User to system
38119                                  <1>       ;                  2 - System to user
38120                                  <1>       ;                  3 - User to user
38121                                  <1>       ;            Bits 2&3, Transfer Type
38122                                  <1>       ;                  0 - Display page transfer
38123                                  <1>       ;                  1 - Display page window transfer
38124                                  <1>       ;                  2 - Frame/Viewport/Window address transfer
38125                                  <1>       ;                  3 - Window handle transfer
38126                                  <1>       ;
38127                                  <1>       ;          /// BL = 0 -> System to system (display page) transfer
38128                                  <1>       ;             CL = Source page
38129                                  <1>       ;             DL = Destination page
38130                                  <1>       ;          /// BL = 1&2 -> user to system & system to user transfer
38131                                  <1>       ;             ECX = User buffer
38132                                  <1>       ;             DL = Video page
38133                                  <1>       ;          /// BL = 5&6 -> user to system, system to user transfer
38134                                  <1>       ;            (window in current display page and in current mode)
38135                                  <1>       ;             ESI = User's buffer address
38136                                  <1>       ;             ECX Low 16 bits = Top left column (X1 position)
38137                                  <1>       ;             ECX High 16 bits = Top row (Y1 position)
38138                                  <1>       ;             EDX Low 16 bits = Bottom right column (X2 position)
38139                                  <1>       ;             EDX High 16 bits = Bottom row (Y2 position)
38140                                  <1>         ;               If BL = 5 ->
```

```
38141                                  <1>       ;             EDI = Swap address (in user's memory space)
38142                                  <1>       ;             (If swap address > 0, previous content of the window
38143                                  <1>       ;             will be saved into swap area in user's memory space)
38144                                  <1>       ;           /// BL = 4 -> system to system transfer
38145                                  <1>       ;             ESI = System's source buffer (video page) address
38146                                  <1>       ;             ECX Low 16 bits = Top left column (X1 position)
38147                                  <1>       ;             ECX High 16 bits = Top row (Y1 position)
38148                                  <1>       ;             EDX Low 16 bits = Bottom right column (X2 position)
38149                                  <1>       ;             EDX High 16 bits = Bottom row (Y2 position)
38150                                  <1>       ;             EDI = System's destination buffer (video page) address
38151                                  <1>       ;
38152                                  <1>       ;     BH = 1 = CGA Graphics (0B8000h) data transfers
38153                                  <1>       ;          BL =
38154                                  <1>       ;            0 = Fill color (color in CL] (32K)
38155                                  <1>       ;            1 = User to system display page transfer
38156                                  <1>       ;            2 = System to user display page transfer
38157                                  <1>       ;            3 = NOT bits in window (ECX, EDX)
38158                                  <1>       ;            4 = Window copy (system to system)
38159                                  <1>       ;            5 = User to system window transfer
38160                                  <1>       ;            6 = System to user window transfer
38161                                  <1>       ;            7 = AND display page bytes with CL
38162                                  <1>       ;            8 = OR display page bytes with CL
38163                                  <1>       ;            9 = XOR display page bytes with CL
38164                                  <1>       ;
38165                                  <1>       ;           /// BL = 0 -> Fill color  (all screen pixels)
38166                                  <1>       ;             CL = Color value
38167                                  <1>       ;           /// BL = 1&2 -> user to system & system to user transfer
38168                                  <1>       ;             ECX = User buffer
38169                                  <1>       ;           /// BL = 5&6 -> user to system, system to user transfer
38170                                  <1>       ;            (window in current display page and in current mode)
38171                                  <1>       ;             ESI = User's buffer address
38172                                  <1>       ;             ECX Low 16 bits = Top left column (X1 position)
38173                                  <1>       ;             ECX High 16 bits = Top row (Y1 position)
38174                                  <1>       ;             EDX Low 16 bits = Bottom right column (X2 position)
38175                                  <1>       ;             EDX High 16 bits = Bottom row (Y2 position)
38176                                  <1>         ;           /// BL = 4 -> system to system (window) transfer
38177                                  <1>       ;             ESI = System's source buffer (video page) address
38178                                  <1>       ;             ECX Low 16 bits = Top left column (X1 position)
38179                                  <1>       ;             ECX High 16 bits = Top row (Y1 position)
38180                                  <1>       ;             EDX Low 16 bits = Bottom right column (X2 position)
38181                                  <1>       ;             EDX High 16 bits = Bottom row (Y2 position)
38182                                  <1>       ;             EDI = System's destination buffer (video page) address
38183                                  <1>       ;           /// BL = 3 -> NOT byte in display page/memory
38184                                  <1>       ;             ECX Low 16 bits = Top left column (X1 position)
38185                                  <1>       ;             ECX High 16 bits = Top row (Y1 position)
38186                                  <1>       ;             EDX Low 16 bits = Bottom right column (X2 position)
38187                                  <1>       ;             EDX High 16 bits = Bottom row (Y2 position)
38188                                  <1>       ;
38189                                  <1>       ;     BH = 2 = VGA Graphics (0A0000h) data transfers
38190                                  <1>       ;          BL =
38191                                  <1>       ;            x0h = Fill color (color in CL] (64K)
38192                                  <1>       ;            x1h = User to system display page transfer
38193                                  <1>       ;            x2h = System to user display page transfer
38194                                  <1>       ;            x3h = NOT bits in window (ECX, EDX)
38195                                  <1>       ;            x4h = Window copy (system to system)
38196                                  <1>       ;            x5h = User to system window transfer
38197                                  <1>       ;            x6h = System to user window transfer
38198                                  <1>       ;            x7h = AND display page bytes with CL
38199                                  <1>       ;            x8h = OR display page bytes with CL
38200                                  <1>       ;            x9h = XOR display page bytes with CL
38201                                  <1>       ;            x = 0 -> screen width = 320
38202                                  <1>       ;            x = 1 -> screen width = 640
38203                                  <1>       ;            x = 2 -> screen width = 800
38204                                  <1>       ;
38205                                  <1>       ;           /// BL = 0 -> Fill color  (all screen pixels)
38206                                  <1>       ;             CL = Color value
38207                                  <1>       ;           /// BL = 1&2 -> user to system & system to user transfer
38208                                  <1>       ;             ECX = User buffer
38209                                  <1>       ;           /// BL = 5&6 -> user to system, system to user transfer
38210                                  <1>       ;            (window in current display page and in current mode)
38211                                  <1>       ;             ESI = User's buffer address
38212                                  <1>       ;             ECX Low 16 bits = Top left column (X1 position)
38213                                  <1>       ;             ECX High 16 bits = Top row (Y1 position)
38214                                  <1>       ;             EDX Low 16 bits = Bottom right column (X2 position)
38215                                  <1>       ;             EDX High 16 bits = Bottom row (Y2 position)
38216                                  <1>             ;           /// BL = 4 -> system to system (window) transfer
38217                                  <1>       ;             ESI = System's source buffer (video page) address
38218                                  <1>       ;             ECX Low 16 bits = Top left column (X1 position)
38219                                  <1>       ;             ECX High 16 bits = Top row (Y1 position)
38220                                  <1>       ;             EDX Low 16 bits = Bottom right column (X2 position)
38221                                  <1>       ;             EDX High 16 bits = Bottom row (Y2 position)
38222                                  <1>       ;             EDI = System's destination buffer (video page) address
38223                                  <1>       ;           /// BL = 3 -> NOT byte in display page/memory
38224                                  <1>       ;             ECX Low 16 bits = Top left column (X1 position)
38225                                  <1>       ;             ECX High 16 bits = Top row (Y1 position)
38226                                  <1>       ;             EDX Low 16 bits = Bottom right column (X2 position)
38227                                  <1>       ;             EDX High 16 bits = Bottom row (Y2 position)
38228                                  <1>       ;
38229                                  <1>       ;     BH = 3 = Super VGA, LINEAR FRAME BUFFER data transfers
38230                                  <1>       ;          BL =
38231                                  <1>       ;            0 = Fill color (color in ECX] (Frame buffer size)
38232                                  <1>       ;            1 = User to system display page transfer
38233                                  <1>       ;            2 = System to user display page transfer
38234                                  <1>       ;            3 = NOT bits in window (ECX, EDX)
38235                                  <1>       ;            4 = Window copy (system to system)
38236                                  <1>       ;            5 = User to system window transfer
38237                                  <1>       ;            6 = System to user window transfer
38238                                  <1>       ;            7 = AND display page bytes with ECX
38239                                  <1>       ;            8 = OR display page bytes with ECX
38240                                  <1>       ;            9 = XOR display page bytes with ECX
38241                                  <1>       ;
38242                                  <1>       ;           /// BL = 0 -> Fill color  (all screen pixels)
```

```
38243                                  <1>       ;            CL = Color value
38244                                  <1>       ;          /// BL = 1&2 -> user to system & system to user transfer
38245                                  <1>       ;            ECX = User buffer
38246                                  <1>       ;          /// BL = 5&6 -> user to system, system to user transfer
38247                                  <1>       ;            (window in current display page and in current mode)
38248                                  <1>       ;            ESI = User's buffer address
38249                                  <1>       ;            ECX Low 16 bits = Top left column (X1 position)
38250                                  <1>       ;            ECX High 16 bits = Top row (Y1 position)
38251                                  <1>       ;            EDX Low 16 bits = Bottom right column (X2 position)
38252                                  <1>       ;            EDX High 16 bits = Bottom row (Y2 position)
38253                                  <1>       ;          /// BL = 4 -> system to system (window) transfer
38254                                  <1>       ;            ESI = System's source buffer (video page) address
38255                                  <1>       ;            ECX Low 16 bits = Top left column (X1 position)
38256                                  <1>       ;            ECX High 16 bits = Top row (Y1 position)
38257                                  <1>       ;            EDX Low 16 bits = Bottom right column (X2 position)
38258                                  <1>       ;            EDX High 16 bits = Bottom row (Y2 position)
38259                                  <1>       ;            EDI = System's destination buffer (video page) address
38260                                  <1>       ;          /// BL = 3 -> NOT byte in display page/memory
38261                                  <1>       ;            ECX Low 16 bits = Top left column (X1 position)
38262                                  <1>       ;            ECX High 16 bits = Top row (Y1 position)
38263                                  <1>       ;            EDX Low 16 bits = Bottom right column (X2 position)
38264                                  <1>       ;            EDX High 16 bits = Bottom row (Y2 position)
38265                                  <1>       ;
38266                                  <1>       ; Outputs:
38267                                  <1>       ;     EAX = transfer/byte count
38268                                  <1>       ;
38269                                  <1>       ;     NOTE: If the source or destination address passes out of
38270                                  <1>       ;     video pages (display memory limits), data will not be transferred
38271                                  <1>       ;     and EAX will return as 0.
38272                                  <1>       ;
38273                                  <1>       ;
38274                                  <1>       ; DIRECT (STANDARD VGA/CGA) DISPLAY MEMORY ACCESS FUNCTIONS:
38275                                  <1>       ;
38276                                  <1>       ;     BH = 4 = CGA direct video memory (0B8000h, 32K) access
38277                                  <1>       ;            Page directory & page tables of the user's
38278                                  <1>       ;            program will be updated to direct access to
38279                                  <1>       ;            0B8000h (32K) video (CGA, color) memory; if
38280                                  <1>       ;            there is not a permission  conflict or lock!
38281                                  <1>       ;             (User's program/process will have permision to
38282                                  <1>       ;            access locked display memory if the owner is
38283                                  <1>       ;            it's parent.)
38284                                  <1>         ;
38285                                  <1>       ;         Screen width = 320
38286                                  <1>       ;
38287                                  <1>       ;     BH = 5 = VGA direct video memory (0A0000h, 64K) access
38288                                  <1>       ;            Page directory & page tables of the user's
38289                                  <1>       ;            program will be updated to direct access to
38290                                  <1>       ;            0A0000h (64K) video (VGA) memory; if there is not
38291                                  <1>       ;            a permission conflict or lock!
38292                                  <1>       ;             (User's program/process will have permision to
38293                                  <1>       ;            access locked display memory if the owner is
38294                                  <1>       ;            it's parent.)
38295                                  <1>       ;
38296                                  <1>       ;         BL = Screen width (320, 640, 800)
38297                                  <1>       ;
38298                                  <1>       ; Outputs:
38299                                  <1>       ;     EAX = Display mmory address for direct access
38300                                  <1>       ;           0A0000h for VGA, 0B8000h for CGA
38301                                  <1>       ;     (Display memory size: 32K for CGA, 64K for VGA)
38302                                  <1>       ;     EAX = 0 if display page access permission has been denied.
38303                                  <1>       ;           (Locked!)
38304                                  <1>       ;
38305                                  <1>       ; LINEAR FRAME BUFFER ACCESS FUNCTIONS:
38306                                  <1>       ;
38307                                  <1>       ;     BH = 6 = Linear Frame Buffer direct video memory access
38308                                  <1>       ;
38309                                  <1>       ;            Page directory & page tables of the user's
38310                                  <1>       ;            program will be updated to direct access to
38311                                  <1>       ;            the configured LFB (Linear Frame Buffer) address,
38312                                  <1>       ;            if there is not a permission conflict or lock!
38313                                  <1>       ;             (User's program/process will have permision to
38314                                  <1>       ;            access locked display memory if the owner is
38315                                  <1>       ;            it's parent.)
38316                                  <1>       ;
38317                                  <1>       ;            Return: EAX = Linear Frame Buffer address
38318                                  <1>       ;                  EDX = Frame Buffer Size in bytes
38319                                  <1>       ;
38320                                  <1>       ;     BH = 7 = Get Linear Frame Buffer info (for current mode)
38321                                  <1>       ;
38322                                  <1>       ;            Return:
38323                                  <1>       ;            EAX = Frame Buffer Address (0 = is not in use)
38324                                  <1>       ;            EDX = Frame Buffer Size in bytes
38325                                  <1>       ;            BL = Current Video Mode
38326                                  <1>       ;                 BL = 0FFh -> Super VGA (Extended VGA)
38327                                  <1>       ;                 If BL = 0FFh,
38328                                  <1>         ;                       BH = 0 = 16 colors
38329                                  <1>       ;                  BH = 1 = 256 colors
38330                                  <1>       ;                  BH = 2 = 66536 colors
38331                                  <1>       ;                  BH = 3 = 24 bits TRUE (16M) colors
38332                                  <1>       ;                  BH = 4 = 32 bits TRUE (16M) colors
38333                                  <1>       ;            ECX = Pixel resolution
38334                                  <1>       ;                  CX = Width (640, 800, 1024, 1366, 1920)
38335                                  <1>       ;                  High 16 bits of ECX = Height
38336                                  <1>       ;
38337                                  <1>       ;     NOTE: Each process will have it's own frame buffer
38338                                  <1>       ;           address and resolution parameters in 'u' area.
38339                                  <1>       ;           Then, if the current frame buffer & resolution
38340                                  <1>       ;           is different, frame buffer r/w functions
38341                                  <1>       ;           will use scale factor to convert process's
38342                                  <1>         ;             pixel coordinates to actual screen coordinates.
38343                                  <1>       ;           resolution -> dimensional scale
38344                                  <1>       ;           color size -> color scale
```

```
38345                                  <1>       ;          * RGB (TRUE) colors to 256 colors conversion:
38346                                  <1>         ;            TRUE Colors -> 8,8,8 (R,G,B; byte 0 is R)
38347                                  <1>       ;           256 colors -> 2,2,2,2 (R,G,B,L; bit 0&1 is R)
38348                                  <1>       ;              bit 6&7 -> luminosity base level (0,1,2,3)
38349                                  <1>       ;              bit 4&5 -> blue level (0,1,2,3)
38350                                  <1>       ;              bit 2%3 -> green level (0,1,2,3)
38351                                  <1>       ;              bit 0&1 -> red level (0,1,2,3)
38352                                  <1>       ;           Example: total red level : luminosity + red level
38353                                  <1>       ;           Luminosity base level: 0 -> 16
38354                                  <1>       ;                                  1 -> 32
38355                                  <1>       ;                                  2 -> 64
38356                                  <1>       ;                                  3 -> 128
38357                                  <1>       ;           Color level:
38358                                  <1>       ;                                 0 -> 0
38359                                  <1>       ;                                 1 -> luminosity level
38360                                  <1>       ;                                 2 -> luminosity level + 64
38361                                  <1>       ;                                 3 -> 255
38362                                  <1>       ;          Luminosity base level = min (R,G,B)
38363                                  <1>       ;                  if it is <16, it will be set to 16
38364                                  <1>       ;          Color levels: Color values are fixed to (nearest)
38365                                  <1>       ;              one of all possible set level (step) values
38366                                  <1>       ;              (according to luminosity base level); then
38367                                  <1>       ;              color levels are set to R-L, G-L, B-L.
38368                                  <1>       ;       For example: If luminosity base level is 32
38369                                  <1>       ;              all possible set values are 0, 32, 96, 255.
38370                                  <1>       ;
38371                                  <1>       ;         * RGB (TRUE) colors to 16 colors conversion:
38372                                  <1>       ;         16 colors: R, B,G, L bits (4 bits)
38373                                  <1>       ;              If any one of R,G,B >= 128 L = 1
38374                                  <1>       ;              If max. value of (R,G,B) >= 32, it is 1
38375                                  <1>       ;                 else all color bits (R&G&B&L) are 0
38376                                  <1>       ;              If the second value >= max. value / 2
38377                                  <1>       ;                 it is 1
38378                                  <1>       ;              If third value value >= max. value / 2
38379                                  <1>       ;                 it is 1
38380                                  <1>       ;           Example: R = 132, G = 64, B = 78
38381                                  <1>       ;                  L = 1, R = 1
38382                                  <1>       ;                  G < 66 --> G = 0
38383                                  <1>       ;                  B >= 66 --> B = 1
38384                                  <1>
38385                                  <1>       ; 16/05/2016
38386 0000CEBB 31C0                   <1>       xor    eax, eax
38387 0000CEBD A3[64030600]           <1>       mov    [u.r0], eax
38388                                  <1>
38389 0000CEC2 20FF                   <1>       and    bh, bh
38390 0000CEC4 0F8572020000           <1>       jnz    sysvideo_13 ; 11/07/2016
38391                                  <1>
38392                                  <1>       ; Video mode 0, 80*25 text mode, CGA 16 colors  ; [CRT_MODE] = 3
38393 0000CECA 88DF                   <1>       mov    bh, bl
38394 0000CECC C0EF02                 <1>       shr    bh, 2
38395 0000CECF 20FF                   <1>       and    bh, bh
38396 0000CED1 0F8598000000           <1>         jnz     sysvideo_4
38397 0000CED7 BF00800B00             <1>       mov    edi, 0B8000h
38398 0000CEDC 20D2                   <1>       and    dl, dl
38399 0000CEDE 7413                   <1>       jz     short sysvideo_1
38400 0000CEE0 80FA07                 <1>       cmp    dl, 7
38401 0000CEE3 0F87C6F5FFFF           <1>       ja     sysret
38402                                  <1> sysvideo_0:
38403 0000CEE9 81C7A00F0000           <1>       add    edi, 80*25*2
38404 0000CEEF FECA                   <1>       dec    dl
38405 0000CEF1 75F6                   <1>       jnz    short sysvideo_0
38406                                  <1> sysvideo_1:
38407 0000CEF3 80E303                 <1>       and    bl, 3
38408 0000CEF6 7530                   <1>       jnz    short sysvideo_2
38409 0000CEF8 80F907                 <1>       cmp    cl, 7
38410 0000CEFB 0F87AEF5FFFF           <1>       ja     sysret
38411                                  <1>       ; system to system video/display page transfer (mode 0)
38412 0000CF01 BE00800B00             <1>       mov    esi, 0B8000h
38413 0000CF06 0FB6C1                 <1>       movzx  eax, cl
38414 0000CF09 BAA00F0000             <1>       mov    edx, 80*25*2
38415 0000CF0E F7E2                   <1>       mul    edx
38416 0000CF10 01C6                   <1>       add    esi, eax
38417 0000CF12 B9A00F0000             <1>       mov    ecx, (80*25*2)
38418 0000CF17 890D[64030600]         <1>       mov    [u.r0], ecx
38419 0000CF1D 66C1E902               <1>       shr    cx, 2 ; /4
38420 0000CF21 F3A5                   <1>       rep    movsd
38421 0000CF23 E987F5FFFF             <1>       jmp    sysret
38422                                  <1> sysvideo_2:
38423 0000CF28 80FB02                 <1>       cmp    bl, 2
38424 0000CF2B 0F877EF5FFFF           <1>         ja      sysret
38425 0000CF31 721F                   <1>       jb     short sysvideo_3
38426                                  <1>       ; system to user video/display page transfer (mode 0)
38427 0000CF33 89FE                   <1>       mov    esi, edi
38428 0000CF35 89CF                   <1>       mov    edi, ecx ; user buffer
38429 0000CF37 B9A00F0000             <1>       mov    ecx, 80*25*2
38430 0000CF3C E841190000             <1>       call   transfer_to_user_buffer ; fast transfer
38431 0000CF41 0F8268F5FFFF           <1>       jc     sysret
38432 0000CF47 890D[64030600]         <1>       mov    [u.r0], ecx
38433 0000CF4D E95DF5FFFF             <1>       jmp    sysret
38434                                  <1> sysvideo_3:
38435                                  <1>       ; user to system video/display page transfer (mode 0)
38436 0000CF52 89CE                   <1>       mov    esi, ecx ; user buffer
38437                                  <1>       ; edi = video page address
38438 0000CF54 B9A00F0000             <1>       mov    ecx, 80*25*2
38439 0000CF59 E86E190000             <1>       call   transfer_from_user_buffer ; fast transfer
38440 0000CF5E 0F824BF5FFFF           <1>       jc     sysret
38441 0000CF64 890D[64030600]         <1>       mov    [u.r0], ecx
38442 0000CF6A E940F5FFFF             <1>       jmp    sysret
38443                                  <1> sysvideo_4:
38444 0000CF6F 80E303                 <1>       and    bl, 3
38445 0000CF72 0F85F6000000           <1>         jnz     sysvideo_9
```

```
38446 0000CF78 80F907              <1>       cmp    cl, 7
38447 0000CF7B 0F872EF5FFFF        <1>       ja     sysret
38448                              <1>       ; system to system video/display page window transfer (mode 0)
38449 0000CF81 81FE00800B00        <1>       cmp    esi, 0B8000h
38450 0000CF87 0F8222F5FFFF        <1>       jb     sysret
38451 0000CF8D 81FE00FD0B00        <1>       cmp    esi, 0B8000h+(80*25*2*8)
38452 0000CF93 0F8316F5FFFF        <1>       jnb    sysret
38453 0000CF99 81FF00800B00        <1>       cmp    edi, 0B8000h
38454 0000CF9F 0F820AF5FFFF        <1>       jb     sysret
38455 0000CFA5 81FF00FD0B00        <1>         cmp     edi, 0B8000h+(80*25*2*8)
38456 0000CFAB 0F83FEF4FFFF        <1>       jnb    sysret
38457                              <1>       ;
38458 0000CFB1 51                  <1>       push   ecx
38459 0000CFB2 52                  <1>       push   edx
38460 0000CFB3 0FB7C1              <1>       movzx  eax, cx ; top left column
38461 0000CFB6 50                  <1>       push   eax
38462 0000CFB7 C1E910              <1>       shr    ecx, 16 ; top row
38463 0000CFBA 66B8A000            <1>       mov    ax, 80*2 ; 80 colums, 160 bytes per row
38464 0000CFBE F7E1                <1>       mul    ecx
38465 0000CFC0 01C6                <1>       add    esi, eax
38466 0000CFC2 01C7                <1>       add    edi, eax
38467 0000CFC4 58                  <1>       pop    eax
38468 0000CFC5 66D1E0              <1>       shl    ax, 1 ; *2
38469 0000CFC8 01C6                <1>       add    esi, eax
38470 0000CFCA 01C7                <1>       add    edi, eax
38471 0000CFCC 5A                  <1>       pop    edx
38472 0000CFCD 59                  <1>       pop    ecx
38473 0000CFCE B800FD0B00          <1>       mov    eax, 0B8000h+(80*25*2*8)
38474 0000CFD3 39C6                <1>       cmp    esi, eax
38475 0000CFD5 0F83D4F4FFFF        <1>       jnb    sysret
38476 0000CFDB 39C6                <1>       cmp    esi, eax
38477 0000CFDD 0F83CCF4FFFF        <1>       jnb    sysret
38478                              <1>
38479 0000CFE3 56                  <1>       push   esi ; ****
38480 0000CFE4 57                  <1>       push   edi ; ***
38481 0000CFE5 52                  <1>       push   edx ; **
38482 0000CFE6 51                  <1>       push   ecx ; *
38483 0000CFE7 C1E910              <1>       shr    ecx, 16 ; top row
38484 0000CFEA C1EA10              <1>       shr    edx, 16 ; bottom row
38485 0000CFED 83F918              <1>       cmp    ecx, 24 ; max. 25 rows
38486 0000CFF0 7773                <1>       ja     short sysvideo_6
38487 0000CFF2 83FA18              <1>       cmp    edx, 24 ; max. 25 rows
38488 0000CFF5 776E                <1>       ja     short sysvideo_6
38489 0000CFF7 28CA                <1>       sub    dl, cl
38490 0000CFF9 726A                <1>       jc     short sysvideo_6
38491 0000CFFB 50                  <1>       push   eax ; *****
38492 0000CFFC 89D3                <1>       mov    ebx, edx ; row count - 1
38493 0000CFFE B8A0000000          <1>       mov    eax, 80*2
38494 0000D003 F7E0                <1>       mul    eax
38495 0000D005 01C6                <1>       add    esi, eax
38496 0000D007 01C7                <1>       add    edi, eax
38497 0000D009 58                  <1>       pop    eax ; *****
38498 0000D00A 39C6                <1>       cmp    esi, eax
38499 0000D00C 7757                <1>       ja     short sysvideo_6
38500 0000D00E 39C7                <1>       cmp    edi, eax
38501 0000D010 7753                <1>       ja     short sysvideo_6
38502 0000D012 59                  <1>       pop    ecx ; *
38503 0000D013 5A                  <1>       pop    edx ; **
38504 0000D014 81E1FFFF0000        <1>       and    ecx, 0FFFFh
38505 0000D01A 81E2FFFF0000        <1>       and    edx, 0FFFFh
38506 0000D020 83F94F              <1>       cmp    ecx, 79 ; max. 80 columns
38507 0000D023 7742                <1>       ja     short sysvideo_7
38508 0000D025 83FA4F              <1>       cmp    edx, 79 ; max. 80 columns
38509 0000D028 773D                <1>       ja     short sysvideo_7
38510 0000D02A 28CA                <1>       sub    dl, cl
38511 0000D02C 7639                <1>       jna    short sysvideo_7
38512                              <1>       ; edx = column count (width) - 1
38513 0000D02E D0E2                <1>       shl    dl, 1
38514 0000D030 01D6                <1>       add    esi, edx
38515 0000D032 01D7                <1>       add    edi, edx
38516 0000D034 39C6                <1>       cmp    esi, eax
38517 0000D036 772F                <1>       ja     short sysvideo_7
38518 0000D038 39C7                <1>       cmp    edi, eax
38519 0000D03A 772B                <1>       ja     short sysvideo_7
38520 0000D03C 5F                  <1>       pop    edi ; ***
38521 0000D03D 5E                  <1>       pop    esi ; ****
38522 0000D03E FEC3                <1>       inc    bl
38523 0000D040 FEC2                <1>       inc    dl ; column count
38524 0000D042 88D7                <1>       mov    bh, dl
38525 0000D044 D0E2                <1>       shl    dl, 1
38526 0000D046 B8A0000000          <1>       mov    eax, 80*2
38527 0000D04B 28D0                <1>       sub    al, dl ; (80 - columns) * 2
38528                              <1> sysvideo_5:
38529 0000D04D 88F9                <1>       mov    cl, bh
38530 0000D04F 0115[64030600]      <1>       add    [u.r0], edx
38531 0000D055 F366A5              <1>       rep    movsw
38532 0000D058 01C6                <1>       add    esi, eax ; next row
38533 0000D05A 01C7                <1>       add    edi, eax ; next row
38534 0000D05C FECB                <1>       dec    bl
38535 0000D05E 75ED                <1>       jnz    short sysvideo_5
38536 0000D060 E94AF4FFFF          <1>       jmp    sysret
38537                              <1>
38538                              <1> sysvideo_6:
38539 0000D065 59                  <1>       pop    ecx ; *
38540 0000D066 5A                  <1>       pop    edx ; **
38541                              <1> sysvideo_7:
38542 0000D067 5F                  <1>       pop    edi ; ***
38543 0000D068 5E                  <1>       pop    esi ; ****
38544 0000D069 E941F4FFFF          <1>       jmp    sysret
38545                              <1>
38546                              <1> sysvideo_9:
38547 0000D06E 80FB02              <1>       cmp    bl, 2
```

```
38548 0000D071 0F8738F4FFFF        <1>         ja      sysret
38549                              <1>
38550 0000D077 56                  <1>       push   esi ; ****
38551 0000D078 57                  <1>       push   edi ; ***
38552 0000D079 52                  <1>       push   edx ; **
38553 0000D07A 51                  <1>       push   ecx ; *
38554                              <1>
38555 0000D07B C1E910              <1>       shr    ecx, 16 ; top row
38556 0000D07E C1EA10              <1>       shr    edx, 16 ; bottom row
38557 0000D081 83F918              <1>       cmp    ecx, 24 ; max. 25 rows
38558 0000D084 77DF                <1>       ja     short sysvideo_6
38559 0000D086 83FA18              <1>       cmp    edx, 24 ; max. 25 rows
38560 0000D089 77DA                <1>       ja     short sysvideo_6
38561 0000D08B 28CA                <1>       sub    dl, cl
38562 0000D08D 72D6                <1>       jc     short sysvideo_6
38563                              <1>
38564 0000D08F 88CD                <1>       mov    ch, cl ; top row
38565 0000D091 8A0D[F64D0100]      <1>       mov    cl, [ACTIVE_PAGE]
38566 0000D097 BFA00F0000          <1>       mov    edi, 80*25*2
38567 0000D09C D3E7                <1>       shl    edi, cl
38568 0000D09E 81C760700B00        <1>       add    edi, 0B8000h - 80*25*2
38569                              <1>
38570 0000D0A4 88D7                <1>       mov    bh, dl ; row count - 1
38571 0000D0A6 88EA                <1>       mov    dl, ch ; top row
38572 0000D0A8 B8A0000000          <1>       mov    eax, 80*2
38573 0000D0AD F7E2                <1>       mul    edx
38574 0000D0AF 01C7                <1>       add    edi, eax
38575                              <1>
38576 0000D0B1 59                  <1>       pop    ecx ; *
38577 0000D0B2 5A                  <1>       pop    edx ; **
38578 0000D0B3 81E1FFFF0000        <1>       and    ecx, 0FFFFh
38579 0000D0B9 81E2FFFF0000        <1>       and    edx, 0FFFFh
38580 0000D0BF 83F94F              <1>       cmp    ecx, 79 ; max. 80 columns
38581 0000D0C2 77A3                <1>       ja     short sysvideo_7
38582 0000D0C4 83FA4F              <1>       cmp    edx, 79 ; max. 80 columns
38583 0000D0C7 779E                <1>       ja     short sysvideo_7
38584                              <1>
38585 0000D0C9 28CA                <1>       sub    dl, cl
38586 0000D0CB 769A                <1>       jna    short sysvideo_7
38587                              <1>
38588 0000D0CD 0FB6C1              <1>       movzx  eax, cl ; left column
38589 0000D0D0 D0E0                <1>       shl    al, 1  ; column * 2
38590 0000D0D2 01C7                <1>       add    edi, eax
38591                              <1>
38592 0000D0D4 FEC2                <1>       inc    dl ; column count
38593 0000D0D6 D0E2                <1>       shl    dl, 1
38594 0000D0D8 88D1                <1>       mov    cl, dl ; column count * 2
38595 0000D0DA B2A0                <1>       mov    dl, 80*2
38596 0000D0DC 58                  <1>       pop    eax ; *** (swap address)
38597 0000D0DD 5E                  <1>       pop    esi ; ****
38598 0000D0DE FEC7                <1>       inc    bh
38599                              <1>
38600                              <1>       ;mov   edx, 80*2
38601 0000D0E0 B2A0                <1>       mov    dl, 80*2
38602                              <1>       ;
38603 0000D0E2 80FB01              <1>       cmp    bl, 1
38604 0000D0E5 7735                <1>       ja     short sysvideo_11
38605                              <1>
38606                              <1>       ; user to system video/display page window transfer (mode 0)
38607 0000D0E7 21C0                <1>       and    eax, eax ; swap address
38608 0000D0E9 7413                <1>       jz     short sysvideo_10 ; no window swap
38609                              <1>       ; save previous window content in user's buffer (swap address)
38610 0000D0EB 56                  <1>       push   esi ; user buffer
38611 0000D0EC 57                  <1>       push   edi ; beginning address of the window
38612 0000D0ED 89FE                <1>       mov    esi, edi
38613 0000D0EF 89C7                <1>       mov    edi, eax
38614 0000D0F1 E88C170000          <1>       call   transfer_to_user_buffer ; fast transfer
38615 0000D0F6 5F                  <1>       pop    edi
38616 0000D0F7 5E                  <1>       pop    esi
38617 0000D0F8 0F82B1F3FFFF        <1>       jc     sysret
38618                              <1> sysvideo_10:
38619                              <1>       ; user to system video/display page window transfer (mode 0)
38620                              <1>       ; esi =      user buffer
38621 0000D0FE E8C9170000          <1>       call   transfer_from_user_buffer ; fast transfer
38622 0000D103 0F82A6F3FFFF        <1>       jc     sysret
38623 0000D109 010D[64030600]      <1>       add    [u.r0], ecx
38624 0000D10F 01D7                <1>       add    edi, edx ; next row
38625 0000D111 01CE                <1>       add    esi, ecx
38626 0000D113 FECF                <1>       dec    bh
38627 0000D115 75E7                <1>       jnz    short sysvideo_10
38628 0000D117 E993F3FFFF          <1>       jmp    sysret
38629                              <1>
38630                              <1> sysvideo_11:
38631                              <1>       ; system to user video/display page window transfer (mode 0)
38632 0000D11C 87FE                <1>       xchg   edi, esi
38633                              <1> sysvideo_12:
38634                              <1>       ; esi = beginning address of the window
38635                              <1>       ; edi =      user buffer
38636 0000D11E E85F170000          <1>       call   transfer_to_user_buffer ; fast transfer
38637 0000D123 0F8286F3FFFF        <1>       jc     sysret
38638 0000D129 010D[64030600]      <1>       add    [u.r0], ecx
38639 0000D12F 01D6                <1>       add    esi, edx ; next row
38640 0000D131 01CF                <1>       add    edi, ecx
38641 0000D133 FECF                <1>       dec    bh
38642 0000D135 75E7                <1>       jnz    short sysvideo_12
38643 0000D137 E973F3FFFF          <1>       jmp    sysret
38644                              <1>
38645                              <1> sysvideo_13:
38646 0000D13C 80FF01              <1>       cmp    bh, 1
38647 0000D13F 0F871F030000        <1>         ja      sysvideo_38
38648                              <1>       ; BH = 1 = CGA Graphics (0B8000h) data transfers
38649                              <1>
```

```
38650 0000D145 20DB                <1>       and    bl, bl
38651 0000D147 751A                <1>       jnz    short sysvideo_14
38652                              <1>
38653                              <1>       ; BL = 0 = Fill color (color in CL] (32K)
38654                              <1>
38655 0000D149 88C8                <1>       mov    al, cl
38656 0000D14B B900800000          <1>       mov    ecx, 32768
38657 0000D150 66890D[64030600]    <1>       mov    [u.r0], cx
38658 0000D157 BF00800B00          <1>       mov    edi, 0B8000h
38659 0000D15C F3AB                <1>       rep    stosd
38660 0000D15E E94CF3FFFF          <1>       jmp    sysret
38661                              <1>
38662                              <1> sysvideo_14:
38663 0000D163 80FB01              <1>       cmp    bl, 1
38664 0000D166 7723                <1>       ja     short sysvideo_16
38665                              <1>
38666 0000D168 89CE                <1>       mov    esi, ecx ; user buffer
38667                              <1>       ; BL = 1 = user to system video/display page transfer
38668                              <1> sysvideo_15:
38669 0000D16A BF00800B00          <1>       mov    edi, 0B8000h
38670                              <1>       ; edi = video page address
38671 0000D16F B900800000          <1>       mov    ecx, 32768
38672 0000D174 E853170000          <1>       call   transfer_from_user_buffer ; fast transfer
38673 0000D179 0F8230F3FFFF        <1>       jc     sysret ; [u.r0] = 0
38674 0000D17F 66890D[64030600]    <1>       mov    [u.r0], cx
38675 0000D186 E924F3FFFF          <1>       jmp    sysret
38676                              <1>
38677                              <1> sysvideo_16:
38678 0000D18B 80FB02              <1>       cmp    bl, 2
38679 0000D18E 7723                <1>       ja     short sysvideo_18
38680                              <1>
38681 0000D190 89CF                <1>       mov    edi, ecx ; user buffer
38682                              <1>       ; BL = 2 = system to user video/display page transfer
38683                              <1> sysvideo_17:
38684 0000D192 BE00800B00          <1>       mov    esi, 0B8000h
38685 0000D197 B900800000          <1>       mov    ecx, 32768
38686 0000D19C E8E1160000          <1>       call   transfer_to_user_buffer ; fast transfer
38687 0000D1A1 0F8208F3FFFF        <1>       jc     sysret ; [u.r0] = 0
38688 0000D1A7 66890D[64030600]    <1>       mov    [u.r0], cx
38689 0000D1AE E9FCF2FFFF          <1>       jmp    sysret
38690                              <1>
38691                              <1> sysvideo_18:
38692 0000D1B3 80FB03              <1>       cmp    bl, 3
38693 0000D1B6 777E                <1>       ja     short sysvideo_23
38694                              <1>
38695                              <1>       ; BL = 3 = NOT bits in window (ECX, EDX)
38696                              <1>
38697 0000D1B8 BF00800B00          <1>       mov    edi, 0B8000h
38698 0000D1BD 89FE                <1>       mov    esi, edi
38699                              <1>
38700 0000D1BF 39CA                <1>       cmp    edx, ecx ; bottom-right > top-left ?
38701 0000D1C1 7716                <1>       ja     short sysvideo_20 ; window
38702                              <1>       ; full screen (update)
38703 0000D1C3 B900800000          <1>       mov    ecx, 32768
38704 0000D1C8 66890D[64030600]    <1>       mov    [u.r0], cx
38705                              <1> sysvideo_19:
38706 0000D1CF F616                <1>       not    byte [esi] ; NOT operation
38707 0000D1D1 46                  <1>       inc    esi
38708 0000D1D2 E2FB                <1>       loop   sysvideo_19
38709 0000D1D4 E9D6F2FFFF          <1>       jmp    sysret
38710                              <1> sysvideo_20:
38711 0000D1D9 0FB7C2              <1>       movzx  eax, dx ; bottom right column
38712 0000D1DC 6629C8              <1>       sub    ax, cx  ; - top left column
38713 0000D1DF 0F82CAF2FFFF        <1>         jb      sysret ; invalid
38714 0000D1E5 6640                <1>       inc    ax ; same column no == 1 column
38715 0000D1E7 50                  <1>       push   eax ; byte count per window row
38716 0000D1E8 52                  <1>       push   edx
38717 0000D1E9 BB40010000          <1>       mov    ebx, 320 ; screen width
38718 0000D1EE 89C8                <1>       mov    eax, ecx
38719 0000D1F0 C1E810              <1>       shr    eax, 16  ; top row
38720 0000D1F3 F7E3                <1>       mul    ebx
38721 0000D1F5 6689CA              <1>       mov    dx, cx ; top left column
38722 0000D1F8 01D0                <1>       add    eax, edx
38723 0000D1FA 01C6                <1>       add    esi, eax ; start address
38724 0000D1FC 59                  <1>       pop    ecx ; edx
38725 0000D1FD 89C8                <1>       mov    eax, ecx
38726 0000D1FF C1E810              <1>       shr    eax, 16 ; bottom row
38727 0000D202 F7E3                <1>       mul    ebx
38728 0000D204 6689CA              <1>       mov    dx, cx ; bottom right column
38729 0000D207 01D0                <1>       add    eax, edx
38730 0000D209 01C7                <1>       add    edi, eax ; stop address (included)
38731 0000D20B 5A                  <1>       pop    edx ; byte count per window row
38732 0000D20C 81FFFFFF0B00        <1>       cmp    edi, 0BFFFFh
38733 0000D212 0F8797F2FFFF        <1>       ja     sysret
38734 0000D218 56                  <1>       push   esi
38735 0000D219 4E                  <1>       dec    esi
38736                              <1> sysvideo_21:
38737 0000D21A 89D1                <1>       mov    ecx, edx
38738                              <1> sysvideo_22:
38739 0000D21C 46                  <1>       inc    esi
38740 0000D21D F616                <1>       not    byte [esi]
38741 0000D21F E2FB                <1>       loop   sysvideo_22
38742 0000D221 01DE                <1>       add    esi, ebx ; bytes per screen row
38743                              <1>       ;
38744 0000D223 39FE                <1>       cmp    esi, edi ; stop address (included in loop)
38745 0000D225 76F3                <1>       jna    short sysvideo_21
38746 0000D227 5E                  <1>       pop    esi
38747 0000D228 29F7                <1>       sub    edi, esi
38748 0000D22A 66893D[64030600]    <1>       mov    [u.r0], di
38749 0000D231 E979F2FFFF          <1>       jmp    sysret
38750                              <1>
38751                              <1> sysvideo_23:
```

```
38752 0000D236 80FB04              <1>       cmp    bl, 4
38753 0000D239 0F87A7000000        <1>         ja      sysvideo_26
38754                              <1>
38755                              <1>       ; BL = 4 = window copy (system to system)
38756                              <1>
38757 0000D23F B800800B00          <1>       mov    eax, 0B8000h
38758 0000D244 39C6                <1>       cmp    esi, eax
38759 0000D246 0F8263F2FFFF        <1>       jb     sysret
38760 0000D24C 39C7                <1>       cmp    edi, eax
38761 0000D24E 0F825BF2FFFF        <1>       jb     sysret
38762 0000D254 6605FF7F            <1>       add    ax, 7FFFh ; 32767
38763 0000D258 39C6                <1>       cmp    esi, eax
38764 0000D25A 0F874FF2FFFF        <1>       ja     sysret
38765 0000D260 39C7                <1>       cmp    edi, eax
38766 0000D262 0F8747F2FFFF        <1>       ja     sysret
38767                              <1>
38768 0000D268 39CA                <1>       cmp    edx, ecx ; bottom-right > top-left ?
38769 0000D26A 7714                <1>       ja     short sysvideo_24 ; window
38770                              <1>       ; full screen copy
38771 0000D26C 89C1                <1>       mov    ecx, eax
38772 0000D26E 29F9                <1>       sub    ecx, edi
38773 0000D270 6641                <1>       inc    cx
38774 0000D272 66890D[64030600]    <1>       mov    [u.r0], cx
38775 0000D279 F3A4                <1>       rep    movsb
38776 0000D27B E92FF2FFFF          <1>       jmp    sysret
38777                              <1> sysvideo_24:
38778 0000D280 0FB7C2              <1>       movzx  eax, dx ; bottom right column
38779 0000D283 6629C8              <1>       sub    ax, cx  ; - top left column
38780 0000D286 0F8223F2FFFF        <1>         jb      sysret ; invalid
38781 0000D28C 6640                <1>       inc    ax ; same column no == 1 column
38782 0000D28E 50                  <1>       push   eax ; byte count per window row
38783                              <1>       ;
38784 0000D28F 52                  <1>       push   edx
38785 0000D290 BB40010000          <1>       mov    ebx, 320 ; screen width
38786 0000D295 89C8                <1>       mov    eax, ecx
38787 0000D297 C1E810              <1>       shr    eax, 16      ; top row
38788 0000D29A F7E3                <1>       mul    ebx
38789 0000D29C 6689CA              <1>       mov    dx, cx ; top left column
38790 0000D29F 01D0                <1>       add    eax, edx
38791 0000D2A1 01C7                <1>       add    edi, eax ; start address
38792 0000D2A3 01C6                <1>       add    esi, eax
38793 0000D2A5 59                  <1>       pop    ecx ; edx
38794 0000D2A6 89C8                <1>       mov    eax, ecx
38795 0000D2A8 C1E810              <1>       shr    eax, 16 ; bottom row
38796 0000D2AB F7E3                <1>       mul    ebx
38797 0000D2AD 6689CA              <1>       mov    dx, cx ; bottom right column
38798 0000D2B0 01D0                <1>       add    eax, edx
38799 0000D2B2 5A                  <1>       pop    edx ; byte count per window row
38800 0000D2B3 0500800B00          <1>       add    eax, 0B8000h
38801 0000D2B8 3DFFFF0B00          <1>       cmp    eax, 0BFFFFh
38802 0000D2BD 0F87ECF1FFFF        <1>       ja     sysret
38803 0000D2C3 57                  <1>       push   edi ; start address
38804 0000D2C4 50                  <1>       push   eax ; stop address (included)
38805                              <1> sysvideo_25:
38806 0000D2C5 89D1                <1>       mov    ecx, edx
38807 0000D2C7 F3A4                <1>       rep    movsb
38808 0000D2C9 4F                  <1>       dec    edi
38809 0000D2CA 4E                  <1>       dec    esi
38810 0000D2CB 01DF                <1>       add    edi, ebx ; bytes per screen row
38811 0000D2CD 01DE                <1>       add    esi, ebx
38812                              <1>       ;
38813 0000D2CF 3B3C24              <1>       cmp    edi, [esp] ; stop addr(included in loop)
38814 0000D2D2 76F1                <1>       jna    short sysvideo_25
38815 0000D2D4 5B                  <1>       pop    ebx ; stop address
38816 0000D2D5 5F                  <1>       pop    edi ; start address
38817 0000D2D6 29FB                <1>       sub    ebx, edi
38818 0000D2D8 6643                <1>       inc    bx
38819 0000D2DA 66891D[64030600]    <1>       mov    [u.r0], bx
38820 0000D2E1 E9C9F1FFFF          <1>       jmp    sysret
38821                              <1>
38822                              <1> sysvideo_26:
38823 0000D2E6 80FB05              <1>       cmp    bl, 5
38824 0000D2E9 0F8795000000        <1>         ja      sysvideo_29
38825                              <1>
38826                              <1>       ; BL = 5 = window copy (user to system)
38827                              <1>
38828 0000D2EF B800800B00          <1>       mov    eax, 0B8000h
38829 0000D2F4 39C7                <1>       cmp    edi, eax
38830 0000D2F6 0F82B3F1FFFF        <1>       jb     sysret
38831 0000D2FC 6605FF7F            <1>       add    ax, 7FFFh ; 32767
38832 0000D300 39C7                <1>       cmp    edi, eax
38833 0000D302 0F87A7F1FFFF        <1>       ja     sysret
38834                              <1>
38835                              <1>       ; esi = user buffer (in user's memory space)
38836 0000D308 39CA                <1>       cmp    edx, ecx ; bottom-right > top-left ?
38837 0000D30A 0F865AFEFFFF        <1>         jna     sysvideo_15 ; full screen copy
38838                              <1>
38839 0000D310 0FB7C2              <1>       movzx  eax, dx ; bottom right column
38840 0000D313 6629C8              <1>       sub    ax, cx  ; - top left column
38841 0000D316 0F8293F1FFFF        <1>         jb      sysret ; invalid
38842 0000D31C 6640                <1>       inc    ax ; same column no == 1 column
38843 0000D31E 50                  <1>       push   eax ; byte count per window row
38844                              <1>
38845 0000D31F 52                  <1>       push   edx
38846 0000D320 BB40010000          <1>       mov    ebx, 320 ; screen width
38847 0000D325 89C8                <1>       mov    eax, ecx
38848 0000D327 C1E810              <1>       shr    eax, 16      ; top row
38849 0000D32A F7E3                <1>       mul    ebx
38850 0000D32C 6689CA              <1>       mov    dx, cx ; top left column
38851 0000D32F 01D0                <1>       add    eax, edx
38852 0000D331 01C7                <1>       add    edi, eax ; start address
38853 0000D333 59                  <1>       pop    ecx ; edx
```

```
38854 0000D334 89C8              <1>       mov   eax, ecx
38855 0000D336 C1E810            <1>       shr   eax, 16 ; bottom row
38856 0000D339 F7E3              <1>       mul   ebx
38857 0000D33B 6689CA            <1>       mov   dx, cx ; bottom right column
38858 0000D33E 01D0              <1>       add   eax, edx
38859 0000D340 5A                <1>       pop   edx ; byte count per window row
38860 0000D341 0500800B00        <1>       add   eax, 0B8000h
38861 0000D346 3DFFFF0B00        <1>       cmp   eax, 0BFFFFh
38862 0000D34B 0F875EF1FFFF      <1>       ja    sysret
38863 0000D351 57                <1>       push  edi ; start address
38864 0000D352 50                <1>       push  eax ; stop address (included)
38865                            <1> sysvideo_27:
38866 0000D353 89D1              <1>       mov   ecx, edx ; byte count
38867                            <1>       ; user to system video/display page window transfer
38868                            <1>       ; esi =      user buffer
38869 0000D355 E872150000        <1>       call  transfer_from_user_buffer ; fast transfer
38870 0000D35A 7221              <1>       jc    short sysvideo_28
38871 0000D35C 010D[64030600]    <1>       add   [u.r0], ecx
38872 0000D362 01DF              <1>       add   edi, ebx ; next row
38873 0000D364 01CE              <1>       add   esi, ecx
38874 0000D366 3B3C24            <1>       cmp   edi, [esp] ; stop addr(included in loop)
38875 0000D369 76E8              <1>       jna   short sysvideo_27
38876 0000D36B 5B                <1>       pop   ebx ; stop address
38877 0000D36C 5F                <1>       pop   edi ; start address
38878 0000D36D 29FB              <1>       sub   ebx, edi
38879 0000D36F 6643              <1>       inc   bx
38880 0000D371 66891D[64030600]  <1>       mov   [u.r0], bx
38881 0000D378 E932F1FFFF        <1>       jmp   sysret
38882                            <1> sysvideo_28:
38883 0000D37D 58                <1>       pop   eax
38884 0000D37E 5A                <1>       pop   edx
38885 0000D37F E92BF1FFFF        <1>       jmp   sysret
38886                            <1>
38887                            <1> sysvideo_29:
38888 0000D384 80FB06            <1>       cmp   bl, 6
38889 0000D387 0F8797000000      <1>         ja      sysvideo_32
38890                            <1>
38891                            <1>       ; BL = 6 = window copy (system to user)
38892                            <1>
38893 0000D38D 89F7              <1>       mov   edi, esi ; user buffer
38894                            <1>
38895 0000D38F B800800B00        <1>       mov   eax, 0B8000h
38896 0000D394 39C6              <1>       cmp   esi, eax
38897 0000D396 0F8213F1FFFF      <1>       jb    sysret
38898 0000D39C 6605FF7F          <1>       add   ax, 7FFFh ; 32767
38899 0000D3A0 39C6              <1>       cmp   esi, eax
38900 0000D3A2 0F8707F1FFFF      <1>       ja    sysret
38901                            <1>
38902                            <1>       ; edi = user buffer (in user's memory space)
38903 0000D3A8 39CA              <1>       cmp   edx, ecx ; bottom-right > top-left ?
38904 0000D3AA 0F86E2FDFFFF      <1>         jna     sysvideo_17 ; full screen copy
38905                            <1>
38906 0000D3B0 0FB7C2            <1>       movzx eax, dx ; bottom right column
38907 0000D3B3 6629C8            <1>       sub   ax, cx  ; - top left column
38908 0000D3B6 0F82F3F0FFFF      <1>         jb      sysret ; invalid
38909 0000D3BC 6640              <1>       inc   ax ; same column no == 1 column
38910 0000D3BE 50                <1>       push  eax ; byte count per window row
38911                            <1>
38912 0000D3BF 52                <1>       push  edx
38913 0000D3C0 BB40010000        <1>       mov   ebx, 320 ; screen width
38914 0000D3C5 89C8              <1>       mov   eax, ecx
38915 0000D3C7 C1E810            <1>       shr   eax, 16      ; top row
38916 0000D3CA F7E3              <1>       mul   ebx
38917 0000D3CC 6689CA            <1>       mov   dx, cx ; top left column
38918 0000D3CF 01D0              <1>       add   eax, edx
38919 0000D3D1 01C6              <1>       add   esi, eax ; start address
38920 0000D3D3 59                <1>       pop   ecx ; edx
38921 0000D3D4 89C8              <1>       mov   eax, ecx
38922 0000D3D6 C1E810            <1>       shr   eax, 16 ; bottom row
38923 0000D3D9 F7E3              <1>       mul   ebx
38924 0000D3DB 6689CA            <1>       mov   dx, cx ; bottom right column
38925 0000D3DE 01D0              <1>       add   eax, edx
38926 0000D3E0 5A                <1>       pop   edx ; byte count per window row
38927 0000D3E1 0500800B00        <1>       add   eax, 0B8000h
38928 0000D3E6 3DFFFF0B00        <1>       cmp   eax, 0BFFFFh
38929 0000D3EB 0F87BEF0FFFF      <1>       ja    sysret
38930 0000D3F1 56                <1>       push  esi ; start address
38931 0000D3F2 50                <1>       push  eax ; stop address (included)
38932                            <1> sysvideo_30:
38933 0000D3F3 89D1              <1>       mov   ecx, edx ; byte count
38934                            <1>       ; user to system video/display page window transfer
38935                            <1>       ; esi =      user buffer
38936 0000D3F5 E888140000        <1>       call  transfer_to_user_buffer ; fast transfer
38937 0000D3FA 7221              <1>       jc    short sysvideo_31
38938 0000D3FC 010D[64030600]    <1>       add   [u.r0], ecx
38939 0000D402 01DF              <1>       add   edi, ebx ; next row
38940 0000D404 01CE              <1>       add   esi, ecx
38941 0000D406 3B3C24            <1>       cmp   edi, [esp] ; stop addr(included in loop)
38942 0000D409 76E8              <1>       jna   short sysvideo_30
38943 0000D40B 5B                <1>       pop   ebx ; stop address
38944 0000D40C 5F                <1>       pop   edi ; start address
38945 0000D40D 29FB              <1>       sub   ebx, edi
38946 0000D40F 6643              <1>       inc   bx
38947 0000D411 66891D[64030600]  <1>       mov   [u.r0], bx
38948 0000D418 E992F0FFFF        <1>       jmp   sysret
38949                            <1> sysvideo_31:
38950 0000D41D 58                <1>       pop   eax
38951 0000D41E 5A                <1>       pop   edx
38952 0000D41F E98BF0FFFF        <1>       jmp   sysret
38953                            <1>
38954                            <1> sysvideo_32:
38955 0000D424 80FB07            <1>       cmp   bl, 7
```

```
38956 0000D427 770F              <1>       ja    short sysvideo_34
38957                            <1>
38958                            <1>       ; BL = 7 = AND display page bytes with CL
38959                            <1>
38960 0000D429 BE00800B00        <1>       mov   esi, 0B8000h
38961 0000D42E B900800000        <1>       mov   ecx, 32768
38962                            <1> sysvideo_33:
38963 0000D433 200E              <1>       and   byte [esi], cl
38964 0000D435 46                <1>       inc   esi
38965 0000D436 E2FB              <1>       loop  sysvideo_33
38966                            <1>
38967                            <1> sysvideo_34:
38968 0000D438 80FB08            <1>       cmp   bl, 8
38969 0000D43B 770F              <1>       ja    short sysvideo_36
38970                            <1>
38971                            <1>       ; BL = 8 = OR display page bytes with CL
38972                            <1>
38973 0000D43D BE00800B00        <1>       mov   esi, 0B8000h
38974 0000D442 B900800000        <1>       mov   ecx, 32768
38975                            <1> sysvideo_35:
38976 0000D447 080E              <1>       or    byte [esi], cl
38977 0000D449 46                <1>       inc   esi
38978 0000D44A E2FB              <1>       loop  sysvideo_35
38979                            <1>
38980                            <1> sysvideo_36:
38981 0000D44C 80FB09            <1>       cmp   bl, 9
38982 0000D44F 0F875AF0FFFF      <1>         ja      sysret ; nothing to do
38983                            <1>
38984                            <1>       ; BL = 9 = XOR display page bytes with CL
38985                            <1>
38986 0000D455 BE00800B00        <1>       mov   esi, 0B8000h
38987 0000D45A B900800000        <1>       mov   ecx, 32768
38988                            <1> sysvideo_37:
38989 0000D45F 300E              <1>       xor   byte [esi], cl
38990 0000D461 46                <1>       inc   esi
38991 0000D462 E2FB              <1>       loop  sysvideo_37
38992                            <1>
38993                            <1> sysvideo_38:
38994 0000D464 80FF02            <1>       cmp   bh, 2
38995 0000D467 0F8733030000      <1>         ja      sysvideo_64
38996                            <1>       ; BH = 2 = VGA Graphics (0A0000h) data transfers
38997                            <1>
38998 0000D46D 88DC              <1>       mov   ah, bl
38999 0000D46F 80E30F            <1>       and   bl, 0Fh
39000 0000D472 C0EC04            <1>       shr   ah, 4
39001 0000D475 C1E310            <1>       shl   ebx, 16
39002 0000D478 66BB4001          <1>       mov   bx, 320 ; 320*200, 320*240
39003 0000D47C 20E4              <1>       and   ah, ah
39004 0000D47E 7413              <1>       jz    short sysvideo_39
39005 0000D480 66D1E3            <1>       shl   bx, 1 ; 640*200, 640 * 400, 640*480
39006 0000D483 80FC02            <1>       cmp   ah, 2
39007 0000D486 720B              <1>       jb    short sysvideo_39
39008 0000D488 0F8721F0FFFF      <1>       ja    sysret ; invalid
39009                            <1>       ; 800*600
39010 0000D48E 6681C3A000        <1>       add   bx, 160 ; 800
39011                            <1> sysvideo_39:
39012 0000D493 C1CB10            <1>       ror   ebx, 16
39013                            <1>
39014 0000D496 20DB              <1>       and   bl, bl
39015 0000D498 7519              <1>       jnz   short sysvideo_40
39016                            <1>
39017                            <1>       ; BL = 0 = Fill color (color in CL] (64K)
39018                            <1>
39019 0000D49A 88C8              <1>       mov   al, cl
39020 0000D49C B900000100        <1>       mov   ecx, 65536
39021 0000D4A1 890D[64030600]    <1>       mov   [u.r0], ecx
39022 0000D4A7 BF00000A00        <1>       mov   edi, 0A0000h
39023 0000D4AC F3AB              <1>       rep   stosd
39024 0000D4AE E9FCEFFFFF        <1>       jmp   sysret
39025                            <1>
39026                            <1> sysvideo_40:
39027 0000D4B3 80FB01            <1>       cmp   bl, 1
39028 0000D4B6 7722              <1>       ja    short sysvideo_42
39029                            <1>
39030 0000D4B8 89CE              <1>       mov   esi, ecx ; user buffer
39031                            <1>       ; BL = 1 = user to system video/display page transfer
39032                            <1> sysvideo_41:
39033 0000D4BA BF00000A00        <1>       mov   edi, 0A0000h
39034                            <1>       ; edi = video page address
39035 0000D4BF B900000100        <1>       mov   ecx, 65536
39036 0000D4C4 E803140000        <1>       call  transfer_from_user_buffer ; fast transfer
39037 0000D4C9 0F82E0EFFFFF      <1>       jc    sysret ; [u.r0] = 0
39038 0000D4CF 890D[64030600]    <1>       mov   [u.r0], ecx
39039 0000D4D5 E9D5EFFFFF        <1>       jmp   sysret
39040                            <1>
39041                            <1> sysvideo_42:
39042 0000D4DA 80FB02            <1>       cmp   bl, 2
39043 0000D4DD 7722              <1>       ja    short sysvideo_44
39044                            <1>
39045 0000D4DF 89CF              <1>       mov   edi, ecx ; user buffer
39046                            <1>       ; BL = 2 = system to user video/display page transfer
39047                            <1> sysvideo_43:
39048 0000D4E1 BE00000A00        <1>       mov   esi, 0A0000h
39049 0000D4E6 B900000100        <1>       mov   ecx, 65536
39050 0000D4EB E892130000        <1>       call  transfer_to_user_buffer ; fast transfer
39051 0000D4F0 0F82B9EFFFFF      <1>       jc    sysret ; [u.r0] = 0
39052 0000D4F6 890D[64030600]    <1>       mov   [u.r0], ecx
39053 0000D4FC E9AEEFFFFF        <1>       jmp   sysret
39054                            <1>
39055                            <1> sysvideo_44:
39056 0000D501 80FB03            <1>       cmp   bl, 3
39057 0000D504 777A              <1>       ja    short sysvideo_49
```

```
39058                                  <1>
39059                                  <1>       ; BL = 3 = NOT bits in window (ECX, EDX)
39060                                  <1>
39061 0000D506 BF00000A00              <1>       mov    edi, 0A0000h
39062 0000D50B 89FE                    <1>       mov    esi, edi
39063                                  <1>
39064 0000D50D 39CA                    <1>       cmp    edx, ecx ; bottom-right > top-left ?
39065 0000D50F 770B                    <1>       ja     short sysvideo_45 ; window
39066                                  <1>       ; full screen (update)
39067 0000D511 B900000100              <1>       mov    ecx, 65536
39068 0000D516 890D[64030600]          <1>       mov    [u.r0], ecx
39069                                  <1> sysvideo_45:
39070 0000D51C F616                    <1>       not    byte [esi] ; NOT operation
39071 0000D51E 46                      <1>       inc    esi
39072 0000D51F E2FB                    <1>       loop   sysvideo_45
39073 0000D521 E989EFFFFF              <1>       jmp    sysret
39074                                  <1> sysvideo_46:
39075 0000D526 0FB7C2                  <1>       movzx  eax, dx ; bottom right column
39076 0000D529 6629C8                  <1>       sub    ax, cx  ; - top left column
39077 0000D52C 0F827DEFFFFF            <1>         jb      sysret ; invalid
39078 0000D532 6640                    <1>       inc    ax ; same column no == 1 column
39079 0000D534 50                      <1>       push   eax ; byte count per window row
39080 0000D535 52                      <1>       push   edx
39081 0000D536 C1EB10                  <1>       shr    ebx, 16 ; 320,640,800 : screen width
39082 0000D539 89C8                    <1>       mov    eax, ecx
39083 0000D53B C1E810                  <1>       shr    eax, 16  ; top row
39084 0000D53E F7E3                    <1>       mul    ebx
39085 0000D540 6689CA                  <1>       mov    dx, cx ; top left column
39086 0000D543 01D0                    <1>       add    eax, edx
39087 0000D545 01C6                    <1>       add    esi, eax ; start address
39088 0000D547 59                      <1>       pop    ecx ; edx
39089 0000D548 89C8                    <1>       mov    eax, ecx
39090 0000D54A C1E810                  <1>       shr    eax, 16 ; bottom row
39091 0000D54D F7E3                    <1>       mul    ebx
39092 0000D54F 6689CA                  <1>       mov    dx, cx ; bottom right column
39093 0000D552 01D0                    <1>       add    eax, edx
39094 0000D554 01C7                    <1>       add    edi, eax ; stop address (included)
39095 0000D556 5A                      <1>       pop    edx ; byte count per window row
39096 0000D557 81FFFFFF0A00            <1>       cmp    edi, 0AFFFFh
39097 0000D55D 0F874CEFFFFF            <1>       ja     sysret
39098 0000D563 56                      <1>       push   esi
39099 0000D564 4E                      <1>       dec    esi
39100                                  <1> sysvideo_47:
39101 0000D565 89D1                    <1>       mov    ecx, edx
39102                                  <1> sysvideo_48:
39103 0000D567 46                      <1>       inc    esi
39104 0000D568 F616                    <1>       not    byte [esi]
39105 0000D56A E2FB                    <1>       loop   sysvideo_48
39106 0000D56C 01DE                    <1>       add    esi, ebx ; bytes per screen row
39107                                  <1>       ;
39108 0000D56E 39FE                    <1>       cmp    esi, edi ; stop address (included in loop)
39109 0000D570 76F3                    <1>       jna    short sysvideo_47
39110 0000D572 5E                      <1>       pop    esi
39111 0000D573 29F7                    <1>       sub    edi, esi
39112 0000D575 893D[64030600]          <1>       mov    [u.r0], edi
39113 0000D57B E92FEFFFFF              <1>       jmp    sysret
39114                                  <1>
39115                                  <1> sysvideo_49:
39116 0000D580 80FB04                  <1>       cmp    bl, 4
39117 0000D583 0F87A1000000            <1>         ja      sysvideo_52
39118                                  <1>
39119                                  <1>       ; BL = 4 = window copy (system to system)
39120                                  <1>
39121 0000D589 B800000A00              <1>       mov    eax, 0A0000h
39122 0000D58E 39C6                    <1>       cmp    esi, eax
39123 0000D590 0F8219EFFFFF            <1>       jb     sysret
39124 0000D596 39C7                    <1>       cmp    edi, eax
39125 0000D598 0F8211EFFFFF            <1>       jb     sysret
39126 0000D59E 6683C0FF                <1>       add    ax, 0FFFFh ; 65535
39127 0000D5A2 39C6                    <1>       cmp    esi, eax
39128 0000D5A4 0F8705EFFFFF            <1>       ja     sysret
39129 0000D5AA 39C7                    <1>       cmp    edi, eax
39130 0000D5AC 0F87FDEEFFFF            <1>       ja     sysret
39131                                  <1>
39132 0000D5B2 39CA                    <1>       cmp    edx, ecx ; bottom-right > top-left ?
39133 0000D5B4 7712                    <1>       ja     short sysvideo_50 ; window
39134                                  <1>       ; full screen copy
39135 0000D5B6 89C1                    <1>       mov    ecx, eax
39136 0000D5B8 29F9                    <1>       sub    ecx, edi
39137 0000D5BA 41                      <1>       inc    ecx
39138 0000D5BB 890D[64030600]          <1>       mov    [u.r0], ecx
39139 0000D5C1 F3A4                    <1>       rep    movsb
39140 0000D5C3 E9E7EEFFFF              <1>       jmp    sysret
39141                                  <1> sysvideo_50:
39142 0000D5C8 0FB7C2                  <1>       movzx  eax, dx ; bottom right column
39143 0000D5CB 6629C8                  <1>       sub    ax, cx  ; - top left column
39144 0000D5CE 0F82DBEEFFFF            <1>         jb      sysret ; invalid
39145 0000D5D4 6640                    <1>       inc    ax ; same column no == 1 column
39146 0000D5D6 50                      <1>       push   eax ; byte count per window row
39147                                  <1>       ;
39148 0000D5D7 52                      <1>       push   edx
39149 0000D5D8 C1EB10                  <1>       shr    ebx, 16 ; 320,640,800 : screen width
39150 0000D5DB 89C8                    <1>       mov    eax, ecx
39151 0000D5DD C1E810                  <1>       shr    eax, 16      ; top row
39152 0000D5E0 F7E3                    <1>       mul    ebx
39153 0000D5E2 6689CA                  <1>       mov    dx, cx ; top left column
39154 0000D5E5 01D0                    <1>       add    eax, edx
39155 0000D5E7 01C7                    <1>       add    edi, eax ; start address
39156 0000D5E9 01C6                    <1>       add    esi, eax
39157 0000D5EB 59                      <1>       pop    ecx ; edx
39158 0000D5EC 89C8                    <1>       mov    eax, ecx
39159 0000D5EE C1E810                  <1>       shr    eax, 16 ; bottom row
```

```
39160 0000D5F1 F7E3              <1>       mul   ebx
39161 0000D5F3 6689CA            <1>       mov   dx, cx ; bottom right column
39162 0000D5F6 01D0              <1>       add   eax, edx
39163 0000D5F8 5A                <1>       pop   edx ; byte count per window row
39164 0000D5F9 0500000A00        <1>       add   eax, 0A0000h
39165 0000D5FE 3DFFFF0A00        <1>       cmp   eax, 0AFFFFh
39166 0000D603 0F87A6EEFFFF      <1>       ja    sysret
39167 0000D609 57                <1>       push  edi ; start address
39168 0000D60A 50                <1>       push  eax ; stop address (included)
39169                            <1> sysvideo_51:
39170 0000D60B 89D1              <1>       mov   ecx, edx
39171 0000D60D F3A4              <1>       rep   movsb
39172 0000D60F 4F                <1>       dec   edi
39173 0000D610 4E                <1>       dec   esi
39174 0000D611 01DF              <1>       add   edi, ebx ; bytes per screen row
39175 0000D613 01DE              <1>       add   esi, ebx
39176                            <1>       ;
39177 0000D615 3B3C24            <1>       cmp   edi, [esp] ; stop addr(included in loop)
39178 0000D618 76F1              <1>       jna   short sysvideo_51
39179 0000D61A 5B                <1>       pop   ebx ; stop address
39180 0000D61B 5F                <1>       pop   edi ; start address
39181 0000D61C 29FB              <1>       sub   ebx, edi
39182 0000D61E 43                <1>       inc   ebx
39183 0000D61F 891D[64030600]    <1>       mov   [u.r0], ebx
39184 0000D625 E985EEFFFF        <1>       jmp   sysret
39185                            <1>
39186                            <1> sysvideo_52:
39187 0000D62A 80FB05            <1>       cmp   bl, 5
39188 0000D62D 0F8791000000      <1>         ja      sysvideo_55
39189                            <1>
39190                            <1>       ; BL = 5 = window copy (user to system)
39191                            <1>
39192 0000D633 B800000A00        <1>       mov   eax, 0A0000h
39193 0000D638 39C7              <1>       cmp   edi, eax
39194 0000D63A 0F826FEEFFFF      <1>       jb    sysret
39195 0000D640 6683C0FF          <1>       add   ax, 0FFFFh ; 65535
39196 0000D644 39C7              <1>       cmp   edi, eax
39197 0000D646 0F8763EEFFFF      <1>       ja    sysret
39198                            <1>
39199                            <1>       ; esi = user buffer (in user's memory space)
39200 0000D64C 39CA              <1>       cmp   edx, ecx ; bottom-right > top-left ?
39201 0000D64E 0F8666FEFFFF      <1>         jna     sysvideo_41 ; full screen copy
39202                            <1>
39203 0000D654 0FB7C2            <1>       movzx eax, dx ; bottom right column
39204 0000D657 6629C8            <1>       sub   ax, cx  ; - top left column
39205 0000D65A 0F824FEEFFFF      <1>         jb      sysret ; invalid
39206 0000D660 6640              <1>       inc   ax ; same column no == 1 column
39207 0000D662 50                <1>       push  eax ; byte count per window row
39208                            <1>
39209 0000D663 52                <1>       push  edx
39210 0000D664 C1EB10            <1>       shr   ebx, 16 ; 320,640,800 : screen width
39211 0000D667 89C8              <1>       mov   eax, ecx
39212 0000D669 C1E810            <1>       shr   eax, 16      ; top row
39213 0000D66C F7E3              <1>       mul   ebx
39214 0000D66E 6689CA            <1>       mov   dx, cx ; top left column
39215 0000D671 01D0              <1>       add   eax, edx
39216 0000D673 01C7              <1>       add   edi, eax ; start address
39217 0000D675 59                <1>       pop   ecx ; edx
39218 0000D676 89C8              <1>       mov   eax, ecx
39219 0000D678 C1E810            <1>       shr   eax, 16 ; bottom row
39220 0000D67B F7E3              <1>       mul   ebx
39221 0000D67D 6689CA            <1>       mov   dx, cx ; bottom right column
39222 0000D680 01D0              <1>       add   eax, edx
39223 0000D682 5A                <1>       pop   edx ; byte count per window row
39224 0000D683 0500000A00        <1>       add   eax, 0A0000h
39225 0000D688 3DFFFF0A00        <1>       cmp   eax, 0AFFFFh
39226 0000D68D 0F871CEEFFFF      <1>       ja    sysret
39227 0000D693 57                <1>       push  edi ; start address
39228 0000D694 50                <1>       push  eax ; stop address (included)
39229                            <1> sysvideo_53:
39230 0000D695 89D1              <1>       mov   ecx, edx ; byte count
39231                            <1>       ; user to system video/display page window transfer
39232                            <1>       ; esi =      user buffer
39233 0000D697 E830120000        <1>       call  transfer_from_user_buffer ; fast transfer
39234 0000D69C 721F              <1>       jc    short sysvideo_54
39235 0000D69E 010D[64030600]    <1>       add   [u.r0], ecx
39236 0000D6A4 01DF              <1>       add   edi, ebx ; next row
39237 0000D6A6 01CE              <1>       add   esi, ecx
39238 0000D6A8 3B3C24            <1>       cmp   edi, [esp] ; stop addr(included in loop)
39239 0000D6AB 76E8              <1>       jna   short sysvideo_53
39240 0000D6AD 5B                <1>       pop   ebx ; stop address
39241 0000D6AE 5F                <1>       pop   edi ; start address
39242 0000D6AF 29FB              <1>       sub   ebx, edi
39243 0000D6B1 43                <1>       inc   ebx
39244 0000D6B2 891D[64030600]    <1>       mov   [u.r0], ebx
39245 0000D6B8 E9F2EDFFFF        <1>       jmp   sysret
39246                            <1> sysvideo_54:
39247 0000D6BD 58                <1>       pop   eax
39248 0000D6BE 5A                <1>       pop   edx
39249 0000D6BF E9EBEDFFFF        <1>       jmp   sysret
39250                            <1>
39251                            <1> sysvideo_55:
39252 0000D6C4 80FB06            <1>       cmp   bl, 6
39253 0000D6C7 0F8793000000      <1>         ja      sysvideo_58
39254                            <1>
39255                            <1>       ; BL = 6 = window copy (system to user)
39256                            <1>
39257 0000D6CD 89F7              <1>       mov   edi, esi ; user buffer
39258                            <1>
39259 0000D6CF B800000A00        <1>       mov   eax, 0A0000h
39260 0000D6D4 39C6              <1>       cmp   esi, eax
39261 0000D6D6 0F82D3EDFFFF      <1>       jb    sysret
```

```
39262 0000D6DC 6683C0FF          <1>       add    ax, 0FFFFh ; 65535
39263 0000D6E0 39C6              <1>       cmp    esi, eax
39264 0000D6E2 0F87C7EDFFFF      <1>       ja     sysret
39265                            <1>
39266                            <1>       ; edi = user buffer (in user's memory space)
39267 0000D6E8 39CA              <1>       cmp    edx, ecx ; bottom-right > top-left ?
39268 0000D6EA 0F86A2FAFFFF      <1>         jna     sysvideo_17 ; full screen copy
39269                            <1>
39270 0000D6F0 0FB7C2            <1>       movzx  eax, dx ; bottom right column
39271 0000D6F3 6629C8            <1>       sub    ax, cx  ; - top left column
39272 0000D6F6 0F82B3EDFFFF      <1>         jb      sysret ; invalid
39273 0000D6FC 6640              <1>       inc    ax ; same column no == 1 column
39274 0000D6FE 50                <1>       push   eax ; byte count per window row
39275                            <1>
39276 0000D6FF 52                <1>       push   edx
39277 0000D700 C1EB10            <1>       shr    ebx, 16 ; 320, 640,800 ; screen width
39278 0000D703 89C8              <1>       mov    eax, ecx
39279 0000D705 C1E810            <1>       shr    eax, 16      ; top row
39280 0000D708 F7E3              <1>       mul    ebx
39281 0000D70A 6689CA            <1>       mov    dx, cx ; top left column
39282 0000D70D 01D0              <1>       add    eax, edx
39283 0000D70F 01C6              <1>       add    esi, eax ; start address
39284 0000D711 59                <1>       pop    ecx ; edx
39285 0000D712 89C8              <1>       mov    eax, ecx
39286 0000D714 C1E810            <1>       shr    eax, 16 ; bottom row
39287 0000D717 F7E3              <1>       mul    ebx
39288 0000D719 6689CA            <1>       mov    dx, cx ; bottom right column
39289 0000D71C 01D0              <1>       add    eax, edx
39290 0000D71E 5A                <1>       pop    edx ; byte count per window row
39291 0000D71F 0500000A00        <1>       add    eax, 0A0000h
39292 0000D724 3DFFFF0A00        <1>       cmp    eax, 0AFFFFh
39293 0000D729 0F8780EDFFFF      <1>       ja     sysret
39294 0000D72F 56                <1>       push   esi ; start address
39295 0000D730 50                <1>       push   eax ; stop address (included)
39296                            <1> sysvideo_56:
39297 0000D731 89D1              <1>       mov    ecx, edx ; byte count
39298                            <1>       ; user to system video/display page window transfer
39299                            <1>       ; esi =      user buffer
39300 0000D733 E84A110000        <1>       call   transfer_to_user_buffer ; fast transfer
39301 0000D738 721F              <1>       jc     short sysvideo_57
39302 0000D73A 010D[64030600]    <1>       add    [u.r0], ecx
39303 0000D740 01DF              <1>       add    edi, ebx ; next row
39304 0000D742 01CE              <1>       add    esi, ecx
39305 0000D744 3B3C24            <1>       cmp    edi, [esp] ; stop addr(included in loop)
39306 0000D747 76E8              <1>       jna    short sysvideo_56
39307 0000D749 5B                <1>       pop    ebx ; stop address
39308 0000D74A 5F                <1>       pop    edi ; start address
39309 0000D74B 29FB              <1>       sub    ebx, edi
39310 0000D74D 43                <1>       inc    ebx
39311 0000D74E 891D[64030600]    <1>       mov    [u.r0], ebx
39312 0000D754 E956EDFFFF        <1>       jmp    sysret
39313                            <1> sysvideo_57:
39314 0000D759 58                <1>       pop    eax
39315 0000D75A 5A                <1>       pop    edx
39316 0000D75B E94FEDFFFF        <1>       jmp    sysret
39317                            <1>
39318                            <1> sysvideo_58:
39319 0000D760 80FB07            <1>       cmp    bl, 7
39320 0000D763 770F              <1>       ja     short sysvideo_60
39321                            <1>
39322                            <1>       ; BL = 7 = AND display page bytes with CL
39323                            <1>
39324 0000D765 BE00000A00        <1>       mov    esi, 0A0000h
39325 0000D76A B900000100        <1>       mov    ecx, 65536
39326                            <1> sysvideo_59:
39327 0000D76F 200E              <1>       and    byte [esi], cl
39328 0000D771 46                <1>       inc    esi
39329 0000D772 E2FB              <1>       loop   sysvideo_59
39330                            <1>
39331                            <1> sysvideo_60:
39332 0000D774 80FB08            <1>       cmp    bl, 8
39333 0000D777 770F              <1>       ja     short sysvideo_62
39334                            <1>
39335                            <1>       ; BL = 8 = OR display page bytes with CL
39336                            <1>
39337 0000D779 BE00000A00        <1>       mov    esi, 0A0000h
39338 0000D77E B900000100        <1>       mov    ecx, 65536
39339                            <1> sysvideo_61:
39340 0000D783 080E              <1>       or     byte [esi], cl
39341 0000D785 46                <1>       inc    esi
39342 0000D786 E2FB              <1>       loop   sysvideo_61
39343                            <1>
39344                            <1> sysvideo_62:
39345 0000D788 80FB09            <1>       cmp    bl, 9
39346 0000D78B 0F871EEDFFFF      <1>         ja      sysret ; nothing to do
39347                            <1>
39348                            <1>       ; BL = 9 = XOR display page bytes with CL
39349                            <1>
39350 0000D791 BE00000A00        <1>       mov    esi, 0A0000h
39351 0000D796 B900000100        <1>       mov    ecx, 65536
39352                            <1> sysvideo_63:
39353 0000D79B 300E              <1>       xor    byte [esi], cl
39354 0000D79D 46                <1>       inc    esi
39355 0000D79E E2FB              <1>       loop   sysvideo_63
39356                            <1>
39357                            <1> sysvideo_64:
39358 0000D7A0 80FF03            <1>       cmp    bh, 3
39359 0000D7A3 7464              <1>       je     short sysvideo_68
39360 0000D7A5 80FF04            <1>       cmp    bh, 4
39361 0000D7A8 7721              <1>       ja     short sysvideo_65
39362                            <1>
39363                            <1>       ; BH = 4
```

```
39364                                   <1>       ; Direct User Access for CGA video memory.
39365                                   <1>       ; Setup user's page tables for direct access to 0B8000h.
39366                                   <1>       ;
39367                                   <1>       ; Permission checks are not implemented yet !
39368                                   <1>       ; (11/07/2016)
39369                                   <1>
39370 0000D7AA B800800B00              <1>       mov    eax, 0B8000h
39371 0000D7AF B908000000              <1>       mov    ecx, 8 ; 8 pages (8*4K=32K)
39372 0000D7B4 89C3                    <1>       mov    ebx, eax ; 12/05/2017 ; virtual = physical
39373 0000D7B6 E8E57EFFFF              <1>       call   direct_memory_access
39374 0000D7BB 0F82EEECFFFF            <1>       jc     sysret
39375                                   <1>       ; eax = 0B8000h if there is not an error
39376 0000D7C1 A3[64030600]            <1>       mov    [u.r0], eax
39377 0000D7C6 E9E4ECFFFF              <1>       jmp    sysret
39378                                   <1>
39379                                   <1> sysvideo_65:
39380 0000D7CB 80FF05                  <1>       cmp    bh, 5
39381 0000D7CE 7721                    <1>       ja     short sysvideo_66
39382                                   <1>
39383                                   <1>       ; BH = 5
39384                                   <1>       ; Direct User Access for VGA video memory.
39385                                   <1>       ; Setup user's page tables for direct access to 0A0000h.
39386                                   <1>       ;
39387                                   <1>       ; Permission checks are not implemented yet !
39388                                   <1>       ; (11/07/2016)
39389                                   <1>
39390 0000D7D0 B800000A00              <1>       mov    eax, 0A0000h
39391 0000D7D5 B910000000              <1>       mov    ecx, 16 ; 16 pages (16*4K=64K)
39392 0000D7DA 89C3                    <1>       mov    ebx, eax ; 12/05/2017 ; virtual = physical
39393 0000D7DC E8BF7EFFFF              <1>       call   direct_memory_access
39394 0000D7E1 0F82C8ECFFFF            <1>       jc     sysret
39395                                   <1>       ; eax = 0A0000h if there is not an error
39396 0000D7E7 A3[64030600]            <1>       mov    [u.r0], eax
39397 0000D7EC E9BEECFFFF              <1>       jmp    sysret
39398                                   <1>
39399                                   <1> sysvideo_66:
39400 0000D7F1 80FF06                  <1>       cmp    bh, 6
39401 0000D7F4 7705                    <1>       ja     short sysvideo_67
39402                                   <1>       ; BH = 6
39403                                   <1>       ; Direct User Access for (Super VGA) Linear Frame Buffer.
39404                                   <1>       ; Setup user's page tables for direct access to LFB.
39405                                   <1>       ;
39406                                   <1>       ; Not implemented yet !
39407                                   <1>       ; (11/07/2016)
39408 0000D7F6 E9B4ECFFFF              <1>       jmp    sysret
39409                                   <1>
39410                                   <1> sysvideo_67:
39411 0000D7FB 80FF07                  <1>       cmp    bh, 7
39412 0000D7FE 0F87ABECFFFF            <1>       ja     sysret ; invalid !
39413                                   <1>
39414                                   <1>       ; BH = 7
39415                                   <1>       ; Get (Super/Extended VGA) Linear Frame Buffer info.
39416                                   <1>       ;
39417                                   <1>       ; Not implemented yet !
39418                                   <1>       ; (11/07/2016)
39419 0000D804 E9A6ECFFFF              <1>       jmp    sysret
39420                                   <1>
39421                                   <1> sysvideo_68:
39422                                   <1>       ; BH = 3
39423                                   <1>       ; Super VGA, LINEAR FRAME BUFFER data transfers
39424                                   <1>       ; Not implemented for yet ! (11/07/2016)
39425 0000D809 E9A1ECFFFF              <1>       jmp    sysret
39426                                   <1>
39427                                   <1> syslink:
39428                                   <1>       ; 13/01/2017 - TRDOS 386 (TRDOS v2.0)
39429                                   <1>       ; temporary !
39430 0000D80E B801000000              <1>       mov    eax, ERR_INV_FNUMBER ; 'invalid function number !'
39431 0000D813 A3[C8030600]            <1>         mov     [u.error], eax
39432 0000D818 A3[64030600]            <1>         mov     [u.r0], eax
39433 0000D81D E96DECFFFF              <1>       jmp    error
39434                                   <1>
39435                                   <1> isdir:
39436                                   <1>       ; 22/06/2015 (Retro UNIX 386 v1 - Beginning)
39437                                   <1>       ; 04/05/2013 - 02/08/2013 (Retro UNIX 8086 v1)
39438                                   <1>       ;
39439                                   <1>       ; 'isdir' check to see if the i-node whose i-number is in r1
39440                                   <1>       ;  is a directory. If it is, an error occurs, because 'isdir'
39441                                   <1>       ;  called by syslink and sysunlink to make sure directories
39442                                   <1>       ;  are not linked. If the user is the super user (u.uid=0),
39443                                   <1>       ; 'isdir' does not bother checking. The current i-node
39444                                   <1>       ;  is not disturbed.
39445                                   <1>       ;
39446                                   <1>       ; INPUTS ->
39447                                   <1>       ;    r1 - contains the i-number whose i-node is being checked.
39448                                   <1>       ;    u.uid - user id
39449                                   <1>       ; OUTPUTS ->
39450                                   <1>       ;    r1 - contains current i-number upon exit
39451                                   <1>       ;       (current i-node back in core)
39452                                   <1>       ;
39453                                   <1>       ; ((AX = R1))
39454                                   <1>       ;
39455                                   <1>         ; ((Modified registers: eAX, eDX, eBX, eCX, eSI, eDI, eBP))
39456                                   <1>       ;
39457                                   <1>
39458                                   <1>       ; / if the i-node whose i-number is in r1 is a directory
39459                                   <1>       ; / there is an error unless super user made the call
39460                                   <1>
39461 0000D822 803D[B0030600]00        <1>       cmp    byte [u.uid], 0
39462                                   <1>              ; tstb u.uid / super user
39463 0000D829 762D                    <1>       jna    short isdir1
39464                                   <1>              ; beq 1f / yes, don't care
39465 0000D82B 66FF35[51040600]        <1>       push   word [ii]
```

```
39466                                  <1>             ; mov ii,-(sp) / put current i-number on stack
39467 0000D832 E84E190000              <1>       call  iget
39468                                  <1>             ; jsr r0,iget / get i-node into core (i-number in r1)
39469 0000D837 66F705[00000600]00-     <1>       test  word [i.flgs], 4000h ; Bit 14 : Directory flag
39470 0000D83F 40                      <1>
39471                                  <1>             ; bit $40000,i.flgs / is it a directory
39472                                  <1>       ;jnz  error
39473                                  <1>             ; bne error9 / yes, error
39474 0000D840 740F                    <1>       jz    short isdir0
39475 0000D842 C705[C8030600]0B00-     <1>       mov   dword [u.error], ERR_NOT_FILE  ; 11 ; ERR_DIR_ACCESS
39476 0000D84A 0000                    <1>
39477                                  <1>                             ; 'permission denied !' error
39478                                  <1>       ; pop ax
39479 0000D84C E93EECFFFF              <1>       jmp   error
39480                                  <1> isdir0:
39481 0000D851 6658                    <1>       pop   ax
39482                                  <1>             ; mov (sp)+,r1 / no, put current i-number in r1 (ii)
39483 0000D853 E82D190000              <1>       call  iget
39484                                  <1>             ; jsr r0,iget / get it back in
39485                                  <1> isdir1: ; 1:
39486 0000D858 C3                      <1>       retn
39487                                  <1>             ; rts r0
39488                                  <1>
39489                                  <1> sysunlink:
39490                                  <1>       ; 13/01/2017 - TRDOS 386 (TRDOS v2.0)
39491                                  <1>       ; temporary !
39492 0000D859 B801000000              <1>       mov   eax, ERR_INV_FNUMBER ; 'invalid function number !'
39493 0000D85E A3[C8030600]            <1>         mov     [u.error], eax
39494 0000D863 A3[64030600]            <1>         mov     [u.r0], eax
39495 0000D868 E922ECFFFF              <1>       jmp   error
39496                                  <1> mkdir:
39497                                  <1>       ; 04/12/2015 (14 byte directory names)
39498                                  <1>       ; 12/10/2015
39499                                  <1>       ; 17/06/2015 (Retro UNIX 386 v1 - Beginning)
39500                                  <1>       ; 29/04/2013 - 01/08/2013 (Retro UNIX 8086 v1)
39501                                  <1>       ;
39502                                  <1>       ; 'mkdir' makes a directory entry from the name pointed to
39503                                  <1>       ; by u.namep into the current directory.
39504                                  <1>       ;
39505                                  <1>       ; INPUTS ->
39506                                  <1>       ;    u.namep - points to a file name
39507                                  <1>       ;                 that is about to be a directory entry.
39508                                  <1>       ;    ii - current directory's i-number.
39509                                  <1>       ; OUTPUTS ->
39510                                  <1>       ;    u.dirbuf+2 - u.dirbuf+10 - contains file name.
39511                                  <1>       ;    u.off - points to entry to be filled
39512                                  <1>       ;           in the current directory
39513                                  <1>       ;    u.base - points to start of u.dirbuf.
39514                                  <1>       ;    r1 - contains i-number of current directory
39515                                  <1>       ;
39516                                  <1>       ; ((AX = R1)) output
39517                                  <1>       ;
39518                                  <1>       ;    (Retro UNIX Prototype : 11/11/2012, UNIXCOPY.ASM)
39519                                  <1>         ;    ((Modified registers: eAX, eDX, eBX, eCX, eSI, eDI, eBP))
39520                                  <1>       ;
39521                                  <1>
39522                                  <1>       ; 17/06/2015 - 32 bit modifications (Retro UNIX 386 v1)
39523 0000D86D 31C0                    <1>       xor   eax, eax
39524 0000D86F BF[9A030600]            <1>       mov     edi, u.dirbuf+2
39525 0000D874 89FE                    <1>       mov   esi, edi
39526 0000D876 AB                      <1>       stosd
39527 0000D877 AB                      <1>       stosd
39528                                  <1>       ; 04/12/2015 (14 byte directory names)
39529 0000D878 AB                      <1>       stosd
39530 0000D879 66AB                    <1>       stosw
39531                                  <1>             ; jsr r0,copyz; u.dirbuf+2; u.dirbuf+10. / clear this
39532 0000D87B 89F7                    <1>       mov   edi, esi ; offset to u.dirbuf
39533                                  <1>       ; 12/10/2015 ([u.namep] -> ebp)
39534                                  <1>       ;mov  ebp, [u.namep]
39535 0000D87D E829040000              <1>       call  trans_addr_nmbp ; convert virtual address to physical
39536                                  <1>             ; esi = physical address (page start + offset)
39537                                  <1>             ; ecx = byte count in the page (1 - 4096)
39538                                  <1>       ; edi = offset to u.dirbuf (edi is not modified in trans_addr_nm)
39539                                  <1>             ; mov u.namep,r2 / r2 points to name of directory entry
39540                                  <1>             ; mov $u.dirbuf+2,r3 / r3 points to u.dirbuf+2
39541                                  <1> mkdir_1: ; 1:
39542 0000D882 45                      <1>       inc   ebp ; 12/10/2015
39543                                  <1>       ;
39544                                  <1>       ; / put characters in the directory name in u.dirbuf+2 - u.dirbuf+10
39545                                  <1>        ; 01/08/2013
39546 0000D883 AC                      <1>       lodsb
39547                                  <1>             ; movb (r2)+,r1 / move character in name to r1
39548 0000D884 20C0                    <1>       and   al, al
39549 0000D886 7427                    <1>       jz    short mkdir_3
39550                                  <1>             ; beq 1f / if null, done
39551 0000D888 3C2F                    <1>       cmp   al, '/'
39552                                  <1>             ; cmp r1,$'/ / is it a "/"?
39553 0000D88A 7414                    <1>       je    short mkdir_err
39554                                  <1>       ;je   error
39555                                  <1>             ; beq error9 / yes, error
39556                                  <1>       ; 12/10/2015
39557 0000D88C 6649                    <1>       dec   cx
39558 0000D88E 7505                    <1>       jnz   short mkdir_2
39559                                  <1>       ; 12/10/2015 ([u.namep] -> ebp)
39560 0000D890 E81C040000              <1>       call  trans_addr_nm ; convert virtual address to physical
39561                                  <1>             ; esi = physical address (page start + offset)
39562                                  <1>             ; ecx = byte count in the page
39563                                  <1>       ; edi = offset to u.dirbuf (edi is not modified in trans_addr_nm)
39564                                  <1> mkdir_2:
39565 0000D895 81FF[A8030600]          <1>       cmp     edi, u.dirbuf+16 ; ; 04/12/2015 (10 -> 16)
39566                                  <1>             ; cmp r3,$u.dirbuf+10. / have we reached the last slot for
39567                                  <1>                                ; / a char?
```

```
39568 0000D89B 74E5                <1>       je     short mkdir_1
39569                              <1>              ; beq 1b / yes, go back
39570 0000D89D AA                  <1>       stosb
39571                              <1>              ; movb r1,(r3)+ / no, put the char in the u.dirbuf
39572 0000D89E EBE2                <1>       jmp    short mkdir_1
39573                              <1>              ; br 1b / get next char
39574                              <1> mkdir_err:
39575                              <1>       ; 17/06/2015
39576 0000D8A0 C705[C8030600]1300- <1>       mov    dword [u.error], ERR_NOT_DIR ; 'not a valid directory !'
39577 0000D8A8 0000                <1>
39578 0000D8AA E9E0EBFFFF          <1>       jmp    error
39579                              <1>
39580                              <1> mkdir_3: ; 1:
39581 0000D8AF A1[78030600]        <1>       mov    eax, [u.dirp]
39582 0000D8B4 A3[80030600]        <1>       mov    [u.off], eax
39583                              <1>              ; mov u.dirp,u.off / pointer to empty current directory
39584                              <1>                                 ; / slot to u.off
39585                              <1> wdir: ; 29/04/2013
39586 0000D8B9 C705[84030600]-     <1>         mov     dword [u.base], u.dirbuf
39587 0000D8BF [98030600]          <1>
39588                              <1>              ; mov $u.dirbuf,u.base / u.base points to created file name
39589 0000D8C3 C705[88030600]1000- <1>         mov     dword [u.count], 16 ; 04/12/2015 (10 -> 16)
39590 0000D8CB 0000                <1>
39591                              <1>              ; mov $10.,u.count / u.count = 10
39592 0000D8CD 66A1[51040600]      <1>       mov    ax, [ii]
39593                              <1>              ; mov ii,r1 / r1 has i-number of current directory
39594 0000D8D3 B201                <1>       mov    dl, 1 ; owner flag mask ; RETRO UNIX 8086 v1 modification !
39595 0000D8D5 E8B1180000          <1>       call   access
39596                              <1>              ; jsr r0,access; 1 / get i-node and set its file up
39597                              <1>                                 ; / for writing
39598                              <1>       ; AX = i-number of current directory
39599                              <1>       ; 01/08/2013
39600 0000D8DA FE05[C6030600]      <1>       inc     byte [u.kcall] ; the caller is 'mkdir' sign
39601 0000D8E0 E8FF110000          <1>       call   writei
39602                              <1>              ; jsr r0,writei / write into directory
39603 0000D8E5 C3                  <1>       retn
39604                              <1>              ; rts r0
39605                              <1>
39606                              <1> sysexec:
39607                              <1>       ; 04/01/2017
39608                              <1>       ; 24/10/2016
39609                              <1>       ; 24/04/2016 - TRDOS 386 (TRDOS v2.0)
39610                              <1>       ; 23/06/2015 - 23/10/2015 (Retro UNIX 386 v1)
39611                              <1>       ; 03/06/2013 - 06/12/2013 (Retro UNIX 8086 v1)
39612                              <1>       ;
39613                              <1>       ; 'sysexec' initiates execution of a file whose path name if
39614                              <1>       ; pointed to by 'name' in the sysexec call.
39615                              <1>       ; 'sysexec' performs the following operations:
39616                              <1>       ;    1. obtains i-number of file to be executed via 'namei'.
39617                              <1>       ;    2. obtains i-node of file to be exceuted via 'iget'.
39618                              <1>       ;    3. sets trap vectors to system routines.
39619                              <1>       ;    4. loads arguments to be passed to executing file into
39620                              <1>       ;      highest locations of user's core
39621                              <1>       ;    5. puts pointers to arguments in locations immediately
39622                              <1>       ;      following arguments.
39623                              <1>       ;    6.      saves number of arguments in next location.
39624                              <1>       ;    7. intializes user's stack area so that all registers
39625                              <1>       ;      will be zeroed and the PS is cleared and the PC set
39626                              <1>       ;      to core when 'sysret' restores registers
39627                              <1>       ;      and does an rti.
39628                              <1>       ;    8. inializes u.r0 and u.sp
39629                              <1>       ;    9. zeros user's core down to u.r0
39630                              <1>       ;   10.      reads executable file from storage device into core
39631                              <1>       ;      starting at location 'core'.
39632                              <1>       ;   11.      sets u.break to point to end of user's code with
39633                              <1>       ;      data area appended.
39634                              <1>       ;   12.      calls 'sysret' which returns control at location
39635                              <1>       ;      'core' via 'rti' instruction.
39636                              <1>       ;
39637                              <1>       ; Calling sequence:
39638                              <1>       ;     sysexec; namep; argp
39639                              <1>       ; Arguments:
39640                              <1>       ;     namep - points to pathname of file to be executed
39641                              <1>       ;     argp  - address of table of argument pointers
39642                              <1>       ;     argp1... argpn - table of argument pointers
39643                              <1>       ;     argp1:<...0> ... argpn:<...0> - argument strings
39644                              <1>       ; Inputs: (arguments)
39645                              <1>       ; Outputs: -
39646                              <1>       ; ..........................................................
39647                              <1>       ;
39648                              <1>       ; Retro UNIX 386 v1 modification:
39649                              <1>       ;     User application runs in it's own virtual space
39650                              <1>       ;     which is izolated from kernel memory (and other
39651                              <1>       ;     memory pages) via 80386   paging in ring 3
39652                              <1>       ;     privilige mode. Virtual start address is always 0.
39653                              <1>       ;     User's core memory starts at linear address 400000h
39654                              <1>       ;     (the end of the 1st 4MB).
39655                              <1>       ;
39656                              <1>       ; Retro UNIX 8086 v1 modification:
39657                              <1>       ;     user/application segment and system/kernel segment
39658                              <1>       ;     are different and sysenter/sysret/sysrele routines
39659                              <1>       ;     are different (user's registers are saved to
39660                              <1>       ;     and then restored from system's stack.)
39661                              <1>       ;
39662                              <1>       ;     NOTE: Retro UNIX 8086 v1 'arg2' routine gets these
39663                              <1>       ;           arguments which were in these registers;
39664                              <1>       ;           but, it returns by putting the 1st argument
39665                              <1>       ;           in 'u.namep' and the 2nd argument
39666                              <1>       ;           on top of stack. (1st argument is offset of the
39667                              <1>       ;           file/path name in the user's program segment.)
39668                              <1>
39669                              <1>       ;call  arg2
```

```
39670                                  <1>       ; * name - 'u.namep' points to address of file/path name
39671                                  <1>       ;          in the user's program segment ('u.segmnt')
39672                                  <1>       ;          with offset in BX register (as sysopen argument 1).
39673                                  <1>       ; * argp - sysexec argument 2 is in CX register
39674                                  <1>       ;          which is on top of stack.
39675                                  <1>       ;
39676                                  <1>             ; jsr r0,arg2 / arg0 in u.namep,arg1 on top of stack
39677                                  <1>
39678                                  <1>       ; 23/06/2015 (32 bit modifications)
39679                                  <1>
39680 0000D8E6 891D[7C030600]          <1>       mov    [u.namep], ebx ; argument 1
39681                                  <1>         ; 18/10/2015
39682 0000D8EC 890D[4C040600]          <1>       mov     [argv], ecx  ; * ; argument 2
39683 0000D8F2 E86F020000              <1>       call   namei
39684                                  <1>             ; jsr r0,namei / namei returns i-number of file
39685                                  <1>                         ; / named in sysexec call in r1
39686                                  <1>       ;jc    error
39687                                  <1>             ; br error9
39688 0000D8F7 731E                    <1>       jnc    short sysexec_0
39689                                  <1>       ;
39690                                  <1>       ; 'file not found !' error
39691 0000D8F9 C705[C8030600]0C00-     <1>       mov    dword [u.error], ERR_FILE_NOT_FOUND
39692 0000D901 0000                    <1>
39693 0000D903 E987EBFFFF              <1>       jmp    error
39694                                  <1> sysexec_not_exf:
39695                                  <1>       ; 'not executable file !' error
39696 0000D908 C705[C8030600]1600-     <1>       mov    dword [u.error], ERR_NOT_EXECUTABLE
39697 0000D910 0000                    <1>
39698 0000D912 E978EBFFFF              <1>       jmp    error
39699                                  <1> sysexec_0:
39700 0000D917 E869180000              <1>       call   iget
39701                                  <1>             ; jsr r0,iget / get i-node for file to be executed
39702 0000D91C 66F705[00000600]10-     <1>         test    word [i.flgs], 10h
39703 0000D924 00                      <1>
39704                                  <1>             ; bit $20,i.flgs / is file executable
39705 0000D925 74E1                    <1>       jz     short sysexec_not_exf
39706                                  <1>       ;jz    error
39707                                  <1>             ; beq error9
39708                                  <1>       ;;
39709 0000D927 E85B180000              <1>       call   iopen
39710                                  <1>             ; jsr r0,iopen / gets i-node for file with i-number
39711                                  <1>                         ; / given in r1 (opens file)
39712                                  <1>       ; AX = i-number of the file
39713 0000D92C 66F705[00000600]20-     <1>       test   word [i.flgs], 20h
39714 0000D934 00                      <1>
39715                                  <1>             ; bit $40,i.flgs / test user id on execution bit
39716 0000D935 7415                    <1>       jz     short sysexec_1
39717                                  <1>             ; beq 1f
39718 0000D937 803D[B0030600]00        <1>       cmp    byte [u.uid], 0 ; 02/08/2013
39719                                  <1>             ; tstb u.uid / test user id
39720 0000D93E 760C                    <1>       jna    short sysexec_1
39721                                  <1>             ; beq 1f / super user
39722 0000D940 8A0D[03000600]          <1>       mov    cl, [i.uid]
39723 0000D946 880D[B0030600]          <1>       mov    [u.uid], cl ; 02/08/2013
39724                                  <1>             ; movb i.uid,u.uid / put user id of owner of file
39725                                  <1>                               ; / as process user id
39726                                  <1> sysexec_1:
39727                                  <1>       ; 18/10/2215
39728                                  <1>       ; 10/10/2015
39729                                  <1>       ; 24/07/2015
39730                                  <1>       ; 21/07/2015
39731                                  <1>       ; 25/06/2015
39732                                  <1>       ; 24/06/2015
39733                                  <1>         ; Moving arguments to the end of [u.upage]
39734                                  <1>       ; (by regarding page borders in user's memory space)
39735                                  <1>       ;
39736                                  <1>       ; 10/10/2015
39737                                  <1>       ; 21/07/2015
39738 0000D94C 89E5                    <1>       mov    ebp, esp ; (**)
39739                                  <1>       ; 18/10/2015
39740 0000D94E 89EF                    <1>       mov    edi, ebp
39741 0000D950 B900010000              <1>       mov    ecx, MAX_ARG_LEN ; 256
39742                                  <1>       ;sub   edi, MAX_ARG_LEN ; 256
39743 0000D955 29CF                    <1>       sub    edi, ecx
39744 0000D957 89FC                    <1>       mov    esp, edi
39745 0000D959 31C0                    <1>       xor    eax, eax
39746 0000D95B A3[8C030600]            <1>       mov    [u.nread], eax ; 0
39747 0000D960 49                      <1>       dec    ecx ; 256 - 1
39748 0000D961 890D[88030600]          <1>       mov    [u.count], ecx ; MAX_ARG_LEN - 1 ; 255
39749                                  <1>       ;mov   dword [u.count], MAX_ARG_LEN - 1 ; 255
39750                                  <1> sysexec_2:
39751 0000D967 8B35[4C040600]          <1>       mov    esi, [argv] ; 18/10/2015
39752 0000D96D E866000000              <1>       call   get_argp
39753 0000D972 B904000000              <1>       mov    ecx, 4 ; mov ecx, 4
39754                                  <1> sysexec_3:
39755 0000D977 21C0                    <1>       and    eax, eax
39756 0000D979 0F84E3070000            <1>         jz      sysexec_6
39757                                  <1>       ; 18/10/2015
39758 0000D97F 010D[4C040600]          <1>       add    [argv], ecx ; 4
39759 0000D985 66FF05[4A040600]        <1>       inc    word [argc]
39760                                  <1>       ;
39761 0000D98C A3[84030600]            <1>       mov    [u.base], eax
39762                                  <1>       ; 23/10/2015
39763 0000D991 66C705[C4030600]00-     <1>       mov    word [u.pcount], 0
39764 0000D999 00                      <1>
39765                                  <1> sysexec_4:
39766 0000D99A E8830E0000              <1>       call   cpass ; get a character from user's core memory
39767 0000D99F 750E                    <1>         jnz      short sysexec_5
39768                                  <1>             ; (max. 255 chars + null)
39769                                  <1>       ; 18/10/2015
39770 0000D9A1 28C0                    <1>       sub    al, al
39771 0000D9A3 AA                      <1>       stosb
```

```
39772 0000D9A4 FF05[8C030600]      <1>       inc    dword [u.nread]
39773 0000D9AA E9B3070000          <1>       jmp    sysexec_6 ; 24/04/2016
39774                              <1> sysexec_5:
39775 0000D9AF AA                  <1>       stosb
39776 0000D9B0 20C0                <1>       and    al, al
39777 0000D9B2 75E6                <1>       jnz    short sysexec_4
39778 0000D9B4 B904000000          <1>       mov    ecx, 4
39779 0000D9B9 390D[48040600]      <1>       cmp    [ncount], ecx ; 4
39780 0000D9BF 72A6                <1>       jb     short sysexec_2
39781 0000D9C1 8B35[44040600]      <1>       mov    esi, [nbase]
39782 0000D9C7 010D[44040600]      <1>       add    [nbase], ecx ; 4
39783 0000D9CD 66290D[48040600]    <1>       sub    [ncount], cx
39784 0000D9D4 8B06                <1>       mov    eax, [esi]
39785 0000D9D6 EB9F                <1>       jmp    short sysexec_3
39786                              <1>
39787                              <1> get_argp:
39788                              <1>       ; 18/10/2015 (nbase, ncount)
39789                              <1>       ; 21/07/2015
39790                              <1>       ; 24/06/2015 (Retro UNIX 386 v1)
39791                              <1>       ; Get (virtual) address of argument from user's core memory
39792                              <1>       ;
39793                              <1>       ; INPUT:
39794                              <1>       ;      esi = virtual address of argument pointer
39795                              <1>       ; OUTPUT:
39796                              <1>       ;      eax = virtual address of argument
39797                              <1>       ;
39798                              <1>       ; Modified registers: EAX, EBX, ECX, EDX, ESI
39799                              <1>       ;
39800 0000D9D8 833D[BC030600]00    <1>       cmp     dword [u.ppgdir], 0 ; /etc/init ?
39801                              <1>                                   ; (the caller is kernel)
39802 0000D9DF 7667                <1>         jna     short get_argpk
39803                              <1>       ;
39804 0000D9E1 89F3                <1>       mov    ebx, esi
39805 0000D9E3 E8A678FFFF          <1>       call   get_physical_addr ; get physical address
39806 0000D9E8 0F8289000000        <1>         jc      get_argp_err
39807 0000D9EE A3[44040600]        <1>       mov    [nbase], eax ; physical address
39808 0000D9F3 66890D[48040600]    <1>       mov    [ncount], cx ; remain byte count in page (1-4096)
39809 0000D9FA B804000000          <1>       mov    eax, 4 ; 21/07/2015
39810 0000D9FF 6639C1              <1>       cmp    cx, ax ; 4
39811 0000DA02 735D                <1>       jnb    short get_argp2
39812 0000DA04 89F3                <1>       mov    ebx, esi
39813 0000DA06 01CB                <1>       add    ebx, ecx
39814 0000DA08 E88178FFFF          <1>       call   get_physical_addr ; get physical address
39815 0000DA0D 7268                <1>       jc     short get_argp_err
39816                              <1>       ;push  esi
39817 0000DA0F 89C6                <1>       mov    esi, eax
39818 0000DA11 66870D[48040600]    <1>       xchg   cx, [ncount]
39819 0000DA18 8735[44040600]      <1>       xchg   esi, [nbase]
39820 0000DA1E B504                <1>       mov    ch, 4
39821 0000DA20 28CD                <1>       sub    ch, cl
39822                              <1> get_argp0:
39823 0000DA22 AC                  <1>       lodsb
39824 0000DA23 6650                <1>       push   ax
39825 0000DA25 FEC9                <1>       dec    cl
39826 0000DA27 75F9                <1>         jnz     short get_argp0
39827 0000DA29 8B35[44040600]      <1>       mov    esi, [nbase]
39828                              <1>       ; 21/07/2015
39829 0000DA2F 0FB6C5              <1>       movzx  eax, ch
39830 0000DA32 0105[44040600]      <1>       add    [nbase], eax
39831 0000DA38 662905[48040600]    <1>       sub    [ncount], ax
39832                              <1> get_argp1:
39833 0000DA3F AC                  <1>       lodsb
39834 0000DA40 FECD                <1>       dec    ch
39835 0000DA42 743D                <1>         jz      short get_argp3
39836 0000DA44 6650                <1>         push ax
39837 0000DA46 EBF7                <1>       jmp     short get_argp1
39838                              <1> get_argpk:
39839                              <1>       ; Argument is in kernel's memory space
39840 0000DA48 66C705[48040600]00- <1>       mov    word [ncount], PAGE_SIZE ; 4096
39841 0000DA50 10                  <1>
39842 0000DA51 8935[44040600]      <1>       mov    [nbase], esi
39843 0000DA57 8305[44040600]04    <1>       add    dword [nbase], 4
39844 0000DA5E 8B06                <1>       mov    eax, [esi] ; virtual addr. = physcal addr.
39845 0000DA60 C3                  <1>       retn
39846                              <1> get_argp2:
39847                              <1>       ; 21/07/2015
39848                              <1>       ;mov   eax, 4
39849 0000DA61 8B15[44040600]      <1>       mov    edx, [nbase] ; 18/10/2015
39850 0000DA67 0105[44040600]      <1>       add    [nbase], eax
39851 0000DA6D 662905[48040600]    <1>       sub    [ncount], ax
39852                              <1>       ;
39853 0000DA74 8B02                <1>       mov    eax, [edx]
39854 0000DA76 C3                  <1>       retn
39855                              <1> get_argp_err:
39856 0000DA77 A3[C8030600]        <1>       mov    [u.error], eax
39857 0000DA7C E90EEAFFFF          <1>       jmp    error
39858                              <1> get_argp3:
39859 0000DA81 B103                <1>       mov    cl, 3
39860                              <1> get_argp4:
39861 0000DA83 C1E008              <1>       shl    eax, 8
39862 0000DA86 665A                <1>       pop    dx
39863 0000DA88 88D0                <1>       mov    al, dl
39864 0000DA8A E2F7                <1>         loop    get_argp4
39865                              <1>       ;pop   esi
39866 0000DA8C C3                  <1>       retn
39867                              <1>
39868                              <1> sysstat:
39869                              <1>       ; 13/01/2017 - TRDOS 386 (TRDOS v2.0)
39870                              <1>       ; temporary !
39871 0000DA8D B801000000          <1>       mov    eax, ERR_INV_FNUMBER ; 'invalid function number !'
39872 0000DA92 A3[C8030600]        <1>         mov     [u.error], eax
39873 0000DA97 A3[64030600]        <1>         mov     [u.r0], eax
```

```
39874 0000DA9C E9EEE9FFFF          <1>       jmp    error
39875                              <1>
39876                              <1> sysfstat:
39877                              <1>       ; 13/01/2017 - TRDOS 386 (TRDOS v2.0)
39878                              <1>       ; temporary !
39879 0000DAA1 B801000000          <1>       mov    eax, ERR_INV_FNUMBER ; 'invalid function number !'
39880 0000DAA6 A3[C8030600]        <1>         mov     [u.error], eax
39881 0000DAAB A3[64030600]        <1>         mov     [u.r0], eax
39882 0000DAB0 E9DAE9FFFF          <1>       jmp    error
39883                              <1>
39884                              <1> fclose:
39885                              <1>       ; 06/10/2016 (TRDOS 386 = TRDOS v2.0)
39886                              <1>       ;
39887                              <1>       ; 18/06/2015 (Retro UNIX 386 v1 - Beginning)
39888                              <1>       ;            (32 bit offset pointer modification)
39889                              <1>       ; 19/04/2013 - 12/01/2014 (Retro UNIX 8086 v1)
39890                              <1>       ;
39891                              <1>       ; Given the file descriptor (index to the u.fp list)
39892                              <1>       ; 'fclose' first gets the i-number of the file via 'getf'.
39893                              <1>       ; If i-node is active (i-number > 0) the entry in
39894                              <1>       ; u.fp list is cleared. If all the processes that opened
39895                              <1>       ; that file close it, then fsp etry is freed and the file
39896                              <1>       ; is closed. If not a return is taken.
39897                              <1>       ; If the file has been deleted while open, 'anyi' is called
39898                              <1>       ; to see anyone else has it open, i.e., see if it is appears
39899                              <1>       ; in another entry in the fsp table. Upon return from 'anyi'
39900                              <1>       ; a check is made to see if the file is special.
39901                              <1>       ;
39902                              <1>       ; INPUTS ->
39903                              <1>       ;    r1 - contains the file descriptor (value=0,1,2...)
39904                              <1>       ;    u.fp - list of entries in the fsp table
39905                              <1>       ;    fsp - table of entries (4 words/entry) of open files.
39906                              <1>       ; OUTPUTS ->
39907                              <1>       ;    r1 - contains the same file descriptor
39908                              <1>       ;    r2 - contains i-number
39909                              <1>       ;
39910                              <1>       ; ((AX = R1))
39911                              <1>       ; ((Modified registers: eDX, eBX, eCX, eSI, eDI, eBP))
39912                              <1>       ;
39913                              <1>       ; Retro UNIX 8086 v1 modification : CF = 1
39914                              <1>       ;             if i-number of the file is 0. (error)
39915                              <1>       ;
39916                              <1>       ; TRDOS 386 (06/10/2016)
39917                              <1>       ;
39918                              <1>       ; INPUT:
39919                              <1>       ;     EAX = File Handle (File Descriptor, File Index)
39920                              <1>       ;
39921                              <1>       ; OUTPUT:
39922                              <1>       ;     CF = 1 -> File not open !
39923                              <1>       ;     CF = 0 -> OK!
39924                              <1>       ;          EBX = File Number (System)
39925                              <1>       ;          [cdev] = Logical DOS Drive Number
39926                              <1>       ;          EAX = File Handle/Number (user)
39927                              <1>       ;
39928                              <1>       ; Modified Registers: EBX
39929                              <1>
39930 0000DAB5 50                  <1>       push   eax ; File handle
39931                              <1>
39932 0000DAB6 E846000000          <1>       call   getf
39933 0000DABB 0F82921F0000        <1>       jc     device_close ; eax = device number
39934                              <1>
39935 0000DAC1 80BB[F25E0100]01    <1>       cmp    byte [ebx+OF_MODE], 1 ; open mode ; 0 = empty entry
39936 0000DAC8 722E                <1>       jb     short fclose_1            ; 1 = read, 2 = write
39937                              <1>
39938 0000DACA 83F801              <1>       cmp    eax, 1 ; is the first cluster number > 0
39939 0000DACD 7229                <1>       jb     short fclose_1 ; no, this is empty entry
39940                              <1>
39941                              <1> fclose_0:
39942 0000DACF FE8B[065F0100]      <1>       dec    byte [ebx+OF_OPENCOUNT] ; decrement the number of processes
39943                              <1>                                      ; that have opened the file
39944 0000DAD5 7921                <1>       jns    short fclose_1 ; jump if not negative (jump if bit 7 is 0)
39945                              <1>                       ; if all processes haven't closed the file, return
39946                              <1>       ;
39947                              <1>       ; eax ; First cluster
39948 0000DAD7 31C0                <1>       xor    eax, eax ; 0
39949 0000DAD9 8883[F25E0100]      <1>       mov    [ebx+OF_MODE], al ; 0 = empty entry
39950                              <1>       ;mov   [ebx+OF_STATUS], al ; 0 = empty entry
39951 0000DADF 66C1E302            <1>       shl    bx, 2
39952 0000DAE3 8983[C05E0100]      <1>       mov    [ebx+OF_FCLUSTER], eax ; 0
39953 0000DAE9 8983[D85F0100]      <1>       mov    [ebx+OF_CCLUSTER], eax ; 0
39954                              <1>       ;mov   [ebx+OF_CCINDEX], eax ; 0
39955 0000DAEF A3[74030600]        <1>       mov    [u.fofp], eax ; 0
39956 0000DAF4 66C1EB02            <1>       shr    bx, 2
39957                              <1> fclose_1: ; 1:
39958 0000DAF8 58                  <1>       pop    eax ; File handle (File Descriptor, File Index)
39959 0000DAF9 C680[6A030600]00    <1>       mov    byte [eax+u.fp], 0 ; clear that entry in the u.fp list
39960 0000DB00 C3                  <1>       retn
39961                              <1>
39962                              <1> getf:
39963                              <1>       ; 12/10/2016
39964                              <1>       ; 11/10/2016
39965                              <1>       ; 08/10/2016
39966                              <1>       ; 06/10/2016 (TRDOS 386 = TRDOS v2.0)
39967                              <1>       ; / get the device number and the i-number of an open file
39968                              <1>       ; 13/05/2015
39969                              <1>       ; 11/05/2015 (Retro UNIX 386 v1 - Beginning)
39970                              <1>       ; 19/04/2013 - 18/11/2013 (Retro UNIX 8086 v1)
39971                              <1>       ;
39972 0000DB01 89C3                <1>       mov    ebx, eax
39973                              <1> getf1:
39974 0000DB03 83FB0A              <1>       cmp    ebx, 10
39975 0000DB06 730A                <1>         jnb  short getf2
```

```
39976 0000DB08 8A9B[6A030600]      <1>       mov    bl, [ebx+u.fp]
39977 0000DB0E 08DB                <1>       or     bl, bl
39978 0000DB10 7503                <1>       jnz    short getf3
39979                              <1> getf2:
39980                              <1>       ; 'File not open !' error (ax=0)
39981 0000DB12 29C0                <1>       sub    eax, eax
39982 0000DB14 C3                  <1>       retn
39983                              <1> getf3:
39984 0000DB15 F6C380              <1>       test   bl, 80h
39985 0000DB18 7530                <1>       jnz    short getf5 ; device
39986 0000DB1A FECB                <1>       dec    bl ; 0 based
39987 0000DB1C 8A83[E85E0100]      <1>       mov    al, [ebx+OF_DRIVE]
39988 0000DB22 A2[46030600]        <1>       mov    [cdev], al
39989 0000DB27 C0E302              <1>       shl    bl, 2 ; *4 (dword offset)
39990 0000DB2A 8B83[385F0100]      <1>       mov    eax, [ebx+OF_SIZE]
39991 0000DB30 A3[55040600]        <1>       mov    [i.size], eax ; file size
39992 0000DB35 8D83[105F0100]      <1>       lea    eax, [ebx+OF_POINTER] ;12/10/2016
39993 0000DB3B A3[74030600]        <1>       mov    [u.fofp], eax
39994 0000DB40 8B83[C05E0100]      <1>       mov    eax, [ebx+OF_FCLUSTER]
39995 0000DB46 C0EB02              <1>       shr    bl, 2 ; /4 (byte offset)
39996                              <1> getf4:
39997 0000DB49 C3                  <1>       retn
39998                              <1> getf5:
39999                              <1>       ; get device number
40000 0000DB4A 80E37F              <1>       and    bl, 7Fh ; 1 to 7Fh
40001 0000DB4D FECB                <1>       dec    bl ; 0 based (0 to 7Eh)
40002 0000DB4F 8A83[1A5D0100]      <1>       mov    al, [ebx+DEV_DRIVER]
40003 0000DB55 8AAB[845C0100]      <1>       mov    ch, [ebx+DEV_ACCESS]
40004 0000DB5B 8A8B[385D0100]      <1>       mov    cl, [ebx+DEV_OPENMODE]
40005 0000DB61 80E5FE              <1>       and    ch, 0FEh ; reset bit 0 ; dev_close
40006 0000DB64 F9                  <1>       stc ; cf = 1
40007 0000DB65 C3                  <1>       retn
40008                              <1>
40009                              <1> namei:
40010                              <1>       ; 04/12/2015 (14 byte file names)
40011                              <1>       ; 18/10/2015 (nbase, ncount)
40012                              <1>       ; 12/10/2015
40013                              <1>       ; 21/08/2015
40014                              <1>       ; 18/07/2015
40015                              <1>       ; 02/07/2015
40016                              <1>       ; 17/06/2015
40017                              <1>       ; 16/06/2015 (Retro UNIX 386 v1 - Beginning)
40018                              <1>       ; 24/04/2013 - 31/07/2013 (Retro UNIX 8086 v1)
40019                              <1>       ;
40020                              <1>       ; 'namei' takes a file path name and returns i-number of
40021                              <1>       ; the file in the current directory or the root directory
40022                              <1>       ; (if the first character of the pathname is '/').
40023                              <1>       ;
40024                              <1>       ; INPUTS ->
40025                              <1>       ;    u.namep - points to a file path name
40026                              <1>       ;    u.cdir - i-number of users directory
40027                              <1>       ;    u.cdev - device number on which user directory resides
40028                              <1>       ; OUTPUTS ->
40029                              <1>       ;    r1 - i-number of file
40030                              <1>       ;    cdev
40031                              <1>       ;    u.dirbuf - points to directory entry where a match
40032                              <1>       ;               occurs in the search for file path name.
40033                              <1>       ;              If no match u.dirb points to the end of
40034                              <1>       ;               the directory and r1 = i-number of the current
40035                              <1>       ;              directory.
40036                              <1>       ; ((AX = R1))
40037                              <1>       ;
40038                              <1>       ; (Retro UNIX Prototype : 07/10/2012 - 05/01/2013, UNIXCOPY.ASM)
40039                              <1>         ; ((Modified registers: eDX, eBX, eCX, eSI, eDI, eBP))
40040                              <1>       ;
40041                              <1>
40042 0000DB66 66A1[68030600]      <1>       mov    ax, [u.cdir]
40043                              <1>              ; mov u.cdir,r1 / put the i-number of current directory
40044                              <1>                             ; / in r1
40045 0000DB6C 668B15[AE030600]    <1>       mov    dx, [u.cdrv]
40046 0000DB73 668915[46030600]    <1>       mov    [cdev], dx       ; NOTE: Retro UNIX 8086 v1
40047                              <1>                               ; device/drive number is in 1 byte,
40048                              <1>                               ; not in 1 word!
40049                              <1>              ; mov u.cdev,cdev / device number for users directory
40050                              <1>                             ; / into cdev
40051                              <1>       ; 12/10/2015
40052                              <1>       ; 16/06/2015 - 32 bit modifications (Retro UNIX 386 v1)
40053                              <1>              ; convert virtual (pathname) addr to physical address
40054 0000DB7A E82C010000          <1>       call    trans_addr_nmbp ; 12/10/2015
40055                              <1>              ; esi = physical address of [u.namep]
40056                              <1>              ; ecx = byte count in the page
40057 0000DB7F 803E2F              <1>       cmp    byte [esi], '/'
40058                              <1>              ; cmpb *u.namep,$'/ / is first char in file name a /
40059 0000DB82 751E                <1>       jne    short namei_1
40060                              <1>              ; bne 1f
40061 0000DB84 FF05[7C030600]      <1>       inc    dword [u.namep]
40062                              <1>              ; inc u.namep / go to next char
40063 0000DB8A 6649                <1>       dec    cx ; remain byte count in the page
40064 0000DB8C 7506                <1>       jnz    short namei_0
40065                              <1>       ; 12/10/2015
40066 0000DB8E E818010000          <1>       call   trans_addr_nmbp ; convert virtual address to physical
40067                              <1>              ; esi = physical address (page start + offset)
40068                              <1>              ; ecx = byte count in the page
40069 0000DB93 4E                  <1>       dec    esi
40070                              <1> namei_0:
40071 0000DB94 46                  <1>       inc    esi  ; go to next char
40072 0000DB95 66A1[50030600]      <1>       mov    ax, [rootdir] ; 09/07/2013
40073                              <1>              ; mov rootdir,r1 / put i-number of rootdirectory in r1
40074 0000DB9B C605[46030600]00    <1>       mov    byte [cdev], 0
40075                              <1>              ; clr cdev / clear device number
40076                              <1> namei_1: ; 1:
40077 0000DBA2 F606FF              <1>       test   byte [esi], 0FFh
```

```
40078 0000DBA5 74A2                <1>       jz     short getf4
40079                              <1>       ;jz      nig
40080                              <1>              ; tstb *u.namep / is the character in file name a nul
40081                              <1>              ; beq nig / yes, end of file name reached;
40082                              <1>                   ; / branch to "nig"
40083                              <1> namei_2: ; 1:
40084                              <1>       ; 18/10/2015
40085 0000DBA7 8935[44040600]      <1>       mov    [nbase], esi
40086 0000DBAD 66890D[48040600]    <1>       mov    [ncount], cx
40087                              <1>       ;
40088                              <1>       ;mov   dx, 2
40089 0000DBB4 B202                <1>       mov    dl, 2 ; user flag (read, non-owner)
40090 0000DBB6 E8D0150000          <1>       call   access
40091                              <1>              ; jsr r0,access; 2 / get i-node with i-number r1
40092                              <1>       ; 'access' will not return here if user has not "r" permission !
40093 0000DBBB 66F705[00000600]00- <1>       test   word [i.flgs], 4000h
40094 0000DBC3 40                  <1>
40095                              <1>              ; bit $40000,i.flgs / directory i-node?
40096 0000DBC4 746A                <1>         jz      short namei_err
40097                              <1>              ; beq error3 / no, got an error
40098                              <1>       ; 16/06/2015 - 32 bit modifications (Retro UNIX 386 v1)
40099 0000DBC6 31C0                <1>       xor    eax, eax
40100 0000DBC8 A3[80030600]        <1>       mov    [u.off], eax ; 0
40101 0000DBCD 66A1[55040600]      <1>       mov    ax, [i.size]
40102 0000DBD3 A3[78030600]        <1>       mov    [u.dirp], eax
40103                              <1>              ; mov i.size,u.dirp / put size of directory in u.dirp
40104                              <1>              ; clr u.off / u.off is file offset used by user
40105 0000DBD8 C705[74030600]-     <1>       mov    dword [u.fofp], u.off
40106 0000DBDE [80030600]          <1>
40107                              <1>              ; mov $u.off,u.fofp / u.fofp is a pointer to
40108                              <1>                                  ; / the offset portion of fsp entry
40109                              <1> namei_3: ; 2:
40110 0000DBE2 C705[84030600]-     <1>       mov    dword [u.base], u.dirbuf
40111 0000DBE8 [98030600]          <1>
40112                              <1>              ; mov $u.dirbuf,u.base / u.dirbuf holds a file name
40113                              <1>                                     ; / copied from a directory
40114 0000DBEC C705[88030600]1000- <1>       mov    dword [u.count], 16 ; 04/12/2015 (10 -> 16)
40115 0000DBF4 0000                <1>
40116                              <1>              ; mov $10.,u.count / u.count is byte count
40117                              <1>                                 ; / for reads and writes
40118 0000DBF6 66A1[51040600]      <1>       mov    ax, [ii]
40119                              <1>       ; 31/07/2013 ('namei_r') - 16/06/2015 ('u.kcall')
40120 0000DBFC FE05[C6030600]      <1>       inc     byte [u.kcall] ; the caller is 'namei' sign
40121 0000DC02 E8A2070000          <1>       call   readi
40122                              <1>              ; jsr r0,readi / read 10. bytes of file
40123                              <1>                   ; with i-number (r1); i.e. read a directory entry
40124 0000DC07 8B0D[8C030600]      <1>       mov    ecx, [u.nread]
40125 0000DC0D 09C9                <1>       or     ecx, ecx
40126                              <1>              ; tst u.nread
40127 0000DC0F 741B                <1>       jz     short nib
40128                              <1>              ; ble nib / gives error return
40129                              <1>       ;
40130 0000DC11 668B1D[98030600]    <1>       mov    bx, [u.dirbuf]
40131 0000DC18 6621DB              <1>       and    bx, bx
40132                              <1>              ; tst u.dirbuf /
40133 0000DC1B 7522                <1>       jnz    short namei_4
40134                              <1>              ; bne 3f / branch when active directory entry
40135                              <1>                     ; / (i-node word in entry non zero)
40136 0000DC1D A1[80030600]        <1>       mov    eax, [u.off]
40137 0000DC22 83E810              <1>       sub    eax, 16 ; 04/12/2015 (10 -> 16)
40138 0000DC25 A3[78030600]        <1>       mov    [u.dirp], eax
40139                              <1>              ; mov u.off,u.dirp
40140                              <1>              ; sub $10.,u.dirp
40141 0000DC2A EBB6                <1>       jmp    short namei_3
40142                              <1>              ; br 2b
40143                              <1>
40144                              <1>       ; 18/07/2013
40145                              <1> nib:
40146 0000DC2C 31C0                <1>       xor    eax, eax  ; xor ax, ax ; ax = 0 -> file not found
40147 0000DC2E F9                  <1>       stc
40148                              <1> nig:
40149 0000DC2F C3                  <1>       retn
40150                              <1>
40151                              <1> namei_err:
40152                              <1>       ; 16/06/2015
40153 0000DC30 C705[C8030600]1300- <1>       mov    dword [u.error], ERR_NOT_DIR ; 'not a directory !' error
40154 0000DC38 0000                <1>
40155 0000DC3A E950E8FFFF          <1>       jmp    error
40156                              <1>
40157                              <1> namei_4: ; 3:
40158                              <1>       ; 18/10/2015
40159                              <1>       ; 12/10/2015
40160                              <1>       ; 21/08/2015
40161                              <1>       ; 18/07/2015
40162 0000DC3F 8B2D[7C030600]      <1>       mov    ebp, [u.namep]
40163                              <1>              ; mov u.namep,r2 / u.namep points into a file name string
40164 0000DC45 BF[9A030600]        <1>       mov    edi, u.dirbuf + 2
40165                              <1>              ; mov $u.dirbuf+2,r3 / points to file name of directory entry
40166                              <1>       ; 18/10/2015
40167 0000DC4A 8B35[44040600]      <1>       mov    esi, [nbase]
40168 0000DC50 668B0D[48040600]    <1>       mov    cx, [ncount]
40169                              <1>       ;
40170 0000DC57 6621C9              <1>       and    cx, cx
40171 0000DC5A 7505                <1>       jnz    short namei_5
40172                              <1>       ;
40173 0000DC5C E850000000          <1>       call   trans_addr_nm ; convert virtual address to physical
40174                              <1>              ; esi = physical address (page start + offset)
40175                              <1>              ; ecx = byte count in the page
40176                              <1> namei_5: ; 3:
40177 0000DC61 45                  <1>       inc    ebp ; 18/07/2015
40178 0000DC62 AC                  <1>       lodsb   ; mov al, [esi] ; inc esi (al = r4)
40179                              <1>              ; movb (r2)+,r4 / move a character from u.namep string into r4
```

```
40180 0000DC63 08C0               <1>       or     al, al
40181 0000DC65 741D               <1>       jz     short namei_7
40182                             <1>              ; beq 3f / if char is nul, then the last char in string
40183                             <1>                     ; / has been moved
40184 0000DC67 3C2F               <1>       cmp    al, '/'
40185                             <1>              ; cmp r4,$'/ / is char a </>
40186 0000DC69 7419               <1>       je     short namei_7
40187                             <1>              ; beq 3f
40188                             <1>       ; 12/10/2015
40189 0000DC6B 6649               <1>       dec    cx ; remain byte count in the page
40190 0000DC6D 7505               <1>       jnz    short namei_6
40191 0000DC6F E83D000000         <1>       call   trans_addr_nm ; convert virtual address to physical
40192                             <1>              ; esi = physical address (page start + offset)
40193                             <1>              ; ecx = byte count in the page
40194                             <1> namei_6:
40195 0000DC74 81FF[A8030600]     <1>         cmp     edi, u.dirbuf + 16 ; 04/12/2015 (10 -> 16)
40196                             <1>              ; cmp r3,$u.dirbuf+10. / have I checked
40197                             <1>                              ; / all 8 bytes of file name
40198 0000DC7A 74E5               <1>       je     short namei_5
40199                             <1>              ; beq 3b
40200 0000DC7C AE                 <1>       scasb
40201                             <1>              ; cmpb (r3)+,r4 / compare char in u.namep string to file name
40202                             <1>                           ; / char read from directory
40203 0000DC7D 74E2               <1>       je     short namei_5
40204                             <1>              ; beq 3b / branch if chars match
40205                             <1>
40206 0000DC7F E95EFFFFFF         <1>         jmp    namei_3 ; 2b
40207                             <1>              ; br 2b / file names do not match go to next directory entry
40208                             <1> namei_7: ; 3:
40209 0000DC84 81FF[A8030600]     <1>       cmp    edi, u.dirbuf + 16 ; 04/12/2015 (10 -> 16)
40210                             <1>              ; cmp r3,$u.dirbuf+10. / if equal all 8 bytes were matched
40211 0000DC8A 740A               <1>       je     short namei_8
40212                             <1>              ; beq 3f
40213 0000DC8C 8A27               <1>       mov    ah, [edi]
40214                             <1>       ;inc   edi
40215 0000DC8E 20E4               <1>       and    ah, ah
40216                             <1>              ; tstb (r3)+ /
40217 0000DC90 0F854CFFFFFF       <1>         jnz     namei_3
40218                             <1>              ; bne 2b
40219                             <1> namei_8: ; 3
40220 0000DC96 892D[7C030600]     <1>       mov    [u.namep], ebp ; 18/07/2015
40221                             <1>              ; mov r2,u.namep / u.namep points to char
40222                             <1>                              ; / following a / or nul
40223                             <1>       ;mov   bx, [u.dirbuf]
40224                             <1>              ; mov u.dirbuf,r1 / move i-node number in directory
40225                             <1>                             ; / entry to r1
40226 0000DC9C 20C0               <1>       and    al, al
40227                             <1>              ; tst r4 / if r4 = 0 the end of file name reached,
40228                             <1>                     ;  / if r4 = </> then go to next directory
40229                             <1>       ; mov  ax, bx
40230 0000DC9E 66A1[98030600]     <1>       mov    ax, [u.dirbuf] ; 17/06/2015
40231 0000DCA4 0F85FDFEFFFF       <1>         jnz     namei_2
40232                             <1>              ; bne 1b
40233                             <1>       ; AX = i-number of the file
40234                             <1> ;;nig:
40235 0000DCAA C3                 <1>       retn
40236                             <1>              ; tst (r0)+ / gives non-error return
40237                             <1> ;;nib:
40238                             <1> ;;    xor    ax, ax ; Retro UNIX 8086 v1 modification !
40239                             <1>                     ; ax = 0 -> file not found
40240                             <1> ;;    stc    ; 27/05/2013
40241                             <1> ;;    retn
40242                             <1>              ; rts r0
40243                             <1>
40244                             <1> trans_addr_nmbp:
40245                             <1>       ; 18/10/2015
40246                             <1>       ; 12/10/2015
40247 0000DCAB 8B2D[7C030600]     <1>       mov    ebp, [u.namep]
40248                             <1> trans_addr_nm:
40249                             <1>       ; Convert virtual (pathname) address to physical address
40250                             <1>       ; (Retro UNIX 386 v1 feature only !)
40251                             <1>       ; 18/10/2015
40252                             <1>       ; 12/10/2015 (u.pnbase & u.pncount has been removed from code)
40253                             <1>       ; 02/07/2015
40254                             <1>       ; 17/06/2015
40255                             <1>       ; 16/06/2015
40256                             <1>       ;
40257                             <1>       ; INPUTS:
40258                             <1>       ;     ebp = pathname address (virtual) ; [u.namep]
40259                             <1>       ;     [u.pgdir] = user's page directory
40260                             <1>       ; OUTPUT:
40261                             <1>       ;       esi = physical address of the pathname
40262                             <1>       ;     ecx = remain byte count in the page
40263                             <1>       ;
40264                             <1>       ; (Modified registers: EAX, EBX, ECX, EDX, ESI)
40265                             <1>       ;
40266 0000DCB1 833D[BC030600]00   <1>         cmp     dword [u.ppgdir], 0  ; /etc/init ? (sysexec)
40267 0000DCB8 7618               <1>       jna    short trans_addr_nmk ; the caller is os kernel;
40268                             <1>                                   ; it is already physical address
40269 0000DCBA 50                 <1>       push   eax
40270 0000DCBB 89EB               <1>       mov    ebx, ebp ; [u.namep] ; pathname address (virtual)
40271 0000DCBD E8CC75FFFF         <1>              call   get_physical_addr ; get physical address
40272 0000DCC2 7204               <1>       jc     short tr_addr_nm_err
40273                             <1>       ; 18/10/2015
40274                             <1>       ; eax = physical address
40275                             <1>       ; cx = remain byte count in page (1-4096)
40276                             <1>              ; 12/10/2015 (cx = [u.pncount])
40277 0000DCC4 89C6               <1>       mov    esi, eax ; 12/10/2015 (esi=[u.pnbase])
40278 0000DCC6 58                 <1>       pop    eax
40279 0000DCC7 C3                 <1>       retn
40280                             <1>
40281                             <1> tr_addr_nm_err:
```

```
40282 0000DCC8 A3[C8030600]        <1>       mov   [u.error], eax
40283                              <1>       ;pop  eax
40284 0000DCCD E9BDE7FFFF          <1>       jmp   error
40285                              <1>
40286                              <1> trans_addr_nmk:
40287                              <1>       ; 12/10/2015
40288                              <1>       ; 02/07/2015
40289 0000DCD2 8B35[7C030600]      <1>       mov   esi, [u.namep]  ; [u.pnbase]
40290 0000DCD8 66B90010            <1>       mov   cx, PAGE_SIZE ; 4096 ; [u.pncount]
40291 0000DCDC C3                  <1>       retn
40292                              <1>
40293                              <1> syschdir:
40294                              <1>       ; / makes the directory specified in the argument
40295                              <1>       ; / the current directory
40296                              <1>       ;
40297                              <1>       ; 23/06/2015 (Retro UNIX 386 v1 - Beginning)
40298                              <1>       ; 19/06/2013 (Retro UNIX 8086 v1)
40299                              <1>       ;
40300                              <1>       ; 'syschdir' makes the directory specified in its argument
40301                              <1>       ; the current working directory.
40302                              <1>       ;
40303                              <1>       ; Calling sequence:
40304                              <1>       ;     syschdir; name
40305                              <1>       ; Arguments:
40306                              <1>       ;     name - address of the path name of a directory
40307                              <1>       ;            terminated by nul byte.
40308                              <1>       ; Inputs: -
40309                              <1>       ; Outputs: -
40310                              <1>       ; ..........................................................
40311                              <1>       ;
40312                              <1>       ; Retro UNIX 8086 v1 modification:
40313                              <1>       ;      The user/application program puts address of
40314                              <1>       ;      the path name in BX register as 'syschdir'
40315                              <1>       ;      system call argument.
40316                              <1>
40317 0000DCDD 891D[7C030600]      <1>       mov   [u.namep], ebx
40318                              <1>             ;jsr r0,arg; u.namep / u.namep points to path name
40319 0000DCE3 E87EFEFFFF          <1>       call  namei
40320                              <1>             ; jsr r0,namei / find its i-number
40321                              <1>       ;jc   error
40322                              <1>             ; br error3
40323 0000DCE8 730F                <1>       jnc   short syschdir0
40324                              <1>       ; 'directory not found !' error
40325 0000DCEA C705[C8030600]0C00- <1>       mov   dword [u.error], ERR_DIR_NOT_FOUND ; 12
40326 0000DCF2 0000                <1>
40327 0000DCF4 E996E7FFFF          <1>       jmp   error
40328                              <1> syschdir0:
40329 0000DCF9 E88D140000          <1>       call  access
40330                              <1>             ; jsr r0,access; 2 / get i-node into core
40331 0000DCFE 66F705[00000600]00- <1>       test  word [i.flgs], 4000h
40332 0000DD06 40                  <1>
40333                              <1>             ; bit $40000,i.flgs / is it a directory?
40334                              <1>       ;jz   error
40335                              <1>             ; beq error3 / no error
40336 0000DD07 750F                <1>       jnz   short syschdir1
40337 0000DD09 C705[C8030600]1300- <1>       mov   dword [u.error], ERR_NOT_DIR ; 'not a valid directory !'
40338 0000DD11 0000                <1>
40339 0000DD13 E977E7FFFF          <1>       jmp   error
40340                              <1> syschdir1:
40341 0000DD18 66A3[68030600]      <1>       mov   [u.cdir], ax
40342                              <1>             ; mov r1,u.cdir / move i-number to users
40343                              <1>                               ; / current directory
40344 0000DD1E 66A1[46030600]      <1>       mov   ax, [cdev]
40345 0000DD24 66A3[AE030600]      <1>       mov   [u.cdrv], ax
40346                              <1>             ; mov cdev,u.cdev / move its device to users
40347                              <1>                                 ; / current device
40348 0000DD2A E980E7FFFF          <1>       jmp   sysret
40349                              <1>             ; br sysret3
40350                              <1>
40351                              <1> syschmod: ; < change mode of file >
40352                              <1>       ; 13/01/2017 - TRDOS 386 (TRDOS v2.0)
40353                              <1>       ; temporary !
40354 0000DD2F B801000000          <1>       mov   eax, ERR_INV_FNUMBER ; 'invalid function number !'
40355 0000DD34 A3[C8030600]        <1>         mov     [u.error], eax
40356 0000DD39 A3[64030600]        <1>         mov     [u.r0], eax
40357 0000DD3E E94CE7FFFF          <1>       jmp   error
40358                              <1>
40359                              <1> isown:
40360                              <1>       ; 22/06/2015 (Retro UNIX 386 v1 - Beginning)
40361                              <1>       ; 04/05/2013 - 07/07/2013 (Retro UNIX 8086 v1)
40362                              <1>       ;
40363                              <1>       ; 'isown' is given a file name (the 1st argument).
40364                              <1>       ;  It find the i-number of that file via 'namei'
40365                              <1>       ;  then gets the i-node into core via 'iget'.
40366                              <1>       ;  It then tests to see if the user is super user.
40367                              <1>       ;  If not, it cheks to see if the user is owner of
40368                              <1>       ;  the file. If he is not an error occurs.
40369                              <1>       ;  If user is the owner 'setimod' is called to indicate
40370                              <1>       ;  the inode has been modificed and the 2nd argument of
40371                              <1>       ;  the call is put in r2.
40372                              <1>       ;
40373                              <1>       ; INPUTS ->
40374                              <1>       ;    arguments of syschmod and syschown calls
40375                              <1>       ; OUTPUTS ->
40376                              <1>       ;    u.uid - id of user
40377                              <1>       ;    imod - set to a 1
40378                              <1>       ;    r2 - contains second argument of the system call
40379                              <1>       ;
40380                              <1>       ;   ((AX=R2) output as 2nd argument)
40381                              <1>       ;
40382                              <1>         ; ((Modified registers: eAX, eDX, eBX, eCX, eSI, eDI, eBP))
40383                              <1>       ;
```

```
40384                                  <1>             ; jsr r0,arg2 / u.namep points to file name
40385                                  <1>       ;; ! 2nd argument on top of stack !
40386                                  <1>       ;; 22/06/2015 - 32 bit modifications
40387                                  <1>       ;; 07/07/2013
40388 0000DD43 891D[7C030600]          <1>       mov   [u.namep], ebx ;; 1st argument
40389 0000DD49 51                      <1>       push  ecx ;; 2nd argument
40390                                  <1>       ;;
40391 0000DD4A E817FEFFFF              <1>       call  namei
40392                                  <1>             ; jsr r0,namei / get its i-number
40393                                  <1>        ; Retro UNIX 8086 v1 modification !
40394                                  <1>        ; ax = 0 -> file not found
40395                                  <1>       ;and   ax, ax
40396                                  <1>       ;jz    error
40397                                  <1>       ;jc    error ; 27/05/2013
40398                                  <1>             ; br error3
40399 0000DD4F 730F                    <1>       jnc   short isown0
40400                                  <1>       ; 'file not found !' error
40401 0000DD51 C705[C8030600]0C00-     <1>       mov   dword [u.error], ERR_FILE_NOT_FOUND ; 12
40402 0000DD59 0000                    <1>
40403 0000DD5B E92FE7FFFF              <1>       jmp   error
40404                                  <1> isown0:
40405 0000DD60 E820140000              <1>       call  iget
40406                                  <1>             ; jsr r0,iget / get i-node into core
40407 0000DD65 A0[B0030600]            <1>       mov   al, [u.uid] ; 02/08/2013
40408 0000DD6A 08C0                    <1>       or    al, al
40409                                  <1>             ; tstb u.uid / super user?
40410 0000DD6C 7417                    <1>       jz    short isown1
40411                                  <1>             ; beq 1f / yes, branch
40412 0000DD6E 3A05[03000600]          <1>       cmp   al, [i.uid]
40413                                  <1>             ; cmpb i.uid,u.uid / no, is this the owner of
40414                                  <1>                               ; / the file
40415                                  <1>       ;jne   error
40416                                  <1>             ; beq 1f / yes
40417                                  <1>             ; jmp error3 / no, error
40418 0000DD74 740F                    <1>       je    short isown1
40419                                  <1>
40420 0000DD76 C705[C8030600]0B00-     <1>       mov   dword [u.error], ERR_NOT_OWNER  ; 11
40421 0000DD7E 0000                    <1>
40422                                  <1>                     ;  'permission denied !' error
40423 0000DD80 E90AE7FFFF              <1>       jmp   error
40424                                  <1> isown1: ; 1:
40425 0000DD85 E8FF130000              <1>       call  setimod
40426                                  <1>             ; jsr r0,setimod / indicates
40427                                  <1>             ;            ; / i-node has been modified
40428 0000DD8A 58                      <1>       pop   eax ; 2nd argument
40429                                  <1>             ; mov (sp)+,r2 / mode is put in r2
40430                                  <1>                    ; / (u.off put on stack with 2nd arg)
40431 0000DD8B C3                      <1>       retn
40432                                  <1>             ; rts r0
40433                                  <1>
40434                                  <1> ;;arg:  ; < get system call arguments >
40435                                  <1>       ; 'arg' extracts an argument for a routine whose call is
40436                                  <1>       ; of form:
40437                                  <1>       ;     sys 'routine' ; arg1
40438                                  <1>       ;           or
40439                                  <1>       ;     sys 'routine' ; arg1 ; arg2
40440                                  <1>       ;           or
40441                                  <1>       ;     sys 'routine' ; arg1;...;arg10 (sys exec)
40442                                  <1>       ;
40443                                  <1>       ; INPUTS ->
40444                                  <1>       ;    u.sp+18 - contains a pointer to one of arg1..argn
40445                                  <1>       ;     This pointers's value is actually the value of
40446                                  <1>       ;     update pc at the the trap to sysent (unkni) is
40447                                  <1>       ;     made to process the sys instruction
40448                                  <1>       ;    r0 - contains the return address for the routine
40449                                  <1>       ;     that called arg. The data in the word pointer
40450                                  <1>       ;     to by the return address is used as address
40451                                  <1>       ;     in which the extracted argument is stored
40452                                  <1>       ;
40453                                  <1>       ; OUTPUTS ->
40454                                  <1>       ;    'address' - contains the extracted argument
40455                                  <1>       ;    u.sp+18 - is incremented by 2
40456                                  <1>       ;    r1 - contains the extracted argument
40457                                  <1>       ;    r0 - points to the next instruction to be
40458                                  <1>       ;       executed in the calling routine.
40459                                  <1>       ;
40460                                  <1>
40461                                  <1>       ; mov u.sp,r1
40462                                  <1>       ; mov *18.(r1),*(r0)+ / put argument of system call
40463                                  <1>                    ; / into argument of arg2
40464                                  <1>       ; add $2,18.(r1) / point pc on stack
40465                                  <1>                          ; / to next system argument
40466                                  <1>       ; rts r0
40467                                  <1>
40468                                  <1> ;;arg2: ; < get system calls arguments - with file name pointer>
40469                                  <1>       ; 'arg2' takes first argument in system call
40470                                  <1>       ;  (pointer to name of the file) and puts it in location
40471                                  <1>       ;  u.namep; takes second argument and puts it in u.off
40472                                  <1>       ;  and on top of the stack
40473                                  <1>       ;
40474                                  <1>       ; INPUTS ->
40475                                  <1>       ;    u.sp, r0
40476                                  <1>       ;
40477                                  <1>       ; OUTPUTS ->
40478                                  <1>       ;    u.namep
40479                                  <1>       ;    u.off
40480                                  <1>       ;    u.off pushed on stack
40481                                  <1>       ;    r1
40482                                  <1>       ;
40483                                  <1>
40484                                  <1>       ; jsr  r0,arg; u.namep / u.namep contains value of
40485                                  <1>                           ; / first arg in sys call
```

```
40486                                  <1>       ; jsr r0,arg; u.off / u.off contains value of
40487                                  <1>                         ; / second arg in sys call
40488                                  <1>       ; mov r0,r1 / r0 points to calling routine
40489                                  <1>       ; mov (sp),r0 / put operation code back in r0
40490                                  <1>       ; mov u.off,(sp) / put pointer to second argument
40491                                  <1>                  ; / on stack
40492                                  <1>       ; jmp (r1) / return to calling routine
40493                                  <1>
40494                                  <1> syschown: ; < change owner of file >
40495                                  <1>       ; 23/06/2015 (Retro UNIX 386 v1 - Beginning)
40496                                  <1>       ; 20/06/2013 - 02/08/2013 (Retro UNIX 8086 v1)
40497                                  <1>       ;
40498                                  <1>       ; 'syschown' changes the owner of the file whose name is given
40499                                  <1>       ; as null terminated string pointed to by 'name' has it's owner
40500                                  <1>       ; changed to 'owner'
40501                                  <1>       ;
40502                                  <1>       ; Calling sequence:
40503                                  <1>       ;     syschown; name; owner
40504                                  <1>       ; Arguments:
40505                                  <1>       ;     name - address of the file name
40506                                  <1>       ;            terminated by null byte.
40507                                  <1>       ;     owner - (new) owner (number/ID)
40508                                  <1>       ;
40509                                  <1>       ; Inputs: -
40510                                  <1>       ; Outputs: -
40511                                  <1>       ; ..........................................................
40512                                  <1>       ;
40513                                  <1>       ; Retro UNIX 8086 v1 modification:
40514                                  <1>       ;      'syschown' system call has two arguments; so,
40515                                  <1>       ;     * 1st argument, name is pointed to by BX register
40516                                  <1>       ;     * 2nd argument, owner number is in CX register
40517                                  <1>       ;
40518                                  <1>       ; / name; owner
40519 0000DD8C E8B2FFFFFF              <1>       call   isown
40520                                  <1>              ; jsr r0,isown / get the i-node and check user status
40521 0000DD91 803D[B0030600]00        <1>       cmp    byte [u.uid], 0 ; 02/08/2013
40522                                  <1>              ; tstb u.uid / super user
40523 0000DD98 7418                    <1>       jz     short syschown1
40524                                  <1>              ; beq 2f / yes, 2f
40525 0000DD9A F605[00000600]20        <1>         test    byte [i.flgs], 20h ; 32
40526                                  <1>              ; bit $40,i.flgs / no, set userid on execution?
40527                                  <1>       ;jnz   error
40528                                  <1>              ; bne 3f / yes error, could create Trojan Horses
40529 0000DDA1 740F                    <1>       jz     short syschown1
40530                                  <1>       ; 'permission denied !'
40531 0000DDA3 C705[C8030600]0B00-     <1>       mov    dword [u.error], ERR_FILE_ACCESS  ; 11
40532 0000DDAB 0000                    <1>
40533 0000DDAD E9DDE6FFFF              <1>       jmp    error
40534                                  <1> syschown1: ; 2:
40535                                  <1>       ; AL = owner (number/ID)
40536 0000DDB2 A2[03000600]            <1>       mov    [i.uid], al ; 23/06/2015
40537                                  <1>              ;  movb      r2,i.uid / no, put the new owners id
40538                                  <1>                           ; / in the i-node
40539 0000DDB7 E9F3E6FFFF              <1>       jmp    sysret
40540                                  <1>       ; 1:
40541                                  <1>              ; jmp sysret4
40542                                  <1>       ; 3:
40543                                  <1>              ; jmp  error
40544                                  <1>
40545                                  <1> systime: ; / get time of year
40546                                  <1>       ; 23/06/2015 (Retro UNIX 386 v1 - Beginning)
40547                                  <1>       ; 20/06/2013 (Retro UNIX 8086 v1)
40548                                  <1>       ;
40549                                  <1>       ; 20/06/2013
40550                                  <1>       ; 'systime' gets the time of the year.
40551                                  <1>       ; The present time is put on the stack.
40552                                  <1>       ;
40553                                  <1>       ; Calling sequence:
40554                                  <1>       ;     systime
40555                                  <1>       ; Arguments: -
40556                                  <1>       ;
40557                                  <1>       ; Inputs: -
40558                                  <1>       ; Outputs: sp+2, sp+4 - present time
40559                                  <1>       ; ..........................................................
40560                                  <1>       ;
40561                                  <1>       ; Retro UNIX 8086 v1 modification:
40562                                  <1>       ;      'systime' system call will return to the user
40563                                  <1>       ;     with unix time (epoch) in DX:AX register pair
40564                                  <1>       ;
40565                                  <1>       ;     !! Major modification on original Unix v1 'systime'
40566                                  <1>       ;     system call for PC compatibility !!
40567                                  <1>
40568 0000DDBC E8CB130000              <1>       call   epoch
40569 0000DDC1 A3[64030600]            <1>       mov    [u.r0], eax
40570                                  <1>              ; mov s.time,4(sp)
40571                                  <1>              ; mov s.time+2,2(sp) / put the present time
40572                                  <1>                                   ; / on the stack
40573                                  <1>              ; br sysret4
40574 0000DDC6 E9E4E6FFFF              <1>       jmp    sysret
40575                                  <1>
40576                                  <1> sysstime: ; / set time
40577                                  <1>       ; 23/06/2015 (Retro UNIX 386 v1 - Beginning)
40578                                  <1>       ; 20/06/2013 - 02/08/2013 (Retro UNIX 8086 v1)
40579                                  <1>       ;
40580                                  <1>       ; 'sysstime' sets the time. Only super user can use this call.
40581                                  <1>       ;
40582                                  <1>       ; Calling sequence:
40583                                  <1>       ;     sysstime
40584                                  <1>       ; Arguments: -
40585                                  <1>       ;
40586                                  <1>       ; Inputs: sp+2, sp+4 - time system is to be set to.
40587                                  <1>       ; Outputs: -
```

```
40588                                  <1>       ; ..........................................................
40589                                  <1>       ;
40590                                  <1>       ; Retro UNIX 8086 v1 modification:
40591                                  <1>       ;     the user calls 'sysstime' with unix (epoch) time
40592                                  <1>       ;     (to be set) is in CX:BX register pair as two arguments.
40593                                  <1>       ;
40594                                  <1>       ;     Retro UNIX 8086 v1 argument transfer method 2 is used
40595                                  <1>       ;     to get sysstime system call arguments from the user;
40596                                  <1>       ;     * 1st argument, lowword of unix time is in BX register
40597                                  <1>       ;     * 2nd argument, highword of unix time is in CX register
40598                                  <1>       ;
40599                                  <1>       ;     !! Major modification on original Unix v1 'sysstime'
40600                                  <1>       ;     system call for PC compatibility !!
40601                                  <1>
40602 0000DDCB 803D[B0030600]00       <1>       cmp   byte [u.uid], 0
40603                                  <1>             ; tstb u.uid / is user the super user
40604                                  <1>       ;ja   error
40605                                  <1>             ; bne error4 / no, error
40606 0000DDD2 760F                  <1>       jna   short systime1
40607                                  <1>       ; 'permission denied !'
40608 0000DDD4 C705[C8030600]0B00-    <1>       mov   dword [u.error], ERR_NOT_SUPERUSER  ; 11
40609 0000DDDC 0000                  <1>
40610 0000DDDE E9ACE6FFFF            <1>       jmp   error
40611                                  <1> systime1:
40612                                  <1>       ; 23/06/2015 (Retro UNIX 386 v1 - 32 bit version)
40613                                  <1>       ; EBX = unix (epoch) time (from user)
40614 0000DDE3 89D8                  <1>       mov   eax, ebx
40615 0000DDE5 E8A4130000            <1>       call  set_date_time
40616                                  <1>             ; mov 4(sp),s.time
40617                                  <1>             ; mov 2(sp),s.time+2 / set the system time
40618 0000DDEA E9C0E6FFFF            <1>       jmp   sysret
40619                                  <1>             ; br sysret4
40620                                  <1>
40621                                  <1> sysbreak:
40622                                  <1>       ; 18/10/2015
40623                                  <1>       ; 07/10/2015
40624                                  <1>       ; 23/06/2015 (Retro UNIX 386 v1 - Beginning)
40625                                  <1>       ; 20/06/2013 - 24/03/2014 (Retro UNIX 8086 v1)
40626                                  <1>       ;
40627                                  <1>       ; 'sysbreak' sets the programs break points.
40628                                  <1>       ; It checks the current break point (u.break) to see if it is
40629                                  <1>       ; between "core" and the stack (sp). If it is, it is made an
40630                                  <1>       ; even address (if it was odd) and the area between u.break
40631                                  <1>       ; and the stack is cleared. The new breakpoint is then put
40632                                  <1>       ; in u.break and control is passed to 'sysret'.
40633                                  <1>       ;
40634                                  <1>       ; Calling sequence:
40635                                  <1>       ;     sysbreak; addr
40636                                  <1>       ; Arguments: -
40637                                  <1>       ;
40638                                  <1>       ; Inputs: u.break - current breakpoint
40639                                  <1>       ; Outputs: u.break - new breakpoint
40640                                  <1>       ;     area between old u.break and the stack (sp) is cleared.
40641                                  <1>       ; ..........................................................
40642                                  <1>       ;
40643                                  <1>       ; Retro UNIX 8086 v1 modification:
40644                                  <1>       ;     The user/application program puts breakpoint address
40645                                  <1>       ;       in BX register as 'sysbreak' system call argument.
40646                                  <1>       ;     (argument transfer method 1)
40647                                  <1>       ;
40648                                  <1>       ;  NOTE: Beginning of core is 0 in Retro UNIX 8086 v1 !
40649                                  <1>       ;     ((!'sysbreak' is not needed in Retro UNIX 8086 v1!))
40650                                  <1>       ;  NOTE:
40651                                  <1>       ;     'sysbreak' clears extended part (beyond of previous
40652                                  <1>       ;     'u.break' address) of user's memory for original unix's
40653                                  <1>       ;     'bss' compatibility with Retro UNIX 8086 v1 (19/11/2013)
40654                                  <1>
40655                                  <1>             ; mov u.break,r1 / move users break point to r1
40656                                  <1>             ; cmp r1,$core / is it the same or lower than core?
40657                                  <1>             ; blos 1f / yes, 1f
40658                                  <1>       ; 23/06/2015
40659 0000DDEF 8B2D[90030600]        <1>       mov   ebp, [u.break] ; virtual address (offset)
40660                                  <1>       ;and  ebp, ebp
40661                                  <1>       ;jz   short sysbreak_3
40662                                  <1>       ; Retro UNIX 386 v1 NOTE: u.break points to virtual address !!!
40663                                  <1>       ; (Even break point address is not needed for Retro UNIX 386 v1)
40664 0000DDF5 8B15[5C030600]        <1>       mov   edx, [u.sp] ; kernel stack at the beginning of sys call
40665 0000DDFB 83C20C                <1>       add   edx, 12 ; EIP -4-> CS -4-> EFLAGS -4-> ESP (user)
40666                                  <1>       ; 07/10/2015
40667 0000DDFE 891D[90030600]        <1>       mov   [u.break], ebx ; virtual address !!!
40668                                  <1>       ;
40669 0000DE04 3B1A                  <1>       cmp   ebx, [edx] ; compare new break point with
40670                                  <1>                        ; with top of user's stack (virtual!)
40671 0000DE06 7327                  <1>       jnb   short sysbreak_3
40672                                  <1>             ; cmp r1,sp / is it the same or higher
40673                                  <1>                     ; / than the stack?
40674                                  <1>             ; bhis 1f / yes, 1f
40675 0000DE08 89DE                  <1>       mov   esi, ebx
40676 0000DE0A 29EE                  <1>       sub   esi, ebp ; new break point - old break point
40677 0000DE0C 7621                  <1>       jna   short sysbreak_3
40678                                  <1>       ;push ebx
40679                                  <1> sysbreak_1:
40680 0000DE0E 89EB                  <1>       mov   ebx, ebp
40681 0000DE10 E87974FFFF            <1>       call  get_physical_addr ; get physical address
40682 0000DE15 0F82ADFEFFFF          <1>       jc    tr_addr_nm_err
40683                                  <1>       ; 18/10/2015
40684 0000DE1B 89C7                  <1>       mov   edi, eax
40685 0000DE1D 29C0                  <1>       sub   eax, eax ; 0
40686                                  <1>             ; ECX = remain byte count in page (1-4096)
40687 0000DE1F 39CE                  <1>       cmp   esi, ecx
40688 0000DE21 7302                  <1>       jnb   short sysbreak_2
40689 0000DE23 89F1                  <1>       mov   ecx, esi
```

```
40690                                  <1> sysbreak_2:
40691 0000DE25 29CE                    <1>         sub    esi, ecx
40692 0000DE27 01CD                    <1>         add    ebp, ecx
40693 0000DE29 F3AA                    <1>         rep    stosb
40694 0000DE2B 09F6                    <1>         or     esi, esi
40695 0000DE2D 75DF                    <1>         jnz    short sysbreak_1
40696                                  <1>         ;
40697                                  <1>                ; bit $1,r1 / is it an odd address
40698                                  <1>                ; beq 2f / no, its even
40699                                  <1>                ; clrb (r1)+ / yes, make it even
40700                                  <1>         ; 2: / clear area between the break point and the stack
40701                                  <1>                ; cmp r1,sp / is it higher or same than the stack
40702                                  <1>                ; bhis 1f / yes, quit
40703                                  <1>                ; clr (r1)+ / clear word
40704                                  <1>                ; br 2b / go back
40705                                  <1>         ;pop   ebx
40706                                  <1> sysbreak_3: ; 1:
40707                                  <1>         ;mov   [u.break], ebx ; virtual address !!!
40708                                  <1>                ; jsr r0,arg; u.break / put the "address"
40709                                  <1>                      ; / in u.break (set new break point)
40710                                  <1>                ; br sysret4 / br sysret
40711 0000DE2F E97BE6FFFF              <1>         jmp    sysret
40712                                  <1>
40713                                  <1> sysseek: ; / moves read write pointer in an fsp entry
40714                                  <1>         ; 06/11/2016 - TRDOS 386 (TRDOS v2.0)
40715                                  <1>         ; 22/06/2015 (Retro UNIX 386 v1 - Beginning)
40716                                  <1>         ; 07/07/2013 - 05/08/2013 (Retro UNIX 8086 v1)
40717                                  <1>         ;
40718                                  <1>         ; 'sysseek' changes the r/w pointer of (3rd word of in an
40719                                  <1>         ; fsp entry) of an open file whose file descriptor is in u.r0.
40720                                  <1>         ; The file descriptor refers to a file open for reading or
40721                                  <1>         ; writing. The read (or write) pointer is set as follows:
40722                                  <1>         ;     * if 'ptrname' is 0, the pointer is set to offset.
40723                                  <1>         ;     * if 'ptrname' is 1, the pointer is set to its
40724                                  <1>         ;       current location plus offset.
40725                                  <1>         ;     * if 'ptrname' is 2, the pointer is set to the
40726                                  <1>         ;       size of file plus offset.
40727                                  <1>         ; The error bit (e-bit) is set for an undefined descriptor.
40728                                  <1>         ;
40729                                  <1>         ; Calling sequence:
40730                                  <1>         ;     sysseek; offset; ptrname
40731                                  <1>         ; Arguments:
40732                                  <1>         ;     offset - number of bytes desired to move
40733                                  <1>         ;              the r/w pointer
40734                                  <1>         ;     ptrname - a switch indicated above
40735                                  <1>         ;
40736                                  <1>         ; Inputs: r0 - file descriptor
40737                                  <1>         ; Outputs: -
40738                                  <1>         ; ..........................................................
40739                                  <1>         ;
40740                                  <1>         ; Retro UNIX 8086 v1 modification:
40741                                  <1>         ;       'sysseek' system call has three arguments; so,
40742                                  <1>         ;     * 1st argument, file descriptor is in BX (BL) register
40743                                  <1>         ;     * 2nd argument, offset is in CX register
40744                                  <1>         ;     * 3rd argument, ptrname/switch is in DX (DL) register
40745                                  <1>
40746 0000DE34 E821000000              <1>         call   seektell
40747                                  <1>         ; EAX = Current R/W pointer of the file
40748                                  <1>         ; EBX = [u.fofp]
40749                                  <1>         ; [u.base] = offset (ECX input)
40750                                  <1>
40751 0000DE39 0305[84030600]          <1>         add    eax, [u.base]
40752 0000DE3F 8903                    <1>         mov    [ebx], eax
40753 0000DE41 E969E6FFFF              <1>         jmp    sysret
40754                                  <1>
40755                                  <1> systell: ; / get the r/w pointer
40756                                  <1>         ; 06/11/2016 - TRDOS 386 (TRDOS v2.0) - temporary !-
40757                                  <1>         ; 22/06/2015 (Retro UNIX 386 v1 - Beginning)
40758                                  <1>         ; 07/07/2013 - 05/08/2013 (Retro UNIX 8086 v1)
40759                                  <1>         ;
40760                                  <1>         ; Retro UNIX 8086 v1 modification:
40761                                  <1>         ; ! 'systell' does not work in original UNIX v1,
40762                                  <1>         ;          it returns with error !
40763                                  <1>         ; Inputs: r0 - file descriptor
40764                                  <1>         ; Outputs: r0 - file r/w pointer
40765                                  <1>
40766                                  <1>         ;xor   ecx, ecx ; 0
40767 0000DE46 BA01000000              <1>         mov    edx, 1 ; 05/08/2013
40768                                  <1>         ;call  seektell
40769 0000DE4B E810000000              <1>         call   seektell0 ; 05/08/2013
40770                                  <1>         ;; 06/11/2016
40771                                  <1>         ;; mov eax, [ebx]
40772 0000DE50 A3[64030600]            <1>         mov    [u.r0], eax
40773 0000DE55 E955E6FFFF              <1>         jmp    sysret
40774                                  <1>
40775                                  <1> ; Original unix v1 'systell' system call:
40776                                  <1>                ; jsr r0,seektell
40777                                  <1>                ; br error4
40778                                  <1>
40779                                  <1> seektell:
40780                                  <1>         ; 06/11/2016 - TRDOS 386 (TRDOS v2.0)
40781                                  <1>         ; 03/01/2016
40782                                  <1>         ; 22/06/2015 (Retro UNIX 386 v1 - Beginning)
40783                                  <1>         ; 07/07/2013 - 05/08/2013 (Retro UNIX 8086 v1)
40784                                  <1>         ;
40785                                  <1>         ; 'seektell' puts the arguments from sysseek and systell
40786                                  <1>         ; call in u.base and u.count. It then gets the i-number of
40787                                  <1>         ; the file from the file descriptor in u.r0 and by calling
40788                                  <1>         ; getf. The i-node is brought into core and then u.count
40789                                  <1>         ; is checked to see it is a 0, 1, or 2.
40790                                  <1>         ; If it is 0 - u.count stays the same
40791                                  <1>         ;          1 - u.count = offset (u.fofp)
```

```
40792                                  <1>       ;        2 - u.count = i.size (size of file)
40793                                  <1>       ;
40794                                  <1>       ; !! Retro UNIX 8086 v1 modification:
40795                                  <1>       ;     Argument 1, file descriptor is in BX;
40796                                  <1>       ;     Argument 2, offset is in CX;
40797                                  <1>       ;     Argument 3, ptrname/switch is in DX register.
40798                                  <1>       ;
40799                                  <1>       ; ((Return -> eax = base for offset (position= base+offset))
40800                                  <1>       ;
40801 0000DE5A 890D[84030600]          <1>       mov    [u.base], ecx ; offset
40802                                  <1> seektell0:
40803 0000DE60 8915[88030600]          <1>       mov    [u.count], edx
40804                                  <1>       ; EBX = file descriptor (file number)
40805 0000DE66 E898FCFFFF              <1>       call   getf1
40806                                  <1>       ; EAX = First cluster of the file
40807                                  <1>       ; EBX = File number (Open file number)
40808                                  <1>       ; [u.fofp] = Pointer to File pointer
40809                                  <1>       ; [i.size] = File size
40810                                  <1>
40811 0000DE6B 09C0                    <1>       or     eax, eax
40812 0000DE6D 7514                    <1>       jnz    short seektell1
40813                                  <1>
40814 0000DE6F B80A000000              <1>       mov    eax, ERR_FILE_NOT_OPEN
40815 0000DE74 A3[64030600]            <1>       mov    [u.r0], eax
40816 0000DE79 A3[C8030600]            <1>       mov    dword [u.error], eax ; 'file not open !'
40817 0000DE7E E90CE6FFFF              <1>       jmp    error
40818                                  <1>
40819                                  <1> seektell1:
40820 0000DE83 8B1D[74030600]          <1>         mov     ebx, [u.fofp]
40821 0000DE89 803D[88030600]01        <1>       cmp    byte [u.count], 1
40822 0000DE90 7705                    <1>       ja     short seektell2
40823 0000DE92 7409                    <1>       je     short seektell3
40824 0000DE94 31C0                    <1>       xor    eax, eax
40825 0000DE96 C3                      <1>       retn
40826                                  <1>
40827                                  <1> seektell2:
40828 0000DE97 A1[55040600]            <1>         mov        eax, [i.size]
40829 0000DE9C C3                      <1>       retn
40830                                  <1>
40831                                  <1> seektell3:
40832 0000DE9D 8B03                    <1>       mov    eax, [ebx]
40833 0000DE9F C3                      <1>       retn
40834                                  <1>
40835                                  <1> sysintr: ; / set interrupt handling
40836                                  <1>       ; 22/06/2015 (Retro UNIX 386 v1 - Beginning)
40837                                  <1>       ; 07/07/2013 (Retro UNIX 8086 v1)
40838                                  <1>       ;
40839                                  <1>       ; 'sysintr' sets the interrupt handling value. It puts
40840                                  <1>       ; argument of its call in u.intr then branches into 'sysquit'
40841                                  <1>       ; routine. u.tty is checked if to see if a control tty exists.
40842                                  <1>       ; If one does the interrupt character in the tty buffer is
40843                                  <1>       ; cleared and 'sysret'is called. If one does not exits
40844                                  <1>       ; 'sysret' is just called.
40845                                  <1>       ;
40846                                  <1>       ; Calling sequence:
40847                                  <1>       ;     sysintr; arg
40848                                  <1>       ; Argument:
40849                                  <1>       ;     arg - if 0, interrupts (ASCII DELETE) are ignored.
40850                                  <1>       ;         - if 1, intterupts cause their normal result
40851                                  <1>       ;           i.e force an exit.
40852                                  <1>       ;         - if arg is a location within the program,
40853                                  <1>       ;           control is passed to that location when
40854                                  <1>       ;           an interrupt occurs.
40855                                  <1>       ; Inputs: -
40856                                  <1>       ; Outputs: -
40857                                  <1>       ; ..........................................................
40858                                  <1>       ;
40859                                  <1>       ; Retro UNIX 8086 v1 modification:
40860                                  <1>       ;      'sysintr' system call sets u.intr to value of BX
40861                                  <1>       ;     then branches into sysquit.
40862                                  <1>       ;
40863 0000DEA0 66891D[AA030600]        <1>       mov    [u.intr], bx
40864                                  <1>             ; jsr r0,arg; u.intr / put the argument in u.intr
40865                                  <1>             ; br 1f / go into quit routine
40866 0000DEA7 E903E6FFFF              <1>       jmp    sysret
40867                                  <1>
40868                                  <1> sysquit:
40869                                  <1>       ; 22/06/2015 (Retro UNIX 386 v1 - Beginning)
40870                                  <1>       ; 07/07/2013 (Retro UNIX 8086 v1)
40871                                  <1>       ;
40872                                  <1>       ; 'sysquit' turns off the quit signal. it puts the argument of
40873                                  <1>       ; the call in u.quit. u.tty is checked if to see if a control
40874                                  <1>       ; tty exists. If one does the interrupt character in the tty
40875                                  <1>       ; buffer is cleared and 'sysret'is called. If one does not exits
40876                                  <1>       ; 'sysret' is just called.
40877                                  <1>       ;
40878                                  <1>       ; Calling sequence:
40879                                  <1>       ;     sysquit; arg
40880                                  <1>       ; Argument:
40881                                  <1>       ;     arg - if 0, this call diables quit signals from the
40882                                  <1>       ;           typewriter (ASCII FS)
40883                                  <1>       ;         - if 1, quits are re-enabled and cause execution to
40884                                  <1>       ;           cease and a core image to be produced.
40885                                  <1>       ;            i.e force an exit.
40886                                  <1>       ;         - if arg is an addres in the program,
40887                                  <1>       ;           a quit causes control to sent to that
40888                                  <1>       ;           location.
40889                                  <1>       ; Inputs: -
40890                                  <1>       ; Outputs: -
40891                                  <1>       ; ..........................................................
40892                                  <1>       ;
40893                                  <1>       ; Retro UNIX 8086 v1 modification:
```

```
40894                                  <1>       ;       'sysquit' system call sets u.quit to value of BX
40895                                  <1>       ;     then branches into 'sysret'.
40896                                  <1>       ;
40897 0000DEAC 66891D[AC030600]        <1>       mov   [u.quit], bx
40898 0000DEB3 E9F7E5FFFF              <1>       jmp   sysret
40899                                  <1>             ; jsr r0,arg; u.quit / put argument in u.quit
40900                                  <1>       ;1:
40901                                  <1>             ; mov u.ttyp,r1 / move pointer to control tty buffer
40902                                  <1>                         ; / to r1
40903                                  <1>             ; beq sysret4 / return to user
40904                                  <1>             ; clrb 6(r1) / clear the interrupt character
40905                                  <1>                      ; / in the tty buffer
40906                                  <1>             ; br sysret4 / return to user
40907                                  <1>
40908                                  <1> syssetuid: ; / set process id
40909                                  <1>       ; 22/06/2015 (Retro UNIX 386 v1 - Beginning)
40910                                  <1>       ; 07/07/2013 - 02/08/2013 (Retro UNIX 8086 v1)
40911                                  <1>       ;
40912                                  <1>       ; 'syssetuid' sets the user id (u.uid) of the current process
40913                                  <1>       ; to the process id in (u.r0). Both the effective user and
40914                                  <1>       ; u.uid and the real user u.ruid are set to this.
40915                                  <1>       ; Only the super user can make this call.
40916                                  <1>       ;
40917                                  <1>       ; Calling sequence:
40918                                  <1>       ;     syssetuid
40919                                  <1>       ; Arguments: -
40920                                  <1>       ;
40921                                  <1>       ; Inputs: (u.r0) - contains the process id.
40922                                  <1>       ; Outputs: -
40923                                  <1>       ; ..........................................................
40924                                  <1>       ;
40925                                  <1>       ; Retro UNIX 8086 v1 modification:
40926                                  <1>       ;       BL contains the (new) user ID of the current process
40927                                  <1>
40928                                  <1>             ; movb *u.r0,r1 / move process id (number) to r1
40929 0000DEB8 3A1D[B1030600]          <1>       cmp   bl, [u.ruid]
40930                                  <1>             ; cmpb r1,u.ruid / is it equal to the real user
40931                                  <1>                           ; / id number
40932 0000DEBE 741E                    <1>       je    short setuid1
40933                                  <1>             ; beq 1f / yes
40934 0000DEC0 803D[B0030600]00        <1>       cmp   byte [u.uid], 0 ; 02/08/2013
40935                                  <1>             ; tstb u.uid / no, is current user the super user?
40936                                  <1>       ;ja   error
40937                                  <1>             ; bne error4 / no, error
40938 0000DEC7 760F                    <1>       jna   short setuid0
40939 0000DEC9 C705[C8030600]0B00-     <1>       mov   dword [u.error], ERR_NOT_SUPERUSER  ; 11
40940 0000DED1 0000                    <1>
40941                                  <1>                             ;  'permission denied !' error
40942 0000DED3 E9B7E5FFFF              <1>       jmp   error
40943                                  <1> setuid0:
40944 0000DED8 881D[B1030600]          <1>       mov   [u.ruid], bl
40945                                  <1> setuid1: ; 1:
40946 0000DEDE 881D[B0030600]          <1>       mov   [u.uid], bl ; 02/08/2013
40947                                  <1>             ; movb r1,u.uid / put process id in u.uid
40948                                  <1>             ; movb r1,u.ruid / put process id in u.ruid
40949 0000DEE4 E9C6E5FFFF              <1>       jmp   sysret
40950                                  <1>             ; br sysret4 / system return
40951                                  <1>
40952                                  <1> sysgetuid: ; < get user id >
40953                                  <1>       ; 22/06/2015 (Retro UNIX 386 v1 - Beginning)
40954                                  <1>       ; 07/07/2013 (Retro UNIX 8086 v1)
40955                                  <1>       ;
40956                                  <1>       ; 'sysgetuid' returns the real user ID of the current process.
40957                                  <1>       ; The real user ID identifies the person who is logged in,
40958                                  <1>       ; in contradistinction to the effective user ID, which
40959                                  <1>       ; determines his access permission at each moment. It is thus
40960                                  <1>       ; useful to programs which operate using the 'set user ID'
40961                                  <1>       ; mode, to find out who invoked them.
40962                                  <1>       ;
40963                                  <1>       ; Calling sequence:
40964                                  <1>       ;     syssetuid
40965                                  <1>       ; Arguments: -
40966                                  <1>       ;
40967                                  <1>       ; Inputs: -
40968                                  <1>       ; Outputs: (u.r0) - contains the real user's id.
40969                                  <1>       ; ..........................................................
40970                                  <1>       ;
40971                                  <1>       ; Retro UNIX 8086 v1 modification:
40972                                  <1>       ;       AL contains the real user ID at return.
40973                                  <1>       ;
40974 0000DEE9 0FB605[B1030600]        <1>       movzx  eax, byte [u.ruid]
40975 0000DEF0 A3[64030600]            <1>       mov   [u.r0], eax
40976                                  <1>             ; movb u.ruid,*u.r0 / move the real user id to (u.r0)
40977 0000DEF5 E9B5E5FFFF              <1>       jmp   sysret
40978                                  <1>             ; br sysret4 / systerm return, sysret
40979                                  <1>
40980                                  <1> anyi:
40981                                  <1>       ; 06/10/2016 (TRDOS 386 = TRDOS v2.0)
40982                                  <1>       ; Major Modification!
40983                                  <1>       ; TRDOS 386 does not permit to delete a file while it is open
40984                                  <1>       ; The role of 'anyi' procedure has beeen changed to ensure that.
40985                                  <1>       ;
40986                                  <1>       ; 22/06/2015 (Retro UNIX 386 v1 - Beginning)
40987                                  <1>       ; 25/04/2013 (Retro UNIX 8086 v1)
40988                                  <1>       ;
40989                                  <1>       ; 'anyi' is called if a file deleted while open.
40990                                  <1>       ; "anyi" checks to see if someone else has opened this file.
40991                                  <1>       ;
40992                                  <1>       ; INPUTS ->
40993                                  <1>       ;    r1 - contains an i-number
40994                                  <1>       ;    fsp - start of table containing open files
40995                                  <1>       ;
```

```
                                <1>       ; OUTPUTS ->
                                <1>       ;    "deleted" flag set in fsp entry of another occurrence of
                                <1>       ;         this file and r2 points 1st word of this fsp entry.
                                <1>       ;    if file not found - bit in i-node map is cleared
                                <1>       ;                   (i-node is freed)
                                <1>       ;              all blocks related to i-node are freed
                                <1>       ;             all flags in i-node are cleared
                                <1>       ; ((AX = R1)) input
                                <1>       ;
                                <1>       ;    (Retro UNIX Prototype : 02/12/2012, UNIXCOPY.ASM)
                                <1>         ;    ((Modified registers: eDX, eCX, eBX, eSI, eDI, eBP))
                                <1>       ;
                                <1>       ; / r1 contains an i-number
                                <1>
                                <1>       ; TRDOS 386 (06/10/2016)
                                <1>       ;
                                <1>       ; INPUT:
                                <1>       ;     EAX = First Cluster
                                <1>       ;      DL = Logical DOS Drive Number
                                <1>       ;
                                <1>       ; OUTPUT:
                                <1>       ;     CF = 1 -> EBX = File Handle/Number/Index
                                <1>       ;     CF = 0 -> EBX = 0
                                <1>       ;
                                <1>       ; Modified Registers: EBX
                                <1>
0000DEFA 31DB                   <1>       xor    ebx, ebx
                                <1> anyi_0:
0000DEFC 80BB[F25E0100]00       <1>       cmp    byte [ebx+OF_MODE], 0 ; 0 = empty entry
0000DF03 770A                   <1>       ja     short anyi_2 ; 1 (r), 2 (w) or 3 (r&w)
                                <1> anyi_1:
0000DF05 FEC3                   <1>       inc    bl
0000DF07 80FB0A                 <1>       cmp    bl, OPENFILES ; max. count of open files
0000DF0A 72F0                   <1>       jb     short anyi_0
0000DF0C 31C0                   <1>       xor    eax, eax
0000DF0E C3                     <1>       retn
                                <1> anyi_2:
0000DF0F 3A93[E85E0100]         <1>       cmp    dl, [ebx+OF_DRIVE]
0000DF15 75EE                   <1>       jne    short anyi_1
0000DF17 66C1E302               <1>       shl    bx, 2 ; *4 (dword offset)
0000DF1B 3B83[C05E0100]         <1>       cmp    eax, [ebx+OF_FCLUSTER]
0000DF21 7406                   <1>       je     short anyi_3
0000DF23 66C1EB02               <1>       shr    bx, 2 ; /4 (byte offset)
0000DF27 EBDC                   <1>       jmp    short anyi_1
                                <1> anyi_3:
0000DF29 66C1EB02               <1>       shr    bx, 2 ; /4 (bytes offset) (index)
0000DF2D F9                     <1>       stc
0000DF2E C3                     <1>       retn
                                <1>
                                <1> ; Retro UNIX 386 v1 Kernel (v0.2) - u7.s
                                <1> ; Last Modification: 14/11/2015
                                <1>
                                <1> sysmount: ; / mount file system
                                <1>       ; 24/10/2016 - TRDOS 386 (TRDOS v2.0)
                                <1>       ; temporary !
0000DF2F B80F000000             <1>       mov    eax, ERR_DRV_NOT_RDY ; drive not ready !
0000DF34 A3[C8030600]           <1>         mov     [u.error], eax
0000DF39 A3[64030600]           <1>         mov     [u.r0], eax
0000DF3E E94CE5FFFF             <1>       jmp    error
                                <1>
                                <1> sysumount: ; / special dismount file system
                                <1>       ; 24/10/2016 - TRDOS 386 (TRDOS v2.0)
                                <1>       ; temporary !
0000DF43 B80F000000             <1>       mov    eax, ERR_DRV_NOT_RDY ; drive not ready !
0000DF48 A3[C8030600]           <1>         mov     [u.error], eax
0000DF4D A3[64030600]           <1>         mov     [u.r0], eax
0000DF52 E938E5FFFF             <1>       jmp    error
                                <1>
                                <1> ; Retro UNIX 386 v1 Kernel (v0.2) - SYS9.INC
                                <1> ; Last Modification: 09/12/2015
                                <1>
                                <1> syssleep:
                                <1>       ; 29/06/2015 - (Retro UNIX 386 v1)
                                <1>       ; 11/06/2014 - (Retro UNIX 8086 v1)
                                <1>       ;
                                <1>       ; Retro UNIX 8086 v1 feature only
                                <1>       ; (INPUT -> none)
                                <1>       ;
0000DF57 0FB61D[B3030600]       <1>       movzx  ebx, byte [u.uno] ; process number
0000DF5E 8AA3[7F000600]         <1>       mov    ah, [ebx+p.ttyc-1] ; current/console tty
0000DF64 E824120000             <1>       call   sleep
0000DF69 E941E5FFFF             <1>       jmp    sysret
                                <1>
                                <1> _vp_clr:
                                <1>       ; Reset/Clear Video Page
                                <1>       ;
                                <1>       ; 30/06/2015 - (Retro UNIX 386 v1)
                                <1>       ; 21/05/2013 - 30/10/2013(Retro UNIX 8086 v1) (U0.ASM)
                                <1>       ;
                                <1>       ; Retro UNIX 8086 v1 feature only !
                                <1>       ;
                                <1>       ; INPUTS ->
                                <1>       ;   BH = video page number
                                <1>       ;
                                <1>       ; OUTPUT ->
                                <1>       ;   none
                                <1>       ; ((Modified registers: eAX, BH, eCX, eDX, eSI, eDI))
                                <1>       ;
                                <1>       ; 04/12/2013
0000DF6E 28C0                   <1>       sub    al, al
                                <1>       ; al = 0 (clear video page)
                                <1>       ; bh = video page ; 13/05/2016
```

```
41098 0000DF70 B407                <1>       mov   ah, 07h
41099                              <1>       ; ah = 7 (attribute/color)
41100 0000DF72 6631C9              <1>       xor   cx, cx ; 0, left upper column (cl) & row (cl)
41101 0000DF75 66BA4F18            <1>       mov   dx, 184Fh ; right lower column & row (dl=24, dh=79)
41102 0000DF79 E88C3AFFFF          <1>       call  _scroll_up
41103                              <1>       ; bh = video page
41104 0000DF7E 6631D2              <1>       xor   dx, dx ; 0 (cursor position)
41105 0000DF81 E9C23DFFFF          <1>       jmp   _set_cpos
41106                              <1>
41107                              <1> sysmsg:
41108                              <1>       ; 13/05/2016
41109                              <1>       ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
41110                              <1>       ; 01/07/2015 - 11/11/2015 (Retro UNIX 386 v1)
41111                              <1>       ; Print user-application message on user's console tty
41112                              <1>       ;
41113                              <1>       ; Input -> EBX = Message address
41114                              <1>       ;         ECX = Message length (max. 255)
41115                              <1>       ;         DL = Color (IBM PC Rombios color attributes)
41116                              <1>       ;
41117 0000DF86 81F9FF000000        <1>       cmp   ecx, MAX_MSG_LEN ; 255
41118 0000DF8C 0F871DE5FFFF        <1>       ja    sysret ; nothing to do with big message size
41119 0000DF92 08C9                <1>       or    cl, cl
41120 0000DF94 0F8415E5FFFF        <1>       jz    sysret
41121 0000DF9A 20D2                <1>       and   dl, dl
41122 0000DF9C 7502                <1>       jnz   short sysmsg0
41123 0000DF9E B207                <1>       mov   dl, 07h ; default color
41124                              <1>             ; (black background, light gray character)
41125                              <1> sysmsg0:
41126 0000DFA0 891D[84030600]      <1>       mov   [u.base], ebx
41127 0000DFA6 8815[F74D0100]      <1>       mov   [ccolor], dl ; color attributes
41128 0000DFAC 89E5                <1>       mov   ebp, esp
41129 0000DFAE 31DB                <1>       xor   ebx, ebx ; 0
41130 0000DFB0 891D[8C030600]      <1>       mov   [u.nread], ebx ; 0
41131                              <1>       ;
41132 0000DFB6 381D[C6030600]      <1>       cmp   [u.kcall], bl ; 0
41133 0000DFBC 7769                <1>       ja    short sysmsgk ; Temporary (01/07/2015)
41134                              <1>       ;
41135 0000DFBE 890D[88030600]      <1>       mov   [u.count], ecx
41136 0000DFC4 41                  <1>       inc   ecx ; + 00h ; ASCIIZ
41137 0000DFC5 29CC                <1>       sub   esp, ecx
41138 0000DFC7 89E7                <1>       mov   edi, esp
41139 0000DFC9 89E6                <1>       mov   esi, esp
41140 0000DFCB 66891D[C4030600]    <1>       mov   [u.pcount], bx ; reset page (phy. addr.) counter
41141                              <1>       ; 11/11/2015
41142 0000DFD2 8A25[94030600]      <1>       mov   ah, [u.ttyp] ; recent open tty
41143                              <1>       ; 0 = none
41144 0000DFD8 FECC                <1>       dec   ah
41145 0000DFDA 790C                <1>       jns   short sysmsg1
41146 0000DFDC 8A1D[B3030600]      <1>       mov   bl, [u.uno] ; process number
41147 0000DFE2 8AA3[7F000600]      <1>       mov   ah, [ebx+p.ttyc-1] ; user's (process's) console tty
41148                              <1> sysmsg1:
41149 0000DFE8 8825[96030600]      <1>       mov   [u.ttyn], ah
41150                              <1> sysmsg2:
41151 0000DFEE E82F080000          <1>       call  cpass
41152 0000DFF3 7416                <1>       jz    short sysmsg5
41153 0000DFF5 AA                  <1>       stosb
41154 0000DFF6 20C0                <1>       and   al, al
41155 0000DFF8 75F4                <1>       jnz   short sysmsg2
41156                              <1> sysmsg3:
41157 0000DFFA 80FC07              <1>       cmp   ah, 7 ; tty number
41158 0000DFFD 7711                <1>       ja    short sysmsg6 ; serial port
41159 0000DFFF E83E000000          <1>       call  print_cmsg
41160                              <1> sysmsg4:
41161 0000E004 89EC                <1>       mov   esp, ebp
41162 0000E006 E9A4E4FFFF          <1>       jmp   sysret
41163                              <1> sysmsg5:
41164 0000E00B C60700              <1>       mov   byte [edi], 0
41165 0000E00E EBEA                <1>       jmp   short sysmsg3
41166                              <1> sysmsg6:
41167 0000E010 8A06                <1>       mov   al, [esi]
41168 0000E012 E873110000          <1>       call  sndc
41169 0000E017 72EB                <1>       jc    short sysmsg4
41170 0000E019 803E00              <1>       cmp   byte [esi], 0  ; 0 is stop character
41171 0000E01C 76E6                <1>       jna   short sysmsg4
41172 0000E01E 46                  <1>       inc   esi
41173 0000E01F 8A25[96030600]      <1>       mov   ah, [u.ttyn]
41174 0000E025 EBE9                <1>       jmp   short sysmsg6
41175                              <1>
41176                              <1> sysmsgk: ; Temporary (01/07/2015)
41177                              <1>       ; The message has been sent by Kernel (ASCIIZ string)
41178                              <1>       ; (ECX -character count- will not be considered)
41179 0000E027 8B35[84030600]      <1>       mov   esi, [u.base]
41180 0000E02D 8A25[F64D0100]      <1>       mov   ah, [ptty] ; present/current screen (video page)
41181 0000E033 8825[96030600]      <1>       mov   [u.ttyn], ah
41182 0000E039 C605[C6030600]00    <1>       mov   byte [u.kcall], 0
41183 0000E040 EBB8                <1>       jmp   short sysmsg3
41184                              <1>
41185                              <1> print_cmsg:
41186                              <1>       ; 13/05/2016 - TRDOS 386 (TRDOS v2.0)
41187                              <1>       ; 01/07/2015 (Retro UNIX 386 v1)
41188                              <1>       ;
41189                              <1>       ; print message (on user's console tty)
41190                              <1>       ;     with requested color
41191                              <1>       ;
41192                              <1>       ; INPUTS:
41193                              <1>       ;     esi = message address
41194                              <1>       ;     [u.ttyn] = tty number (0 to 7)
41195                              <1>       ;     [ccolor] = color attributes (IBM PC BIOS colors)
41196                              <1>
41197 0000E042 8A3D[96030600]      <1>       mov   bh, [u.ttyn]
41198                              <1>       ;mov  bh, ah
41199                              <1>
```

```
41200 0000E048 AC                  <1>       lodsb
41201                              <1> pcmsg1:
41202 0000E049 56                  <1>       push   esi
41203 0000E04A 8A1D[F74D0100]      <1>       mov    bl, [ccolor]
41204                              <1>       ;mov   bh, [u.ttyn]
41205 0000E050 E85D3CFFFF          <1>       call   _write_tty
41206 0000E055 5E                  <1>       pop    esi
41207 0000E056 AC                  <1>       lodsb
41208 0000E057 20C0                <1>       and    al, al  ; 0
41209 0000E059 75EE                <1>       jnz    short pcmsg1
41210 0000E05B C3                  <1>       retn
41211                              <1>
41212                              <1> sysgeterr:
41213                              <1>       ; 09/12/2015
41214                              <1>       ; 21/09/2015 - (Retro UNIX 386 v1 feature only!)
41215                              <1>       ; Get last error number or page fault count
41216                              <1>       ; (for debugging)
41217                              <1>       ;
41218                              <1>       ; Input -> EBX = return type
41219                              <1>       ;         0 = last error code (which is in 'u.error')
41220                              <1>       ;         FFFFFFFFh = page fault count for running process
41221                              <1>       ;         FFFFFFFEh = total page fault count
41222                              <1>       ;         1 .. FFFFFFFDh = undefined
41223                              <1>       ;
41224                              <1>       ; Output -> EAX = last error number or page fault count
41225                              <1>       ;         (depending on EBX input)
41226                              <1>       ;
41227 0000E05C 21DB                <1>       and    ebx, ebx
41228 0000E05E 750B                <1>       jnz    short glerr_2
41229                              <1> glerr_0:
41230 0000E060 A1[C8030600]        <1>       mov    eax, [u.error]
41231                              <1> glerr_1:
41232 0000E065 A3[64030600]        <1>       mov    [u.r0], eax
41233 0000E06A C3                  <1>       retn
41234                              <1> glerr_2:
41235 0000E06B 43                  <1>       inc    ebx ; FFFFFFFFh -> 0, FFFFFFFEh -> FFFFFFFFh
41236 0000E06C 74FD                <1>       jz     short glerr_2 ; page fault count for process
41237 0000E06E 43                  <1>       inc    ebx ; FFFFFFFFh -> 0
41238 0000E06F 75EF                <1>       jnz    short glerr_0
41239 0000E071 A1[80050600]        <1>       mov    eax, [PF_Count] ; total page fault count
41240 0000E076 EBED                <1>         jmp     short glerr_1
41241                              <1> glerr_3:
41242 0000E078 A1[CC030600]        <1>       mov    eax, [u.pfcount]
41243 0000E07D EBE6                <1>       jmp    short glerr_1
41244                              <1>
41245                              <1> load_and_run_file:
41246                              <1>       ; 22/01/2017
41247                              <1>       ; 07/01/2017
41248                              <1>       ; 04/01/2017
41249                              <1>       ; 24/10/2016
41250                              <1>       ; 24/04/2016, 02/05/2016, 03/05/2016, 06/05/2016
41251                              <1>       ; 23/04/2016 (TRDOS 386 = TRDOS v2.0)
41252                              <1>       ; 23/10/2015 (Retro UNIX 386 v1, 'sysexec')
41253                              <1>       ; 23/06/2015 (Retro UNIX 386 v1 - Beginning)
41254                              <1>       ; 03/06/2013 - 06/12/2013 (Retro UNIX 8086 v1)
41255                              <1>       ; EAX = First Cluster number
41256                              <1>       ; EDX = File Size
41257                              <1>       ; ESI = Argument list address
41258                              <1>       ; [argc] = argument count
41259                              <1>       ; [u.nread] = argument list length
41260                              <1>       ; [esp] = return address to the caller (*)
41261                              <1>       ;
41262 0000E07F 8935[4C040600]      <1>       mov    [argv], esi
41263 0000E085 8915[55040600]      <1>       mov    [i.size], edx
41264 0000E08B A3[51040600]        <1>       mov    [ii], eax
41265                              <1>
41266                              <1>       ;sti   ; 07/01/2017
41267                              <1>       ;mov   eax, [k_page_dir]
41268                              <1>       ;mov   [u.pgdir], eax
41269 0000E090 31C0                <1>       xor    eax, eax ; clc ; *** ; 04/01/2017
41270                              <1>       ;mov   [u.r0], eax ; 0 ; 07/01/2017
41271                              <1>
41272                              <1>       ; 06/05/2016
41273                              <1>       ; Set 'sysexit' return order to MainProg
41274                              <1>       ;
41275 0000E092 58                  <1>       pop    eax ; * 'loc_load_and_run_file_8:' address
41276                              <1>       ;; 22/01/2017
41277                              <1>       ;;cli ; 07/01/2017
41278 0000E093 8B25[644D0100]      <1>       mov    esp, [tss.esp0]
41279                              <1>       ;
41280                              <1>       ; 'loc_load_run_file_8' address has
41281                              <1>       ; 'jmp loc_file_rw_restore_retn' instruction
41282                              <1>       ; 'loc_file_rw_restore_retn:' will return to
41283                              <1>       ; [mainprog_return_addr]
41284                              <1>       ; just after 'call command_interpreter'
41285                              <1>       ;
41286 0000E099 68[3B630000]        <1>       push   _end_of_mainprog ; we must not return to here !
41287 0000E09E FF35[4C5B0100]      <1>       push   dword [mainprog_return_addr]
41288 0000E0A4 89E5                <1>       mov    ebp, esp ; **
41289                              <1>       ;
41290 0000E0A6 9C                  <1>       pushfd  ; EFLAGS      ; IRETD ; ***
41291 0000E0A7 6A08                <1>       push   KCODE ; cs    ; IRETD
41292 0000E0A9 50                  <1>       push   eax ; * (eip) ; IRETD
41293 0000E0AA 8925[5C030600]      <1>       mov    [u.sp], esp
41294                              <1>       ;mov   byte [u.quant], time_count
41295 0000E0B0 1E                  <1>       push   ds
41296 0000E0B1 06                  <1>       push   es
41297 0000E0B2 0FA0                <1>       push   fs
41298 0000E0B4 0FA8                <1>       push   gs
41299                              <1>       ;mov   eax, [u.r0]
41300 0000E0B6 29C0                <1>       sub    eax, eax
41301 0000E0B8 60                  <1>       pushad
```

```
41302 0000E0B9 68[AFC40000]        <1>       push   sysret
41303                              <1>       ;push  sysrel1 ; 07/01/2017
41304 0000E0BE 8925[60030600]      <1>       mov    [u.usp], esp
41305                              <1>       ;
41306 0000E0C4 E85E060000          <1>       call   wswap ; Save MainProg (process 1) 'u' structure
41307                              <1>                    ; and registers for return (from program)
41308 0000E0C9 89EC                <1>       mov    esp, ebp ; **
41309                              <1>       ;;22/01/2017
41310                              <1>       ;;sti ; 07/01/2017
41311 0000E0CB 50                  <1>       push   eax  ; * 'loc_load_and_run_file_8:' address
41312                              <1>       ;
41313                              <1>       ;;; 02/05/2016
41314                              <1>       ;;; Create a new process (parent: MainProg)
41315 0000E0CC 31F6                <1>       xor    esi, esi
41316                              <1> cnpm_1: ; search p.stat table for unused process number
41317 0000E0CE 46                  <1>       inc    esi
41318 0000E0CF 80BE[AF000600]00    <1>       cmp    byte [esi+p.stat-1], 0 ; SFREE
41319                              <1>                             ; is process active, unused, dead
41320 0000E0D6 760B                <1>       jna    short cnpm_2 ; it's unused so branch
41321 0000E0D8 6683FE10            <1>       cmp    si, nproc    ; all processes checked
41322 0000E0DC 72F0                <1>       jb     short cnpm_1    ; no, branch back
41323 0000E0DE E9CE82FFFF          <1>       jmp    panic
41324                              <1> cnpm_2:
41325 0000E0E3 A1[B8030600]        <1>       mov    eax, [u.pgdir] ; page directory of MainProg
41326 0000E0E8 A3[BC030600]        <1>       mov    [u.ppgdir], eax ; parent's page directory
41327 0000E0ED E8876AFFFF          <1>       call   allocate_page
41328 0000E0F2 0F82B982FFFF        <1>       jc     panic
41329                              <1>       ; EAX = UPAGE (user structure page) address
41330 0000E0F8 A3[B4030600]        <1>       mov    [u.upage], eax ; memory page for 'user' struct (child)
41331 0000E0FD 89F7                <1>       mov    edi, esi
41332 0000E0FF 66C1E702            <1>       shl    di, 2
41333 0000E103 8987[BC000600]      <1>       mov    [edi+p.upage-4], eax ; memory page for 'user' struct
41334 0000E109 E8E56AFFFF          <1>       call   clear_page ; 03/05/2016
41335                              <1>       ;movzx eax, byte [p.ttyc] ; console tty (for MainProg)
41336 0000E10E 6629C0              <1>       sub    ax, ax ; 0
41337 0000E111 668986[7F000600]    <1>       mov     [esi+p.ttyc-1], ax ; al - set child's console tty
41338                              <1>                               ; ah - reset child's wait channel
41339 0000E118 89F0                <1>       mov    eax, esi
41340 0000E11A A2[B3030600]        <1>       mov    [u.uno], al ; child process number
41341 0000E11F FE86[AF000600]      <1>         inc     byte [esi+p.stat-1] ; 1, SRUN
41342 0000E125 66D1E6              <1>       shl    si, 1 ; multiply si by 2 to get index into p.pid table
41343 0000E128 66FF05[4E030600]    <1>       inc    word [mpid] ; increment m.pid; get a new process name
41344 0000E12F 66A1[4E030600]      <1>       mov    ax, [mpid]
41345 0000E135 668986[1E000600]    <1>       mov    [esi+p.pid-2], ax ; put new process name
41346                              <1>                               ; in child process' name slot
41347                              <1>       ;mov   ax, [p.pid]  ; get process name of MainProg
41348 0000E13C 66B80100            <1>       mov    ax, 1
41349 0000E140 668986[3E000600]    <1>       mov    [esi+p.ppid-2], ax ; put parent process name
41350                              <1>                                 ; in parent process slot for child
41351 0000E147 6648                <1>       dec    ax ; 0
41352 0000E149 66A3[94030600]      <1>       mov    [u.ttyp], ax ; 0
41353                              <1>       ;;;
41354 0000E14F A1[51040600]        <1>       mov    eax,  [ii]
41355                              <1>       ; Retro UNIX 386 v1, 'sysexec' (u2.s)
41356 0000E154 E82E100000          <1>       call   iopen
41357                              <1>       ; 06/06/2016
41358 0000E159 C605[A9030600]01    <1>       mov    byte [u.pri], 1 ; normal priority
41359                              <1>       ;
41360 0000E160 EB0A                <1>       jmp    short sysexec_7 ; 02/05/2016
41361                              <1>
41362                              <1> sysexec_6:
41363                              <1>       ; 04/01/2017
41364                              <1>       ; 24/10/2016
41365                              <1>       ;;02/05/2016
41366                              <1>       ; 23/04/2016
41367                              <1>       ; 18/10/2015 ('sysexec_6')
41368                              <1>       ; 23/06/2015
41369 0000E162 A1[B8030600]        <1>       mov    eax, [u.pgdir] ; physical address of page directory
41370                              <1>       ;;cmp  eax, [k_page_dir] ; TRDOS MainProg ?
41371                              <1>       ;;je   short sysexec_7
41372 0000E167 E8466BFFFF          <1>       call   deallocate_page_dir
41373                              <1> sysexec_7:
41374 0000E16C E8766AFFFF          <1>       call   make_page_dir
41375 0000E171 0F823A82FFFF        <1>       jc     panic  ; allocation error
41376                              <1>                     ; after a deallocation would be nonsence !?
41377                              <1>       ; 24/07/2015
41378                              <1>       ; map kernel pages (1st 4MB) to PDE 0
41379                              <1>       ;     of the user's page directory
41380                              <1>       ;     (It is needed for interrupts!)
41381                              <1>       ; 18/10/2015
41382 0000E177 8B15[C84D0100]      <1>       mov    edx, [k_page_dir] ; Kernel's page directory
41383 0000E17D 8B02                <1>       mov    eax, [edx] ; physical address of
41384                              <1>                       ; kernel's first page table (1st 4 MB)
41385                              <1>                       ; (PDE 0 of kernel's page directory)
41386 0000E17F 8B15[B8030600]      <1>       mov    edx, [u.pgdir]
41387 0000E185 8902                <1>       mov    [edx], eax ; PDE 0 (1st 4MB)
41388                              <1>       ;
41389                              <1>       ; 20/07/2015
41390 0000E187 BB00004000          <1>       mov    ebx, CORE ; start address = 0 (virtual) + CORE
41391                              <1>       ; 18/10/2015
41392 0000E18C BE[3C040600]        <1>       mov    esi, pcore ; physical start address
41393                              <1> sysexec_8:
41394 0000E191 B907000000          <1>       mov    ecx, PDE_A_USER + PDE_A_WRITE + PDE_A_PRESENT
41395 0000E196 E86A6AFFFF          <1>       call   make_page_table
41396 0000E19B 0F821082FFFF        <1>       jc     panic
41397                              <1>       ;mov   ecx, PTE_A_USER + PTE_A_WRITE + PTE_A_PRESENT
41398 0000E1A1 E86D6AFFFF          <1>       call   make_page ; make new page, clear and set the pte
41399 0000E1A6 0F820582FFFF        <1>       jc     panic
41400                              <1>       ;
41401 0000E1AC 8906                <1>       mov    [esi], eax ; 24/06/2015
41402                              <1>       ; ebx = virtual address (24/07/2015)
41403 0000E1AE E80570FFFF          <1>       call   add_to_swap_queue
```

```
41404                                  <1>       ; 18/10/2015
41405 0000E1B3 81FE[40040600]          <1>       cmp    esi, ecore ; user's stack (last) page ?
41406 0000E1B9 740C                    <1>       je     short sysexec_9 ; yes
41407 0000E1BB BE[40040600]            <1>       mov    esi, ecore  ; physical address of the last page
41408                                  <1>       ; 20/07/2015
41409 0000E1C0 BB00F0FFFF              <1>       mov    ebx, (ECORE - PAGE_SIZE) + CORE
41410                                  <1>       ; ebx = virtual end address + segment base address - 4K
41411 0000E1C5 EBCA                    <1>         jmp     short sysexec_8
41412                                  <1> sysexec_9:
41413                                  <1>       ; 24/04/2016
41414                                  <1>       ; 18/10/2015
41415                                  <1>       ; 26/08/2015
41416                                  <1>       ; 25/06/2015
41417                                  <1>       ; move arguments from kernel stack to [ecore]
41418                                  <1>       ; (argument list/line will be copied from kernel stack
41419                                  <1>       ; frame to the last (stack) page of user's core memory)
41420                                  <1>       ; 18/10/2015
41421 0000E1C7 8B3D[40040600]          <1>       mov    edi, [ecore]
41422 0000E1CD 81C700100000            <1>       add    edi, PAGE_SIZE
41423 0000E1D3 0FB705[4A040600]        <1>       movzx  eax, word [argc]
41424 0000E1DA 09C0                    <1>       or     eax, eax
41425 0000E1DC 7509                    <1>       jnz    short sysexec_10
41426 0000E1DE 89FB                    <1>       mov    ebx, edi
41427 0000E1E0 83EB04                  <1>       sub    ebx, 4
41428 0000E1E3 8903                    <1>       mov    [ebx], eax ; 0
41429 0000E1E5 EB44                    <1>       jmp    short sysexec_13
41430                                  <1> sysexec_10:
41431 0000E1E7 8B0D[8C030600]          <1>       mov    ecx, [u.nread]
41432                                  <1>       ;mov   esi, TextBuffer
41433 0000E1ED 8B35[4C040600]          <1>       mov    esi, [argv] ; 24/04/2016 (TRDOS 386  = TRDOS v2.0)
41434 0000E1F3 29CF                    <1>       sub    edi, ecx ; page end address - argument list length
41435 0000E1F5 89C2                    <1>       mov    edx, eax
41436 0000E1F7 FEC2                    <1>       inc    dl ; argument count + 1 for argc value
41437 0000E1F9 C0E202                  <1>       shl    dl, 2  ; 4 * (argument count + 1)
41438 0000E1FC 89FB                    <1>       mov    ebx, edi
41439 0000E1FE 80E3FC                  <1>       and    bl, 0FCh ; 32 bit (dword) alignment
41440 0000E201 29D3                    <1>       sub    ebx, edx
41441 0000E203 89FA                    <1>       mov    edx, edi
41442 0000E205 F3A4                    <1>       rep    movsb
41443 0000E207 89D6                    <1>       mov    esi, edx
41444 0000E209 89DF                    <1>       mov    edi, ebx
41445 0000E20B BA00F0BFFF              <1>       mov    edx, ECORE - PAGE_SIZE ; virtual addr. of the last page
41446 0000E210 2B15[40040600]          <1>       sub    edx, [ecore] ; difference (virtual - physical)
41447 0000E216 AB                      <1>       stosd  ; eax = argument count
41448                                  <1> sysexec_11:
41449 0000E217 89F0                    <1>       mov    eax, esi
41450 0000E219 01D0                    <1>       add    eax, edx
41451 0000E21B AB                      <1>       stosd  ; eax = virtual address
41452 0000E21C FE0D[4A040600]          <1>       dec    byte [argc]
41453 0000E222 7407                    <1>       jz     short sysexec_13
41454                                  <1> sysexec_12:
41455 0000E224 AC                      <1>       lodsb
41456 0000E225 20C0                    <1>       and    al, al
41457 0000E227 75FB                    <1>       jnz    short sysexec_12
41458 0000E229 EBEC                    <1>       jmp    short sysexec_11
41459                                  <1> sysexec_13:
41460                                  <1>       ; 24/10/2016
41461                                  <1>       ; 24/04/2016 - TRDOS 386 (TRDOS v2.0)
41462                                  <1>       ; 23/06/2015 - 19/10/2015 (Retro UNIX 386 v1, 'sysexec_13')
41463                                  <1>       ;
41464                                  <1>       ; moving arguments to [ecore] is OK here..
41465                                  <1>       ;
41466                                  <1>       ; ebx = beginning addres of argument list pointers
41467                                  <1>             ;     in user's stack
41468 0000E22B 2B1D[40040600]          <1>       sub    ebx, [ecore]
41469 0000E231 81C300F0BFFF            <1>       add     ebx, (ECORE - PAGE_SIZE)
41470                                  <1>                   ; end of core - 4096 (last page)
41471                                  <1>                   ; (virtual address)
41472 0000E237 891D[4C040600]          <1>       mov    [argv], ebx
41473 0000E23D 891D[90030600]          <1>       mov    [u.break], ebx ; available user memory
41474                                  <1>       ;
41475 0000E243 29C0                    <1>       sub    eax, eax
41476 0000E245 C705[88030600]2000-     <1>       mov    dword [u.count], 32 ; Executable file header size
41477 0000E24D 0000                    <1>
41478 0000E24F C705[74030600]-         <1>       mov    dword [u.fofp], u.off
41479 0000E255 [80030600]              <1>
41480 0000E259 A3[80030600]            <1>       mov    [u.off], eax ; 0
41481 0000E25E A3[84030600]            <1>       mov    [u.base], eax ; 0, start of user's core (virtual)
41482                                  <1>       ; 24/10/2016
41483 0000E263 A0[8E4E0100]            <1>       mov    al, [Current_Drv]
41484 0000E268 A2[46030600]            <1>       mov    [cdev], al
41485                                  <1>       ;
41486 0000E26D A1[51040600]            <1>       mov    eax, [ii] ; Fist Cluster of the Program (PRG) file
41487                                  <1>       ; EAX = First cluster of the executable file
41488 0000E272 E832010000              <1>       call   readi
41489                                  <1>
41490 0000E277 8B0D[90030600]          <1>       mov    ecx, [u.break] ; top of user's stack (physical addr.)
41491 0000E27D 890D[88030600]          <1>       mov    [u.count], ecx ; save for overrun check
41492                                  <1>       ;
41493 0000E283 8B0D[8C030600]          <1>       mov    ecx, [u.nread]
41494 0000E289 890D[90030600]          <1>       mov    [u.break], ecx ; virtual address (offset from start)
41495 0000E28F 80F920                  <1>       cmp    cl, 32
41496 0000E292 7540                    <1>         jne     short sysexec_15
41497                                  <1>       ;:
41498                                  <1>       ; Retro UNIX 386 v1 (32 bit) executable file header format
41499 0000E294 8B35[3C040600]          <1>       mov    esi, [pcore] ; start address of user's core memory
41500                                  <1>                           ; (phys. start addr. of the exec. file)
41501 0000E29A AD                      <1>       lodsd
41502 0000E29B 663DEB1E                <1>       cmp    ax, 1EEBh ; EBH, 1Eh -> jump to +32
41503 0000E29F 7533                    <1>       jne    short sysexec_15
41504 0000E2A1 AD                      <1>       lodsd
41505 0000E2A2 89C1                    <1>       mov    ecx, eax ; text (code) section size
```

```
41506 0000E2A4 AD                  <1>       lodsd
41507 0000E2A5 01C1                <1>       add    ecx, eax ; + data section size (initialized data)
41508 0000E2A7 89CB                <1>       mov    ebx, ecx
41509 0000E2A9 AD                  <1>       lodsd
41510 0000E2AA 01C3                <1>       add    ebx, eax ; + bss section size (for overrun checking)
41511 0000E2AC 3B1D[88030600]      <1>       cmp    ebx, [u.count]
41512 0000E2B2 7711                <1>       ja     short sysexec_14  ; program overruns stack !
41513                              <1>       ;
41514                              <1>       ; add bss section size to [u.break]
41515 0000E2B4 0105[90030600]      <1>       add    [u.break], eax
41516                              <1>       ;
41517 0000E2BA 83E920              <1>       sub    ecx, 32  ; header size (already loaded)
41518                              <1>       ;cmp   ecx, [u.count]
41519                              <1>       ;jnb   short sysexec_16
41520 0000E2BD 890D[88030600]      <1>       mov    [u.count], ecx ; required read count
41521 0000E2C3 EB29                <1>       jmp    short sysexec_16
41522                              <1> sysexec_14:
41523                              <1>       ; insufficient (out of) memory
41524 0000E2C5 C705[C8030600]0400- <1>       mov    dword [u.error], ERR_MINOR_IM ; 1
41525 0000E2CD 0000                <1>
41526 0000E2CF E9BBE1FFFF          <1>       jmp    error
41527                              <1> sysexec_15:
41528 0000E2D4 8B15[55040600]      <1>         mov  edx, [i.size] ; file size
41529 0000E2DA 29CA                <1>       sub    edx, ecx ; file size - loaded bytes
41530 0000E2DC 7626                <1>       jna    short sysexec_17 ; no need to next read
41531 0000E2DE 01D1                <1>       add    ecx, edx ; [i.size]
41532 0000E2E0 3B0D[88030600]      <1>       cmp    ecx, [u.count] ; overrun check (!)
41533 0000E2E6 77DD                <1>       ja     short sysexec_14
41534 0000E2E8 8915[88030600]      <1>       mov    [u.count], edx
41535                              <1> sysexec_16:
41536 0000E2EE A1[51040600]        <1>       mov    eax, [ii] ; first cluster
41537 0000E2F3 E8B1000000          <1>       call   readi
41538 0000E2F8 8B0D[8C030600]      <1>       mov    ecx, [u.nread]
41539 0000E2FE 010D[90030600]      <1>       add    [u.break], ecx
41540                              <1> sysexec_17:
41541 0000E304 A1[51040600]        <1>       mov    eax, [ii] ; first cluster
41542 0000E309 E87A0E0000          <1>       call   iclose
41543 0000E30E 31C0                <1>       xor     eax, eax
41544 0000E310 FEC0                <1>       inc    al
41545 0000E312 66A3[AA030600]      <1>       mov    [u.intr], ax ; 1 (interrupt/time-out is enabled)
41546 0000E318 66A3[AC030600]      <1>       mov    [u.quit], ax ; 1 ('crtl+brk' signal is enabled)
41547 0000E31E 833D[BC030600]00    <1>         cmp  dword [u.ppgdir], 0  ; is the caller MainProg (kernel) ?
41548 0000E325 770C                <1>       ja     short sysexec_18 ; no, the caller is user process
41549                              <1>       ; If the caller is kernel (MainProg), 'sysexec' will come here
41550 0000E327 8B15[C84D0100]      <1>       mov    edx, [k_page_dir] ; kernel's page directory
41551 0000E32D 8915[BC030600]      <1>       mov    [u.ppgdir], edx ; next time 'sysexec' must not come here
41552                              <1> sysexec_18:
41553                              <1>       ; 02/05/2016
41554                              <1>       ; 24/04/2016 (TRDOS 386 = TRDOS v2.0)
41555                              <1>       ; 18/10/2015 (Retro UNIX 386 v1)
41556                              <1>       ; 05/08/2015
41557                              <1>       ; 29/07/2015
41558 0000E333 8B2D[4C040600]      <1>       mov    ebp, [argv] ; user's stack pointer must point to argument
41559                              <1>                          ; list pointers (argument count)
41560 0000E339 FA                  <1>       cli
41561 0000E33A 8B25[644D0100]      <1>         mov     esp, [tss.esp0]  ; ring 0 (kernel) stack pointer
41562                              <1>       ;mov        esp, [u.sp] ; Restore Kernel stack
41563                              <1>                          ; for this process
41564                              <1>       ;add   esp, 20 ; --> EIP, CS, EFLAGS, ESP, SS
41565                              <1>       ;xor   eax, eax ; 0
41566 0000E340 FEC8                <1>       dec    al ; eax = 0
41567 0000E342 66BA2300            <1>       mov    dx, UDATA
41568 0000E346 6652                <1>       push   dx  ; user's stack segment
41569 0000E348 55                  <1>       push   ebp ; user's stack pointer
41570                              <1>                  ; (points to number of arguments)
41571                              <1>
41572                              <1>       ; 04/01/2017
41573                              <1>       ; MainProg comes here while [sysflg]= 0FFh
41574                              <1>       ; (but sysexec comes here while [sysflg]= 0)
41575 0000E349 C605[5B030600]00    <1>       mov    byte [sysflg], 0 ; 04/01/2017
41576                              <1>                             ; (timer_int sysflg control)
41577 0000E350 FB                  <1>       sti
41578 0000E351 9C                  <1>       pushfd ; EFLAGS
41579                              <1>              ; Set IF for enabling interrupts in user mode
41580                              <1>       ;or    dword [esp], 200h
41581                              <1>       ;
41582                              <1>       ;mov   bx, UCODE
41583                              <1>       ;push  bx ; user's code segment
41584 0000E352 6A1B                <1>       push   UCODE
41585                              <1>       ;push  0
41586 0000E354 50                  <1>       push   eax ; EIP (=0) - start address -
41587 0000E355 8925[5C030600]      <1>       mov    [u.sp], esp ; 29/07/2015
41588                              <1>       ; 05/08/2015
41589                              <1>       ; Remedy of a General Protection Fault during 'iretd' is here !
41590                              <1>       ; ('push dx' would cause to general protection fault,
41591                              <1>       ; after 'pop ds' etc.)
41592                              <1>       ;
41593                              <1>       ;; push dx ; ds (UDATA)
41594                              <1>       ;; push dx ; es (UDATA)
41595                              <1>       ;; push dx ; fs (UDATA)
41596                              <1>       ;; push dx ; gs (UDATA)
41597                              <1>       ;
41598                              <1>       ; This is a trick to prevent general protection fault
41599                              <1>       ; during 'iretd' intruction at the end of 'sysrele' (in u1.s):
41600 0000E35B 8EC2                <1>       mov    es, dx ; UDATA
41601 0000E35D 06                  <1>       push   es ; ds (UDATA)
41602 0000E35E 06                  <1>       push   es ; es (UDATA)
41603 0000E35F 06                  <1>       push   es ; fs (UDATA)
41604 0000E360 06                  <1>       push   es ; gs (UDATA)
41605 0000E361 66BA1000            <1>       mov    dx, KDATA
41606 0000E365 8EC2                <1>       mov    es, dx
41607                              <1>       ;
```

```
41608                                  <1>       ;; pushad simulation
41609 0000E367 89E5                    <1>       mov    ebp, esp ; esp before pushad
41610 0000E369 50                      <1>       push   eax ; eax (0)
41611 0000E36A 50                      <1>       push   eax ; ecx (0)
41612 0000E36B 50                      <1>       push   eax ; edx (0)
41613 0000E36C 50                      <1>       push   eax ; ebx (0)
41614 0000E36D 55                      <1>       push   ebp ; esp before pushad
41615 0000E36E 50                      <1>       push   eax ; ebp (0)
41616 0000E36F 50                      <1>       push   eax ; esi (0)
41617 0000E370 50                      <1>       push   eax ; edi (0)
41618                                  <1>       ;
41619 0000E371 A3[64030600]            <1>       mov    [u.r0], eax ; eax = 0
41620 0000E376 8925[60030600]          <1>       mov    [u.usp], esp
41621                                  <1>
41622                                  <1>       ; 02/05/2016
41623                                  <1>       ;inc   byte [sysflg] ; 0FFh -> 0
41624                                  <1>       ;mov   byte [sysflg], 0 ; 04/01/2017
41625 0000E37C 0FB61D[B3030600]        <1>       movzx  ebx, byte [u.uno]
41626 0000E383 6683BB[3E000600]01      <1>       cmp    word [ebx+p.ppid-2], 1 ; MainProg
41627 0000E38B 0F8720E1FFFF            <1>       ja     sysret0 ; 03/05/2016
41628 0000E391 68[AFC40000]            <1>       push   sysret ; *
41629 0000E396 8925[60030600]          <1>       mov    [u.usp], esp
41630 0000E39C E886030000              <1>       call   wswap ; save child process 'u' structure and
41631                                  <1>                    ; registers
41632 0000E3A1 8305[60030600]04        <1>       add    dword [u.usp], 4 ; 03/05/2016
41633                                  <1> sysexec_19: ; 02/05/2016
41634 0000E3A8 C3                      <1>       retn ; * 'sysret' ; byte [sysflg] -> 0FFh
41635                                  <1>
41636                                  <1> readi:
41637                                  <1>       ; 01/05/2016
41638                                  <1>       ; 25/04/2016 - TRDOS 386 (TRDOS v2.0)
41639                                  <1>       ; 20/05/2015 - Retro UNIX 386 v1
41640                                  <1>       ; 11/03/2013 - 31/07/2013 (Retro UNIX 8086 v1)
41641                                  <1>       ;
41642                                  <1>       ; Reads from a file whose the first cluster number in EAX
41643                                  <1>       ;
41644                                  <1>       ; INPUTS ->
41645                                  <1>       ;    EAX - First cluster number of the file
41646                                  <1>       ;    u.count - byte count user desires
41647                                  <1>       ;    u.base - points to user buffer
41648                                  <1>       ;    u.fofp - points to dword with current file offset
41649                                  <1>       ;    i.size - file size
41650                                  <1>       ;    cdev - logical dos drive number of the file
41651                                  <1>       ; OUTPUTS ->
41652                                  <1>       ;    u.count - cleared
41653                                  <1>       ;    u.nread - accumulates total bytes passed back
41654                                  <1>       ;
41655                                  <1>       ; ((EAX)) input/output
41656                                  <1>       ; (Retro UNIX Prototype : 14/12/2012 - 01/03/2013, UNIXCOPY.ASM)
41657                                  <1>         ; ((Modified registers: edx, ebx, ecx, esi, edi))
41658                                  <1>
41659 0000E3A9 31D2                    <1>       xor    edx, edx ; 0
41660 0000E3AB 8915[8C030600]          <1>       mov    [u.nread], edx ; 0
41661 0000E3B1 668915[C4030600]        <1>       mov    [u.pcount], dx ; 19/05/2015
41662 0000E3B8 3915[88030600]          <1>       cmp    [u.count], edx ; 0
41663 0000E3BE 7701                    <1>       ja     short readi_1
41664 0000E3C0 C3                      <1>       retn
41665                                  <1> readi_1:
41666                                  <1> dskr:
41667                                  <1>       ; 01/05/2016
41668                                  <1>       ; 25/04/2016 - TRDOS 386 (TRDOS v2.0)
41669                                  <1>       ; 24/05/2015 - 12/10/2015 (Retro UNIX 386 v1)
41670                                  <1>       ; 26/04/2013 - 03/08/2013 (Retro UNIX 8086 v1)
41671                                  <1> dskr_0:
41672 0000E3C1 8B15[55040600]          <1>         mov  edx, [i.size]
41673 0000E3C7 8B1D[74030600]          <1>       mov    ebx, [u.fofp]
41674 0000E3CD 2B13                    <1>       sub    edx, [ebx]
41675 0000E3CF 7647                    <1>       jna    short dskr_4
41676                                  <1>       ;
41677 0000E3D1 50                      <1>       push   eax ; 01/05/2016
41678 0000E3D2 3B15[88030600]          <1>       cmp     edx, [u.count]
41679 0000E3D8 7306                    <1>       jnb    short dskr_1
41680 0000E3DA 8915[88030600]          <1>       mov    [u.count], edx
41681                                  <1> dskr_1:
41682                                  <1>       ; EAX = First Cluster
41683                                  <1>       ; [Current_Drv] = Physical drive number
41684 0000E3E0 E83B000000              <1>       call   mget_r
41685                                  <1>       ; NOTE: in 'mget_r', relevant sector will be read in buffer
41686                                  <1>       ; if it is not already in buffer !
41687 0000E3E5 BB[8C050600]            <1>       mov    ebx, readi_buffer
41688 0000E3EA 803D[C6030600]00        <1>       cmp    byte [u.kcall], 0 ; the caller is 'namei' sign (=1)
41689 0000E3F1 770F                    <1>       ja     short dskr_3   ; zf=0 -> the caller is 'namei'
41690 0000E3F3 66833D[C4030600]00      <1>       cmp    word [u.pcount], 0
41691 0000E3FB 7705                    <1>       ja     short dskr_3
41692                                  <1> dskr_2:
41693                                  <1>       ; [u.base] = virtual address to transfer (as destination address)
41694 0000E3FD E894010000              <1>       call   trans_addr_w ; translate virtual address to physical (w)
41695                                  <1> dskr_3:
41696                                  <1>       ; EBX (r5) = system (I/O) buffer address -physical-
41697 0000E402 E8F7010000              <1>       call   sioreg
41698 0000E407 87F7                    <1>       xchg   esi, edi
41699                                  <1>       ; EDI = file (user data) offset
41700                                  <1>       ; ESI = sector (I/O) buffer offset
41701                                  <1>       ; ECX = byte count
41702 0000E409 F3A4                    <1>       rep    movsb
41703                                  <1>       ; eax = remain bytes in buffer
41704                                  <1>         ;       (check if remain bytes in the buffer > [u.pcount])
41705 0000E40B 09C0                    <1>       or     eax, eax
41706 0000E40D 75EE                    <1>       jnz    short dskr_2 ; (page end before system buffer end!)
41707 0000E40F 58                      <1>       pop    eax  ; (first cluster number)
41708 0000E410 390D[88030600]          <1>       cmp    [u.count], ecx ; 0
41709 0000E416 77A9                    <1>       ja     short dskr_0
```

```
41710                                  <1> dskr_4:
41711 0000E418 C605[C6030600]00        <1>       mov    byte [u.kcall], 0
41712 0000E41F C3                      <1>       retn
41713                                  <1>
41714                                  <1> mget_r:
41715                                  <1>       ; 24/10/2016
41716                                  <1>       ; 22/10/2016
41717                                  <1>       ; 12/10/2016
41718                                  <1>       ; 29/04/2016
41719                                  <1>       ; 25/04/2016 - TRDOS 386 (TRDOS v2.0)
41720                                  <1>       ; 03/06/2015 (Retro UNIX 386 v1, 'mget', u.5s)
41721                                  <1>       ; 22/03/2013 - 31/07/2013 (Retro UNIX 8086 v1)
41722                                  <1>       ;
41723                                  <1>       ; Get existing or (allocate) a new disk block for file
41724                                  <1>       ;
41725                                  <1>       ; INPUTS ->
41726                                  <1>       ;    [u.fofp] = file offset pointer
41727                                  <1>       ;    EAX = First Cluster
41728                                  <1>       ;    [cdev] = Logical dos drive number
41729                                  <1>       ;    ([u.off] = file offset)
41730                                  <1>       ; OUTPUTS ->
41731                                  <1>       ;    EAX = logical sector number
41732                                  <1>       ;    ESI = Logical Dos Drive Description Table address
41733                                  <1>       ;
41734                                  <1>       ; Modified registers: EDX, EBX, ECX, ESI, EDI
41735                                  <1>
41736 0000E420 8B35[74030600]          <1>       mov     esi, [u.fofp]
41737 0000E426 8B1E                    <1>       mov    ebx, [esi] ; (u.off)
41738                                  <1>
41739 0000E428 29C9                    <1>       sub    ecx, ecx
41740 0000E42A 8A2D[46030600]          <1>       mov    ch, [cdev]
41741                                  <1>
41742 0000E430 BE00010900              <1>       mov    esi, Logical_DOSDisks
41743 0000E435 01CE                    <1>       add    esi, ecx
41744                                  <1>
41745 0000E437 380D[005B0100]          <1>       cmp    [readi.valid], cl ; 0
41746 0000E43D 7649                    <1>       jna    short mget_r_0
41747                                  <1>
41748 0000E43F 3A2D[015B0100]          <1>       cmp    ch, [readi.drv]
41749 0000E445 7541                    <1>       jne    short mget_r_0
41750                                  <1>
41751 0000E447 3B05[145B0100]          <1>       cmp    eax, [readi.fclust]
41752 0000E44D 7565                    <1>       jne    short mget_r_3
41753                                  <1>
41754 0000E44F 89D8                    <1>       mov    eax, ebx ; file offset
41755 0000E451 668B0D[085B0100]        <1>       mov    cx, [readi.bpc]
41756 0000E458 41                      <1>       inc    ecx ; <= 65536
41757 0000E459 29D2                    <1>       sub    edx, edx
41758 0000E45B F7F1                    <1>       div    ecx
41759                                  <1>
41760 0000E45D 8B3D[105B0100]          <1>       mov    edi, [readi.c_index] ; cluster index
41761                                  <1>
41762 0000E463 39F8                    <1>       cmp    eax, edi
41763 0000E465 757A                    <1>         jne     short mget_r_4  ; (*)
41764                                  <1>
41765                                  <1>       ; edx = byte offset in cluster (<= 65535)
41766 0000E467 668915[0A5B0100]        <1>       mov    [readi.offset], dx
41767 0000E46E 66C1EA09                <1>       shr    dx, 9 ; / 512
41768 0000E472 8815[035B0100]          <1>       mov    [readi.s_index], dl ; sector index in cluster (0 to spc -1)
41769                                  <1>
41770 0000E478 A1[0C5B0100]            <1>       mov    eax, [readi.cluster]  ; > 0 if [readi.valid] = 1
41771 0000E47D 8B15[185B0100]          <1>       mov    edx, [readi.fs_index]
41772 0000E483 E99A000000              <1>         jmp     mget_r_7
41773                                  <1>
41774                                  <1> mget_r_0:
41775 0000E488 882D[015B0100]          <1>       mov    [readi.drv], ch ; physical drive number
41776 0000E48E 807E0300                <1>       cmp    byte [esi+LD_FATType], 0
41777 0000E492 7707                    <1>       ja     short mget_r_1
41778 0000E494 8A4E12                  <1>       mov    cl, [esi+LD_FS_BytesPerSec+1]
41779 0000E497 D0E9                    <1>       shr    cl, 1 ;  ; 1 for 512 bytes, 4 for 2048 bytes
41780 0000E499 EB03                    <1>       jmp    short mget_r_2
41781                                  <1> mget_r_1:
41782 0000E49B 8A4E13                  <1>       mov    cl, [esi+LD_BPB+BPB_SecPerClust]
41783                                  <1> mget_r_2:
41784 0000E49E 880D[025B0100]          <1>       mov    [readi.spc], cl  ; sectors per cluster
41785                                  <1>       ; NOTE: readi bytes per sector value is always 512 !
41786 0000E4A4 66C1E109                <1>       shl    cx, 9 ; * 512
41787 0000E4A8 6649                    <1>       dec    cx ; bytes per cluster - 1
41788 0000E4AA 66890D[085B0100]        <1>       mov    [readi.bpc], cx
41789 0000E4B1 6629C9                  <1>       sub    cx, cx
41790                                  <1> mget_r_3:
41791 0000E4B4 A3[145B0100]            <1>       mov    [readi.fclust], eax ; first cluster (or FDT address)
41792 0000E4B9 880D[005B0100]          <1>       mov    [readi.valid], cl ; 0
41793                                  <1>       ;mov   [readi.s_index], cl ; 0
41794                                  <1>       ;mov   [readi.offset], cx ; 0
41795 0000E4BF 890D[105B0100]          <1>       mov    [readi.c_index], ecx ; 0
41796 0000E4C5 890D[0C5B0100]          <1>       mov    [readi.cluster], ecx ; 0
41797 0000E4CB 890D[045B0100]          <1>       mov    [readi.sector], ecx ; 0
41798                                  <1>
41799 0000E4D1 89D8                    <1>       mov    eax, ebx ; file offset
41800 0000E4D3 668B0D[085B0100]        <1>       mov    cx, [readi.bpc]
41801 0000E4DA 41                      <1>       inc    ecx ; <= 65536
41802 0000E4DB 29D2                    <1>       sub    edx, edx
41803 0000E4DD F7F1                    <1>       div    ecx
41804                                  <1>       ;mov   edi, [readi.c_index] ; previous cluster index
41805 0000E4DF 29FF                    <1>       sub    edi, edi
41806                                  <1> mget_r_4:
41807 0000E4E1 A3[105B0100]            <1>       mov    [readi.c_index], eax ; cluster index
41808                                  <1>       ; edx = byte offset in cluster (<= 65535)
41809 0000E4E6 668915[0A5B0100]        <1>       mov    [readi.offset], dx
41810 0000E4ED 66C1EA09                <1>       shr    dx, 9 ; / 512
41811 0000E4F1 8815[035B0100]          <1>       mov    [readi.s_index], dl ; sector index in cluster (0 to spc -1)
```

```
41812                                  <1>
41813 0000E4F7 89C1                    <1>       mov    ecx, eax ; current cluster index
41814 0000E4F9 A1[145B0100]            <1>       mov    eax, [readi.fclust]
41815 0000E4FE 09C9                    <1>       or     ecx, ecx ; cluster index
41816 0000E500 741B                    <1>       jz     short mget_r_6
41817                                  <1>
41818 0000E502 39CF                    <1>       cmp    edi, ecx
41819 0000E504 7710                    <1>       ja     short mget_r_5 ; old cluster index is higher
41820 0000E506 8B15[0C5B0100]          <1>       mov    edx, [readi.cluster]
41821 0000E50C 21D2                    <1>       and    edx, edx
41822 0000E50E 7406                    <1>       jz     short mget_r_5
41823                                  <1>       ; valid 'readi' parameters (*)
41824 0000E510 89D0                    <1>       mov    eax, edx
41825 0000E512 29F9                    <1>       sub    ecx, edi
41826 0000E514 740C                    <1>       jz     short mget_r_7
41827                                  <1> mget_r_5:
41828                                  <1>       ; EAX = Beginning cluster
41829                                  <1>       ; EDX = Sector index in disk/file section
41830                                  <1>       ;       (Only for SINGLIX file system!)
41831                                  <1>       ; ECX = Cluster sequence number after the beginning cluster
41832                                  <1>       ; ESI = Logical DOS Drive Description Table address
41833 0000E516 E811DEFFFF              <1>       call   get_cluster_by_index
41834 0000E51B 724E                    <1>       jc     short mget_r_err
41835                                  <1>       ; EAX = Cluster number
41836                                  <1> mget_r_6:
41837 0000E51D A3[0C5B0100]            <1>       mov    [readi.cluster], eax ; FDT number for Singlix File System
41838                                  <1> mget_r_7:
41839 0000E522 807E0300                <1>       cmp    byte [esi+LD_FATType], 0
41840 0000E526 765F                    <1>       jna    short mget_r_12
41841                                  <1>
41842 0000E528 83E802                  <1>       sub    eax, 2
41843 0000E52B 0FB615[025B0100]        <1>       movzx  edx, byte [readi.spc]
41844 0000E532 F7E2                    <1>       mul    edx
41845                                  <1>
41846 0000E534 034668                  <1>       add    eax, [esi+LD_DATABegin]
41847 0000E537 8A15[035B0100]          <1>       mov    dl, [readi.s_index]
41848 0000E53D 01D0                    <1>       add    eax, edx
41849                                  <1> mget_r_8:
41850                                  <1>       ; eax = logical sector number
41851 0000E53F 803D[005B0100]00        <1>       cmp    byte [readi.valid], 0
41852 0000E546 7608                    <1>       jna    short mget_r_9
41853 0000E548 3B05[045B0100]          <1>       cmp    eax, [readi.sector]
41854 0000E54E 7436                    <1>       je     short mget_r_11 ; sector is already in 'readi' buffer
41855                                  <1> mget_r_9:
41856 0000E550 A3[045B0100]            <1>       mov    [readi.sector], eax
41857 0000E555 BB[8C050600]            <1>       mov    ebx, readi_buffer ; buffer address
41858 0000E55A B901000000              <1>       mov    ecx, 1
41859                                  <1>       ; 29/04/2016
41860                                  <1>       ;xor   dl, dl
41861                                  <1>
41862                                  <1>       ; EAX = Logical sector number
41863                                  <1>       ; ECX = Sector count
41864                                  <1>       ; EBX = Buffer address
41865                                  <1>       ; (EDX = 0)
41866                                  <1>       ; ESI = Logical DOS drive description table address
41867                                  <1>
41868 0000E55F E83A0C0000              <1>       call   disk_read
41869 0000E564 7314                    <1>       jnc    short mget_r_10
41870                                  <1>
41871                                  <1>       ; 22/10/2016 (15h -> 17)
41872 0000E566 B811000000              <1>       mov    eax, 17 ; Drive not ready or read error !
41873                                  <1> mget_r_err:
41874 0000E56B A3[C8030600]            <1>       mov    [u.error], eax
41875                                  <1>       ; 12/10/2016
41876 0000E570 A3[64030600]            <1>       mov    [u.r0], eax
41877 0000E575 E915DFFFFF              <1>       jmp    error
41878                                  <1> mget_r_10:
41879 0000E57A C605[005B0100]01        <1>       mov    byte [readi.valid], 1 ; 24/10/2016
41880 0000E581 A1[045B0100]            <1>       mov    eax, [readi.sector]
41881                                  <1> mget_r_11:
41882 0000E586 C3                      <1>       retn
41883                                  <1> mget_r_12:
41884                                  <1>       ; EAX = FDT number
41885                                  <1>       ; EDX = Sector index from FDT sector (0,1,2,3,4...)
41886 0000E587 40                      <1>       inc    eax ; the first data sector in FS disk section
41887 0000E588 8915[185B0100]          <1>       mov    [readi.fs_index], edx
41888 0000E58E 01D0                    <1>       add    eax, edx
41889 0000E590 EBAD                    <1>       jmp    short mget_r_8
41890                                  <1>
41891                                  <1> trans_addr_r:
41892                                  <1>       ; 12/10/2016
41893                                  <1>       ; 02/05/2016 - TRDOS 386 (TRDOS v2.0)
41894                                  <1>       ; Translate virtual address to physical address
41895                                  <1>       ; for reading from user's memory space
41896                                  <1>       ; 04/06/2015 - 18/10/2015 (Retro UNIX 386 v1)
41897                                  <1>
41898 0000E592 31D2                    <1>       xor    edx, edx ; 0 (read access sign)
41899 0000E594 EB04                    <1>       jmp    short trans_addr_rw
41900                                  <1>
41901                                  <1> trans_addr_w:
41902                                  <1>       ; 12/10/2016
41903                                  <1>       ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
41904                                  <1>       ; Translate virtual address to physical address
41905                                  <1>       ; for writing to user's memory space
41906                                  <1>       ; 04/06/2015 - 18/10/2015 (Retro UNIX 386 v1)
41907                                  <1>
41908 0000E596 29D2                    <1>       sub    edx, edx
41909 0000E598 FEC2                    <1>       inc    dl ; 1 (write access sign)
41910                                  <1> trans_addr_rw:
41911 0000E59A 50                      <1>       push   eax
41912 0000E59B 53                      <1>       push   ebx
41913 0000E59C 52                      <1>       push   edx ; r/w sign (in DL)
```

```
41914                                  <1>       ;
41915 0000E59D 8B1D[84030600]          <1>       mov    ebx, [u.base]
41916 0000E5A3 E8E66CFFFF              <1>       call   get_physical_addr ; get physical address
41917 0000E5A8 730F                    <1>       jnc    short passc_0
41918 0000E5AA A3[C8030600]            <1>       mov    [u.error], eax
41919 0000E5AF A3[64030600]            <1>       mov    [u.r0], eax ; 12/10/2016
41920                                  <1>       ;pop   edx
41921                                  <1>       ;pop   ebx
41922                                  <1>       ;pop   eax
41923 0000E5B4 E9D6DEFFFF              <1>       jmp    error
41924                                  <1> passc_0:
41925 0000E5B9 F6C202                  <1>       test   dl, PTE_A_WRITE ; writable page
41926 0000E5BC 5A                      <1>       pop    edx
41927 0000E5BD 751C                    <1>       jnz    short passc_1
41928                                  <1>
41929 0000E5BF 20D2                    <1>       and    dl, dl
41930 0000E5C1 7418                    <1>       jz     short passc_1
41931                                  <1>       ; read only (duplicated) page -must be copied to a new page-
41932                                  <1>       ; EBX = linear address
41933 0000E5C3 51                      <1>       push   ecx
41934 0000E5C4 E85E69FFFF              <1>       call   copy_page
41935 0000E5C9 59                      <1>       pop    ecx
41936 0000E5CA 721E                    <1>       jc     short passc_2
41937 0000E5CC 50                      <1>       push   eax ; physical address of the new/allocated page
41938 0000E5CD E8E66BFFFF              <1>       call   add_to_swap_queue
41939 0000E5D2 58                      <1>       pop    eax
41940 0000E5D3 81E3FF0F0000            <1>       and    ebx, PAGE_OFF ; 0FFFh
41941                                  <1>       ;mov   ecx, PAGE_SIZE
41942                                  <1>       ;sub   ecx, ebx
41943 0000E5D9 01D8                    <1>       add    eax, ebx
41944                                  <1> passc_1:
41945 0000E5DB A3[C0030600]            <1>       mov    [u.pbase], eax ; physical address
41946 0000E5E0 66890D[C4030600]        <1>       mov    [u.pcount], cx ; remain byte count in page (1-4096)
41947 0000E5E7 5B                      <1>       pop    ebx
41948 0000E5E8 58                      <1>       pop    eax
41949 0000E5E9 C3                      <1>       retn
41950                                  <1> passc_2:
41951 0000E5EA B804000000              <1>       mov    eax, ERR_MINOR_IM ; "Insufficient memory !" error
41952 0000E5EF A3[64030600]            <1>       mov    [u.r0], eax ; 12/10/2016
41953 0000E5F4 A3[C8030600]            <1>       mov    dword [u.error], eax
41954                                  <1>       ;pop   ebx
41955                                  <1>       ;pop   eax
41956 0000E5F9 E991DEFFFF              <1>       jmp    error
41957                                  <1>
41958                                  <1> sioreg:
41959                                  <1>       ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
41960                                  <1>       ; 19/05/2015 - 25/07/2015 (Retro UNIX 386 v1)
41961                                  <1>       ; 12/03/2013 - 22/07/2013 (Retro UNIX 8086 v1)
41962                                  <1>       ; INPUTS ->
41963                                  <1>       ;     EBX = system buffer (data) address (r5)
41964                                  <1>       ;     [u.fofp] = pointer to file offset pointer
41965                                  <1>       ;     [u.base] = virtual address of the user buffer
41966                                  <1>       ;     [u.pbase] = physical address of the user buffer
41967                                  <1>       ;     [u.count] = byte count
41968                                  <1>       ;     [u.pcount] = byte count within page frame
41969                                  <1>       ; OUTPUTS ->
41970                                  <1>       ;     ESI = user data offset (r1)
41971                                  <1>       ;     EDI = system (I/O) buffer offset (r2)
41972                                  <1>       ;     ECX = byte count (r3)
41973                                  <1>       ;     EAX = remain bytes after byte count within page frame
41974                                  <1>       ;      (If EAX > 0, transfer will continue from the next page)
41975                                  <1>         ;
41976                                  <1>       ; ((Modified registers:  EDX))
41977                                  <1>
41978 0000E5FE 8B35[74030600]          <1>         mov     esi, [u.fofp]
41979 0000E604 8B3E                    <1>         mov     edi, [esi]
41980 0000E606 89F9                    <1>       mov    ecx, edi
41981 0000E608 81C900FEFFFF            <1>       or     ecx, 0FFFFFE00h
41982 0000E60E 81E7FF010000            <1>       and    edi, 1FFh
41983 0000E614 01DF                    <1>       add    edi, ebx ; EBX = system buffer (data) address
41984 0000E616 F7D9                    <1>       neg    ecx
41985 0000E618 3B0D[88030600]          <1>       cmp    ecx, [u.count]
41986 0000E61E 7606                    <1>       jna    short sioreg_0
41987 0000E620 8B0D[88030600]          <1>       mov    ecx, [u.count]
41988                                  <1> sioreg_0:
41989 0000E626 803D[C6030600]00        <1>       cmp    byte [u.kcall], 0
41990 0000E62D 7613                    <1>       jna    short sioreg_1
41991                                  <1>        ; the caller is 'mkdir' or 'namei'
41992 0000E62F A1[84030600]            <1>       mov    eax, [u.base]
41993 0000E634 A3[C0030600]            <1>       mov    [u.pbase], eax ; physical address = virtual address
41994 0000E639 66890D[C4030600]        <1>       mov    word [u.pcount], cx ; remain bytes in buffer (1 sector)
41995 0000E640 EB0B                    <1>       jmp    short sioreg_2
41996                                  <1> sioreg_1:
41997 0000E642 0FB715[C4030600]        <1>       movzx  edx, word [u.pcount]
41998 0000E649 39D1                    <1>       cmp    ecx, edx
41999 0000E64B 772A                    <1>       ja     short sioreg_4 ; transfer count > [u.pcount]
42000                                  <1> sioreg_2: ; 2:
42001 0000E64D 31C0                    <1>       xor    eax, eax
42002                                  <1> sioreg_3:
42003 0000E64F 010D[8C030600]          <1>       add    [u.nread], ecx
42004 0000E655 290D[88030600]          <1>       sub    [u.count], ecx
42005 0000E65B 010D[84030600]          <1>       add    [u.base], ecx
42006 0000E661 010E                    <1>         add  [esi], ecx
42007 0000E663 8B35[C0030600]          <1>       mov    esi, [u.pbase]
42008 0000E669 66290D[C4030600]        <1>       sub    [u.pcount], cx
42009 0000E670 010D[C0030600]          <1>       add    [u.pbase], ecx
42010 0000E676 C3                      <1>         retn
42011                                  <1> sioreg_4:
42012                                  <1>       ; transfer count > [u.pcount]
42013                                  <1>       ; (ecx > edx)
42014 0000E677 89C8                    <1>       mov    eax, ecx
42015 0000E679 29D0                    <1>       sub    eax, edx ; remain bytes for 1 sector (block) transfer
```

```
42016 0000E67B 89D1              <1>       mov    ecx, edx ; current transfer count = [u.pcount]
42017 0000E67D EBD0              <1>       jmp    short sioreg_3
42018                            <1>
42019                            <1> tswitch: ; Retro UNIX 386 v1
42020                            <1> tswap:
42021                            <1>       ; 16/01/2017
42022                            <1>       ; 21/05/2016 - TRDOS 386 (TRDOS v2.0)
42023                            <1>       ; 10/05/2015 - 01/09/2015 (Retro UNIX 386 v1)
42024                            <1>       ; 14/04/2013 - 14/02/2014 (Retro UNIX 8086 v1)
42025                            <1>       ; time out swap, called when a user times out.
42026                            <1>       ; the user is put on the low priority queue.
42027                            <1>       ; This is done by making a link from the last user
42028                            <1>       ; on the low priority queue to him via a call to 'putlu'.
42029                            <1>       ; then he is swapped out.
42030                            <1>
42031                            <1>       ; TRDOS 386 (TRDOS v2.0) modification ->  ** 21/05/2016 **
42032                            <1>       ;     * when a high priority (event) process will be stopped
42033                            <1>       ;     (swapped out, swithched out/off), 'tswap/tswitch' will
42034                            <1>       ;     not add it to a run queue.
42035                            <1>       ;     /// What for: Process may be already in a run queue,
42036                            <1>       ;     it is unspeficied state because process might be started
42037                            <1>       ;     by a timer event which does not regard previous priority
42038                            <1>       ;     level and run queue of the process (for fast executing!).
42039                            <1>       ;     After the 'run for event', process will be sequenced
42040                            <1>       ;     to run by it's actual run queue. ///
42041                            <1>       ;
42042                            <1>       ; Retro UNIX 386 v1 modification ->
42043                            <1>       ;       swap (software task switch) is performed by changing
42044                            <1>       ;     user's page directory (u.pgdir) instead of segment change
42045                            <1>       ;     as in Retro UNIX 8086 v1.
42046                            <1>       ;
42047                            <1>       ; RETRO UNIX 8086 v1 modification ->
42048                            <1>       ;       'swap to disk' is replaced with 'change running segment'
42049                            <1>       ;     according to 8086 cpu (x86 real mode) architecture.
42050                            <1>       ;     pdp-11 was using 64KB uniform memory while IBM PC
42051                            <1>       ;     compatibles was using 1MB segmented memory
42052                            <1>       ;     in 8086/8088 times.
42053                            <1>       ;
42054                            <1>       ; INPUTS ->
42055                            <1>       ;    u.uno - users process number
42056                            <1>       ;    runq+4 - lowest priority queue
42057                            <1>       ; OUTPUTS ->
42058                            <1>       ;    r0 - users process number
42059                            <1>       ;    r2 - lowest priority queue address
42060                            <1>       ;
42061                            <1>       ; ((AX = R0, BX = R2)) output
42062                            <1>       ; ((Modified registers: EDX, EBX, ECX, ESI, EDI))
42063                            <1>       ;
42064                            <1>
42065                            <1>       NOTE:
42066                            <1>       ;* [u.pri] priority level is specified by run queue which is process
42067                            <1>       ;  comes to run from.
42068                            <1>       ;* Initial [u.pri] is 1 ('normal/regular') for programs
42069                            <1>       ;  (which are launched by MainProg or 'sysexec'), it is changed
42070                            <1>       ;  to 2 ('high') by timer event, if program uses 'systimer' system call.
42071                            <1>       ;* Program (Process) also can change it's running priority
42072                            <1>       ;  from 1 to 0 or up to 2 by using 'syspri' system call; but,
42073                            <1>       ;  if program selects priority level 2 (high) for running, next time
42074                            <1>       ;  it is reduced to 1 (normal/regular) because 'syspri' adds this
42075                            <1>       ;  program to 'run for normal' queue while running duration is a bit
42076                            <1>       ;  protected from swap/switch out immediate, behalf of other high
42077                            <1>       ;  priority process in sequence. Program (with high priority) will not
42078                            <1>       ;  be swapped/switched out (by timer event) before it's time quantum
42079                            <1>       ;  will be elapsed, but, this will be temporary if program is not using
42080                            <1>       ;  timer event function.
42081                            <1>
42082                            <1>       ;For example:
42083                            <1>       ;If a process frequently gets a timer event, it runs at high priority
42084                            <1>       ;level but when it returns from running it returns to actual run queue,
42085                            <1>       ;not to 'run for event' queue again.
42086                            <1>       ;'tswap' will not change the sequence at return/stop(swap out) stage.
42087                            <1>       ;But if priority level not high (=2, 'run for event'), 'tswap/tswitch'
42088                            <1>       ;will add the stopping process to relevant run queue according to
42089                            <1>       ;[u.pri] priority level.
42090                            <1>
42091                            <1>       ; 16/01/2017
42092 0000E67F BB[54030600]      <1>       mov    ebx, runq+2  ; 'runq_normal' ; normal/regular priority
42093                            <1>       ; 21/05/2016
42094                            <1>       ;cmp   byte [u.pri], 2     ; high priority (run for event) ?
42095                            <1>       ;jnb   short swap
42096                            <1>       ; 16/01/2017
42097                            <1>       ; (Normal and also high/event priority processes will be added to
42098                            <1>       ; normal priority run queue for ensuring circular running sequence!)
42099                            <1>       ; (Timer interrupt or 'syspri' system call may change priority and run
42100                            <1>       ; queue to high/event level.)
42101 0000E684 803D[A9030600]00  <1>       cmp    byte [u.pri], 0
42102 0000E68B 7702              <1>       ja     short tswap_1; normal priority run queue
42103                            <1>       ;
42104 0000E68D 43                <1>       inc    ebx
42105 0000E68E 43                <1>       inc    ebx          ; runq+4, 'runq_background', low priority
42106                            <1> tswap_1:
42107 0000E68F A0[B3030600]      <1>       mov    al, [u.uno]
42108                            <1>                     ; movb u.uno,r1 / move users process number to r1
42109                            <1>             ; mov  $runq+4,r2
42110                            <1>                     ; / move lowest priority queue address to r2
42111                            <1>             ; ebx = run queue
42112 0000E694 E8FE000000        <1>       call   putlu
42113                            <1>             ; jsr r0,putlu / create link from last user on Q to
42114                            <1>                         ; / u.uno's user
42115                            <1>
42116                            <1> switch: ; Retro UNIX 386 v1
42117                            <1> swap:
```

```
                                        <1>       ; 02/01/2017
                                        <1>       ; 21/05/2016
                                        <1>       ; 20/05/2016
                                        <1>       ; 02/05/2016
                                        <1>       ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
                                        <1>       ; 10/05/2015 - 02/09/2015 (Retro UNIX 386 v1)
                                        <1>       ; 14/04/2013 - 08/03/2014 (Retro UNIX 8086 v1)
                                        <1>       ;
                                        <1>       ; 'swap' is routine that controls the swapping of processes
                                        <1>       ; in and out of core.
                                        <1>       ;
                                        <1>       ; TRDOS 386 (TRDOS v2.0) modification ->  ** 20/05/2016 **
                                        <1>       ;     * 3 different priority level is applied
                                        <1>       ;     (just as original unix v1)
                                        <1>       ;     1) high priority (event) run queue, 'runq_event'
                                        <1>       ;     2) normal priority (regular) run queue, 'runq_normal'
                                        <1>       ;     3) low priority (background) run queue, 'runq_backgroud'
                                        <1>       ;     'swap' code will run a process which has max. priority
                                        <1>       ;       (for earliest event at first)
                                        <1>       ;
                                        <1>       ; Retro UNIX 386 v1 modification ->
                                        <1>       ;       swap (software task switch) is performed by changing
                                        <1>       ;     user's page directory (u.pgdir) instead of segment change
                                        <1>       ;     as in Retro UNIX 8086 v1.
                                        <1>       ;
                                        <1>       ; RETRO UNIX 8086 v1 modification ->
                                        <1>       ;       'swap to disk' is replaced with 'change running segment'
                                        <1>       ;     according to 8086 cpu (x86 real mode) architecture.
                                        <1>       ;     pdp-11 was using 64KB uniform memory while IBM PC
                                        <1>       ;     compatibles was using 1MB segmented memory
                                        <1>       ;     in 8086/8088 times.
                                        <1>       ;
                                        <1>       ; INPUTS ->
                                        <1>       ;    runq table - contains processes to run.
                                        <1>       ;    p.link - contains next process in line to be run.
                                        <1>       ;    u.uno - process number of process in core
                                        <1>       ;    s.stack - swap stack used as an internal stack for swapping.
                                        <1>       ; OUTPUTS ->
                                        <1>       ;    (original unix v1 -> present process to its disk block)
                                        <1>       ;    (original unix v1 -> new process into core ->
                                        <1>       ;        Retro Unix 8086 v1 -> segment registers changed
                                        <1>       ;        for new process)
                                        <1>       ;    u.quant = 3 (Time quantum for a process)
                                        <1>       ;     ((INT 1Ch count down speed -> 18.2 times per second)
                                        <1>       ;    RETRO UNIX 8086 v1 will use INT 1Ch (18.2 times per second)
                                        <1>       ;      for now, it will swap the process if there is not
                                        <1>       ;      a keyboard event (keystroke) (Int 15h, function 4Fh)
                                        <1>       ;      or will count down from 3 to 0 even if there is a
                                        <1>       ;       keyboard event locking due to repetitive key strokes.
                                        <1>       ;      u.quant will be reset to 3 for RETRO UNIX 8086 v1.
                                        <1>       ;
                                        <1>       ; ((Modified registers: EAX, EDX, EBX, ECX, ESI, EDI))
                                        <1>
                                        <1>       ;NOTE:
                                        <1>       ;High priority queue is the first for selecting a process to run.
                                        <1>       ;If there is not a process in high priority level run queue,
                                        <1>       ;a process in normal priority run queue will be selected
                                        <1>       ;or a proces in low priority run queue will be selected if normal
                                        <1>       ;priority level run queue is empty.
                                        <1>
                                        <1>       ; 21/05/2016 -(3 priority levels, 3 run queues)
0000E699 BE[52030600]                   <1>       mov   esi, runq ; 'runq_event' ; high priority, 'run for event'
0000E69E C605[5C5B0100]03               <1>       mov   byte [priority], 3 ; high priority + 1
0000E6A5 31DB                           <1>       xor   ebx, ebx ; 02/01/2017
                                        <1> swap_0: ; 1: / search runq table for highest priority process
0000E6A7 66AD                           <1>       lodsw  ; mov ax, [esi], add esi+2
                                        <1>       ;xor   ebx, ebx ; 02/05/2016
0000E6A9 6621C0                         <1>       and   ax, ax ; are there any processes to run in this Q entry
0000E6AC 750E                           <1>       jnz   short swap_2
                                        <1>       ; 21/05/2026
                                        <1>       ; runq_normal = runq+2, runq_background = runq+4
0000E6AE FE0D[5C5B0100]                 <1>       dec   byte [priority] ; 3 -> 3, 2 -> 1, 1-> 0
0000E6B4 75F1                           <1>       jnz   short swap_0
                                        <1>       ;cmp   esi, runq+6  ; if zero compare address to end of table
                                        <1>       ;jb    short swap_0 ; if not at end, go back
                                        <1> swap_1:
                                        <1>       ; 02/05/2016
                                        <1>       ; 29/04/2016 (TRDOS 386 = TRDOS v2.0)
                                        <1>       ; No user process to run...
                                        <1>       ; Run the kernel process... MainProg: Internal Command Interpreter
0000E6B6 FEC0                           <1>       inc   al ; mov al, 1  ; process number of MainProg
0000E6B8 FEC3                           <1>       inc   bl ; mov bl, al ; 1
0000E6BA EB1E                           <1>       jmp   short swap_4
                                        <1> swap_2:
                                        <1>       ; 21/05/2016
0000E6BC FE0D[5C5B0100]                 <1>       dec   byte [priority] ; priority level of present user/process
                                        <1>                             ; 0, 1, 2
0000E6C2 4E                             <1>       dec   esi
0000E6C3 4E                             <1>         dec     esi
                                        <1>       ;
0000E6C4 88C3                           <1>       mov   bl, al
0000E6C6 38E0                           <1>       cmp   al, ah ; is there only 1 process in the queue to be run
0000E6C8 740A                           <1>       je    short swap_3 ; yes
0000E6CA 8AA3[9F000600]                 <1>       mov   ah, [ebx+p.link-1]
0000E6D0 8826                           <1>             mov   [esi], ah ; move next process in line into run queue
0000E6D2 EB06                           <1>       jmp   short swap_4
                                        <1> swap_3:
0000E6D4 6631D2                         <1>       xor   dx, dx
0000E6D7 668916                         <1>       mov   [esi], dx ; zero the entry; no processes on the Q
                                        <1> swap_4:
0000E6DA 8A25[B3030600]                 <1>       mov   ah, [u.uno]
0000E6E0 38C4                           <1>       cmp   ah, al ;is this process the same as the process in core?
```

```
42220 0000E6E2 743B                <1>             je    short swap_8 ; yes, don't have to swap
42221 0000E6E4 08E4                <1>       or    ah, ah  ; is the process # = 0
42222 0000E6E6 740D                <1>             jz    short swap_6 ; 'sysexit'
42223                              <1>       ;cmp  ah, al ;is this process the same as the process in core?
42224                              <1>             ;je   short swap_8 ; yes, don't have to swap
42225 0000E6E8 8925[60030600]      <1>       mov   [u.usp], esp ; return  address for 'syswait' & 'sleep'
42226 0000E6EE E834000000          <1>       call  wswap   ; write out core to disk
42227 0000E6F3 EB1C                <1>       jmp   short swap_7
42228                              <1> swap_6:
42229                              <1>       ; Deallocate memory pages belong to the process
42230                              <1>       ; which is being terminated.
42231                              <1>       ; (Retro UNIX 386 v1 modification !)
42232                              <1>       ;
42233 0000E6F5 53                  <1>       push  ebx
42234 0000E6F6 A1[B8030600]        <1>       mov   eax, [u.pgdir]  ; page directory of the process
42235 0000E6FB 8B1D[BC030600]      <1>       mov   ebx, [u.ppgdir] ; page directory of the parent process
42236 0000E701 E8AC65FFFF          <1>       call  deallocate_page_dir
42237 0000E706 A1[B4030600]        <1>       mov   eax, [u.upage] ; 'user' structure page of the process
42238 0000E70B E84766FFFF          <1>       call  deallocate_page
42239 0000E710 5B                  <1>       pop   ebx
42240                              <1> swap_7:
42241 0000E711 C0E302              <1>       shl   bl, 2 ; * 4
42242 0000E714 8B83[BC000600]      <1>       mov   eax, [ebx+p.upage-4] ; the 'u' page of the new process
42243 0000E71A E840000000          <1>       call  rswap ; read new process into core
42244                              <1> swap_8:
42245                              <1>       ; Retro UNIX  8086 v1 modification !
42246 0000E71F C605[A8030600]04    <1>       mov   byte [u.quant], time_count
42247 0000E726 C3                  <1>       retn
42248                              <1>
42249                              <1> wswap:  ; < swap out, swap to disk >
42250                              <1>       ; 28/02/2017 (fnsave)
42251                              <1>       ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
42252                              <1>       ; 09/05/2015 (Retro UNIX 386 v1)
42253                              <1>       ; 26/05/2013 - 08/03/2014 (Retro UNIX 8086 v1)
42254                              <1>       ; 'wswap' writes out the process that is in core onto its
42255                              <1>       ; appropriate disk area.
42256                              <1>       ;
42257                              <1>       ; Retro UNIX 386 v1 modification ->
42258                              <1>       ;       User (u) structure content and the user's register content
42259                              <1>       ;     will be copied to the process's/user's UPAGE (a page for
42260                              <1>       ;     saving 'u' structure and user registers for task switching).
42261                              <1>       ;     u.usp - points to kernel stack address which contains
42262                              <1>       ;             user's registers while entering system call.
42263                              <1>       ;     u.sp  - points to kernel stack address
42264                              <1>       ;             to return from system call -for IRET-.
42265                              <1>       ;     [u.usp]+32+16 = [u.sp]
42266                              <1>       ;     [u.usp] -> edi, esi, ebp, esp (= [u.usp]+32), ebx,
42267                              <1>       ;             edx, ecx, eax, gs, fs, es, ds, -> [u.sp].
42268                              <1>       ;
42269                              <1>       ; Retro UNIX 8086 v1 modification ->
42270                              <1>       ;       'swap to disk' is replaced with 'change running segment'
42271                              <1>       ;     according to 8086 cpu (x86 real mode) architecture.
42272                              <1>       ;     pdp-11 was using 64KB uniform memory while IBM PC
42273                              <1>       ;     compatibles was using 1MB segmented memory
42274                              <1>       ;     in 8086/8088 times.
42275                              <1>       ;
42276                              <1>       ; INPUTS ->
42277                              <1>       ;    u.break - points to end of program
42278                              <1>       ;    u.usp - stack pointer at the moment of swap
42279                              <1>       ;    core - beginning of process program
42280                              <1>       ;    ecore - end of core
42281                              <1>       ;    user - start of user parameter area
42282                              <1>       ;    u.uno - user process number
42283                              <1>       ;    p.dska - holds block number of process
42284                              <1>       ; OUTPUTS ->
42285                              <1>       ;    swp I/O queue
42286                              <1>       ;    p.break - negative word count of process
42287                              <1>       ;    r1 - process disk address
42288                              <1>       ;    r2 - negative word count
42289                              <1>       ;
42290                              <1>       ; RETRO UNIX 8086 v1 input/output:
42291                              <1>       ;
42292                              <1>       ; INPUTS ->
42293                              <1>       ;    u.uno - process number (to be swapped out)
42294                              <1>       ; OUTPUTS ->
42295                              <1>       ;    none
42296                              <1>       ;
42297                              <1>       ;   ((Modified registers: ECX, ESI, EDI))
42298                              <1>       ;
42299                              <1>
42300                              <1>       ; 28/02/2017
42301                              <1>       ;cmp  byte [multi_tasking], 0  ; Musti tasking mode ?
42302                              <1>       ;jna  short wswp
42303 0000E727 803D[DA030600]00    <1>       cmp   byte [u.fpsave], 0 ; 28/02/2017
42304 0000E72E 7606                <1>       jna   short wswp
42305 0000E730 DD35[DC030600]      <1>       fnsave [u.fpregs] ; save floating point registers (94 bytes)
42306                              <1> wswp:
42307 0000E736 8B3D[B4030600]      <1>       mov   edi, [u.upage] ; process's user (u) structure page addr
42308 0000E73C B938000000          <1>       mov   ecx, (U_SIZE + 3) / 4
42309 0000E741 BE[5C030600]        <1>       mov   esi, user ; active user (u) structure
42310 0000E746 F3A5                <1>       rep   movsd
42311                              <1>       ;
42312 0000E748 8B35[60030600]      <1>       mov   esi, [u.usp] ; esp (system stack pointer,
42313                              <1>                          ;      points to user registers)
42314 0000E74E 8B0D[5C030600]      <1>       mov   ecx, [u.sp]  ; return address from the system call
42315                              <1>                          ; (for IRET)
42316                              <1>                          ; [u.sp] -> EIP (user)
42317                              <1>                          ; [u.sp+4]-> CS (user)
42318                              <1>                          ; [u.sp+8] -> EFLAGS (user)
42319                              <1>                          ; [u.sp+12] -> ESP (user)
42320                              <1>                          ; [u.sp+16] -> SS (user)
42321 0000E754 29F1                <1>       sub   ecx, esi     ; required space for user registers
```

```
42322 0000E756 83C114              <1>       add    ecx, 20           ; +5 dwords to return from system call
42323                              <1>                              ; (for IRET)
42324 0000E759 C1E902              <1>       shr    ecx, 2
42325 0000E75C F3A5                <1>       rep    movsd
42326 0000E75E C3                  <1>       retn
42327                              <1>
42328                              <1> rswap:  ; < swap in, swap from disk >
42329                              <1>       ; 28/02/2017 (frstor)
42330                              <1>       ; 15/01/2017
42331                              <1>       ; 14/01/2017
42332                              <1>       ; 21/05/2016
42333                              <1>       ; 03/05/2016
42334                              <1>       ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
42335                              <1>       ; 09/05/2015 - 15/09/2015 (Retro UNIX 386 v1)
42336                              <1>       ; 26/05/2013 - 08/03/2014 (Retro UNIX 8086 v1)
42337                              <1>       ; 'rswap' reads a process whose number is in r1,
42338                              <1>       ; from disk into core.
42339                              <1>       ;
42340                              <1>       ; Retro UNIX 386 v1 modification ->
42341                              <1>       ;       User (u) structure content and the user's register content
42342                              <1>       ;     will be restored from process's/user's UPAGE (a page for
42343                              <1>       ;     saving 'u' structure and user registers for task switching).
42344                              <1>       ;     u.usp - points to kernel stack address which contains
42345                              <1>       ;             user's registers while entering system call.
42346                              <1>       ;     u.sp  - points to kernel stack address
42347                              <1>       ;             to return from system call -for IRET-.
42348                              <1>       ;     [u.usp]+32+16 = [u.sp]
42349                              <1>       ;     [u.usp] -> edi, esi, ebp, esp (= [u.usp]+32), ebx,
42350                              <1>       ;             edx, ecx, eax, gs, fs, es, ds, -> [u.sp].
42351                              <1>       ;
42352                              <1>       ; RETRO UNIX 8086 v1 modification ->
42353                              <1>       ;       'swap to disk' is replaced with 'change running segment'
42354                              <1>       ;     according to 8086 cpu (x86 real mode) architecture.
42355                              <1>       ;     pdp-11 was using 64KB uniform memory while IBM PC
42356                              <1>       ;     compatibles was using 1MB segmented memory
42357                              <1>       ;     in 8086/8088 times.
42358                              <1>       ;
42359                              <1>       ; INPUTS ->
42360                              <1>       ;    r1 - process number of process to be read in
42361                              <1>       ;    p.break - negative of word count of process
42362                              <1>       ;    p.dska - disk address of the process
42363                              <1>       ;    u.emt - determines handling of emt's
42364                              <1>       ;    u.ilgins - determines handling of illegal instructions
42365                              <1>       ; OUTPUTS ->
42366                              <1>       ;    8 = (u.ilgins)
42367                              <1>       ;    24 = (u.emt)
42368                              <1>       ;    swp - bit 10 is set to indicate read
42369                              <1>       ;            (bit 15=0 when reading is done)
42370                              <1>       ;    swp+2 - disk block address
42371                              <1>       ;    swp+4 - negative word count
42372                              <1>       ;      ((swp+6 - address of user structure))
42373                              <1>       ;
42374                              <1>       ; RETRO UNIX 8086 v1 input/output:
42375                              <1>       ;
42376                              <1>       ; INPUTS ->
42377                              <1>       ;    AL      - new process number (to be swapped in)
42378                              <1>       ; OUTPUTS ->
42379                              <1>       ;    none
42380                              <1>       ;
42381                              <1>       ;   ((Modified registers: EAX, ECX, ESI, EDI, ESP))
42382                              <1>       ;
42383                              <1>       ; Retro UNIX 386 v1 - modification ! 14/05/2015
42384 0000E75F 89C6                <1>       mov    esi, eax  ; process's user (u) structure page addr
42385 0000E761 B938000000          <1>       mov    ecx, (U_SIZE + 3) / 4
42386 0000E766 BF[5C030600]        <1>       mov    edi, user ; active user (u) structure
42387 0000E76B F3A5                <1>       rep    movsd
42388 0000E76D 58                  <1>       pop    eax    ; 'rswap' return address
42389                              <1>       ;
42390                              <1>       ;cli
42391 0000E76E 8B3D[60030600]      <1>       mov    edi, [u.usp] ; esp (system stack pointer,
42392                              <1>                           ;     points to user registers)
42393 0000E774 89FC                <1>       mov    esp, edi     ; 14/01/2017
42394 0000E776 8B0D[5C030600]      <1>       mov    ecx, [u.sp]  ; return address from the system call
42395                              <1>                           ; (for IRET)
42396                              <1>                           ; [u.sp] -> EIP (user)
42397                              <1>                           ; [u.sp+4]-> CS (user)
42398                              <1>                           ; [u.sp+8] -> EFLAGS (user)
42399                              <1>                           ; [u.sp+12] -> ESP (user)
42400                              <1>                           ; [u.sp+16] -> SS (user)
42401 0000E77C 29F9                <1>       sub    ecx, edi     ; required space for user registers
42402 0000E77E 83C114              <1>       add    ecx, 20           ; +5 dwords to return from system call
42403                              <1>                           ; (for IRET)
42404 0000E781 C1E902              <1>       shr    ecx, 2
42405 0000E784 F3A5                <1>       rep    movsd
42406                              <1>       ;mov   esp, [u.usp] ; 15/09/2015
42407                              <1>       ;sti
42408                              <1>       ; 28/02/2017
42409                              <1>       ;cmp   byte [multi_tasking], 0  ; Musti tasking mode ?
42410                              <1>       ;jna   short rswp_retn
42411 0000E786 803D[DA030600]00    <1>       cmp    byte [u.fpsave], 0
42412 0000E78D 7606                <1>       jna    short rswp_retn
42413 0000E78F DD25[DC030600]      <1>       frstor [u.fpregs] ; restore floating point regs (94 bytes)
42414                              <1> rswp_retn:
42415 0000E795 50                  <1>       push   eax    ; 'rswap' return address
42416 0000E796 C3                  <1>       retn
42417                              <1>
42418                              <1> putlu:
42419                              <1>       ; 20/05/2016
42420                              <1>       ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
42421                              <1>       ; 10/05/2015 - 12/09/2015 (Retro UNIX 386 v1)
42422                              <1>       ; 15/04/2013 - 23/02/2014 (Retro UNIX 8086 v1)
42423                              <1>       ; 'putlu' is called with a process number in r1 and a pointer
```

```
42424                                  <1>       ; to lowest priority Q (runq+4) in r2. A link is created from
42425                                  <1>       ; the last process on the queue to process in r1 by putting
42426                                  <1>       ; the process number in r1 into the last process's link.
42427                                  <1>       ;
42428                                  <1>       ; INPUTS ->
42429                                  <1>       ;    r1 - user process number
42430                                  <1>       ;    r2 - points to lowest priority queue
42431                                  <1>       ;    p.dska - disk address of the process
42432                                  <1>       ;    u.emt - determines handling of emt's
42433                                  <1>       ;    u.ilgins - determines handling of illegal instructions
42434                                  <1>       ; OUTPUTS ->
42435                                  <1>       ;    r3 - process number of last process on the queue upon
42436                                  <1>       ;        entering putlu
42437                                  <1>       ;    p.link-1 + r3 - process number in r1
42438                                  <1>       ;    r2 - points to lowest priority queue
42439                                  <1>       ;
42440                                  <1>       ; ((Modified registers: EDX, EBX))
42441                                  <1>       ;
42442                                  <1>       ; / r1 = user process no.; r2 points to lowest priority queue
42443                                  <1>
42444                                  <1>       ; EBX = r2
42445                                  <1>       ; EAX = r1 (AL=r1b)
42446                                  <1>
42447                                  <1>       ; 20/05/2016
42448                                  <1>       ; AL = process number (1 to 16) // Retro UNIX 8086, 386 v1 //
42449                                  <1>       ;     (max. 16 processes available for current kernel version)
42450                                  <1>       ; EBX = run queue address ; 20/05/2016 (TRDOS 386)
42451                                  <1>            ; which is one of following addresses:
42452                                  <1>            ;  1) 'runq_event' high priority run queue
42453                                  <1>            ;  2) 'runq_normal' normal/regular priority run queue
42454                                  <1>            ;  3) 'runq_background' low priority run queue
42455                                  <1>
42456                                  <1>       ;mov   ebx, runq
42457 0000E797 0FB613                 <1>       movzx        edx, byte [ebx]
42458 0000E79A 43                     <1>       inc    ebx
42459 0000E79B 20D2                   <1>       and    dl, dl
42460                                  <1>            ; tstb (r2)+ / is queue empty?
42461 0000E79D 740A                   <1>            jz     short putlu_1
42462                                  <1>            ; beq 1f / yes, branch
42463 0000E79F 8A13                   <1>       mov    dl, [ebx] ; 12/09/2015
42464                                  <1>            ; movb (r2),r3 / no, save the "last user" process number
42465                                  <1>                       ; / in r3
42466 0000E7A1 8882[9F000600]         <1>            mov    [edx+p.link-1], al
42467                                  <1>            ; movb r1,p.link-1(r3) / put pointer to user on
42468                                  <1>                       ; / "last users" link
42469 0000E7A7 EB03                   <1>       jmp    short putlu_2
42470                                  <1>            ; br 2f /
42471                                  <1> putlu_1: ; 1:
42472 0000E7A9 8843FF                 <1>       mov    [ebx-1], al
42473                                  <1>                   ; movb r1,-1(r2) / user is only user;
42474                                  <1>                       ; / put process no. at beginning and at end
42475                                  <1> putlu_2: ; 2:
42476 0000E7AC 8803                   <1>       mov    [ebx], al
42477                                  <1>                   ; movb r1,(r2) / user process in r1 is now the last entry
42478                                  <1>                       ; / on the queue
42479 0000E7AE 88C2                   <1>       mov    dl, al
42480 0000E7B0 88B2[9F000600]         <1>         mov     [edx+p.link-1], dh ; 0
42481                                  <1>            ; dec r2 / restore r2
42482 0000E7B6 C3                     <1>         retn
42483                                  <1>            ; rts r0
42484                                  <1> sysver:
42485                                  <1>       ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
42486 0000E7B7 C705[64030600]0002-     <1>       mov    dword [u.r0], 200h ; AH = major version, AL = minor version
42487 0000E7BF 0000                   <1>
42488 0000E7C1 E9E9DCFFFF             <1>       jmp    sysret
42489                                  <1>
42490                                  <1> sysreserved1:
42491                                  <1>       ; 19/05/2016 - TRDOS 386 (TRDOS v2.0)
42492                                  <1>       ; // name and content will be changed later //
42493 0000E7C6 C705[64030600]E007-     <1>       mov    dword [u.r0], 2016
42494 0000E7CE 0000                   <1>
42495 0000E7D0 E9DADCFFFF             <1>       jmp    sysret
42496                                  <1>
42497                                  <1> syspri: ; change running priority (of the process)
42498                                  <1>       ; 21/05/2016
42499                                  <1>       ; 20/05/2026 - TRDOS 386 (TRDOS v2.0)
42500                                  <1>       ; INPUT ->
42501                                  <1>       ;     BL = priority level
42502                                  <1>       ;        0 = low running priority (running on background)
42503                                  <1>       ;        1 = normal/regular priority (running as regular)
42504                                  <1>       ;        2 = high/event priority (running for event)
42505                                  <1>       ;        >2 = invalid, it will accepted as 2 (event)
42506                                  <1>       ;        0FFh = get/return current running priority only
42507                                  <1>       ; OUTPUT ->
42508                                  <1>       ;     * if current [u.pri] < 2
42509                                  <1>       ;       if BL input < 0FFh ->
42510                                  <1>       ;          [u.pri] is updated as in BL input (0,1,2)
42511                                  <1>       ;       if BL input = 0FFh -> AL = [u.pri] (current)
42512                                  <1>       ;
42513                                  <1>       ;     * if current [u.pri] = 2
42514                                  <1>       ;       if BL input < 0FFh -> cf = 1 & AL = 2
42515                                  <1>       ;       if BL input = 0FFh -> cf = 0 & AL = 2
42516                                  <1>       ;
42517                                  <1>       ;     NOTE:
42518                                  <1>       ;     If [u.pri] = 2, it can not be changed to 1 or 0;
42519                                  <1>       ;     because, run queue of the running process is unspecified
42520                                  <1>       ;     at this      stage. Process might be started by a timer event
42521                                  <1>       ;     or priority might be changed to high by previous
42522                                  <1>       ;     'syspri' system    call. In both cases, the process is in
42523                                  <1>       ;     'runq_normal' or 'runq_background' queue.
42524                                  <1>       ;     As result of this fact, when the [u.quant] time quantum
42525                                  <1>       ;     of the process is elapsed or 'sysrele' system call is
```

```
42526                                  <1>       ;     instructed by the process, 'tswap' ('tswitch') procedure
42527                                  <1>       ;     will be called (to 'swap' or 'switch' out the procedure)
42528                                  <1>       ;     and it will not call 'putlu' to add the (stopping)
42529                                  <1>       ;     process to relevant run queue when [u.pri] = 2.
42530                                  <1>       ;     (Otherwise, it would be possible to add process to
42531                                  <1>       ;     a run queue while it is already in a run queue, wrongly.)
42532                                  <1>       ;
42533                                  <1>       ;     If [u.pri]< 2, 'tswap/tswitch' procedure will call
42534                                  <1>       ;     'putlu' to add process to relevant run queue
42535                                  <1>       ;     according to [u.pri] value. ('runq_normal' for 1,
42536                                  <1>       ;     'runq_background' for 0).
42537                                  <1>       ;
42538                                  <1>       ;     If BL input >= 2 and < 0FFh while [u.pri] < 2,
42539                                  <1>       ;     process will be added to 'runq_normal' queue and
42540                                  <1>       ;     [u.pri] will be set to 2. (in 'syspri' system call)
42541                                  <1>       ;
42542                                  <1>
42543 0000E7D5 29C0                   <1>       sub   eax, eax ; 0
42544 0000E7D7 A3[C8030600]           <1>       mov   [u.error], eax
42545                                  <1>
42546 0000E7DC A0[A9030600]           <1>       mov   al, [u.pri]
42547 0000E7E1 A3[64030600]           <1>       mov   [u.r0], eax
42548                                  <1>
42549 0000E7E6 FEC3                   <1>       inc   bl
42550 0000E7E8 0F84C1DCFFFF           <1>       jz    sysret ; 0FFh -> 0, get priority level
42551                                  <1>
42552 0000E7EE 3C02                   <1>       cmp   al, 2
42553 0000E7F0 0F8399DCFFFF           <1>       jnb   error ; CF = 1 & AL = 2 (& last error = 0)
42554                                  <1>
42555 0000E7F6 FECB                   <1>       dec   bl
42556 0000E7F8 80FB02                 <1>       cmp   bl, 2
42557 0000E7FB 7602                   <1>       jna   short syspri_1
42558 0000E7FD B302                   <1>       mov   bl, 2
42559                                  <1> syspri_1:
42560 0000E7FF 881D[A9030600]         <1>       mov   [u.pri], bl
42561 0000E805 80FB02                 <1>       cmp   bl, 2
42562 0000E808 0F82A1DCFFFF           <1>         jb      sysret
42563                                  <1>
42564                                  <1>       ; here...
42565                                  <1>       ; Priority of current process has been changed to high
42566                                  <1>       ; ('run for event') but current process will be added to
42567                                  <1>       ; 'run as normal' queue. ('run for event' high priority
42568                                  <1>       ; queue is under control of timer -& RTC- interrupt only!)
42569                                  <1>       ;
42570                                  <1>       ; (Otherwise, process can fall into black hole!
42571                                  <1>       ; e.g. if it is not in waiting list and it has not got
42572                                  <1>       ; a timer event and it is not in a run queue!
42573                                  <1>       ; Because, when [u.pri] is 2, 'tswap/tswitch' will not
42574                                  <1>       ; add the stopping process to a run queue.)
42575                                  <1>
42576 0000E80E A0[B3030600]           <1>       mov   al, [u.uno]
42577 0000E813 BB[54030600]           <1>       mov   ebx, runq_normal ; normal priority !
42578                                  <1>                              ; [u.pri] is set to high
42579                                  <1>                              ; but 'runq_event' queue is set
42580                                  <1>                              ; only by the kernel's timer
42581                                  <1>                              ; event function (timer interrupt).
42582 0000E818 E87AFFFFFF             <1>       call  putlu
42583 0000E81D E98DDCFFFF             <1>       jmp   sysret
42584                                  <1>
42585                                  <1> cpass: ; / get next character from user area of core and put it in AL (r1)
42586                                  <1>       ; 02/05/2016 - TRDOS 386 (TRDOS v2.0)
42587                                  <1>       ; 19/05/2015 - 18/10/2015 (Retro UNIX 386 v1)
42588                                  <1>       ; 14/08/2013 - 20/09/2013 (Retro UNIX 8086 v1)
42589                                  <1>       ; INPUTS ->
42590                                  <1>       ;     [u.base] = virtual address in user area
42591                                  <1>       ;     [u.count] = byte count (max.)
42592                                  <1>       ;     [u.pcount] = byte count in page (0 = reset)
42593                                  <1>       ; OUTPUTS ->
42594                                  <1>       ;     AL = the character which is pointed by [u.base]
42595                                  <1>       ;     zf = 1 -> transfer count has been completed
42596                                  <1>         ;
42597                                  <1>       ; ((Modified registers:  EAX, EDX, ECX))
42598                                  <1>       ;
42599 0000E822 833D[88030600]00       <1>       cmp   dword [u.count], 0  ; have all the characters been transferred
42600                                  <1>                                 ; i.e., u.count, # of chars. left
42601 0000E829 763F                   <1>       jna   short cpass_3    ; to be transferred = 0?) yes, branch
42602 0000E82B FF0D[88030600]         <1>       dec   dword [u.count]         ; no, decrement u.count
42603                                  <1>         ; 19/05/2015
42604                                  <1>       ;(Retro UNIX 386 v1 - translation from user's virtual address
42605                                  <1>       ;                  to physical address
42606 0000E831 66833D[C4030600]00     <1>       cmp   word [u.pcount], 0 ; byte count in page = 0 (initial value)
42607                                  <1>                              ; 1-4095 --> use previous physical base address
42608                                  <1>                              ; in [u.pbase]
42609 0000E839 770E                   <1>       ja    short cpass_1
42610 0000E83B 833D[BC030600]00       <1>       cmp     dword [u.ppgdir], 0  ; is the caller os kernel
42611 0000E842 7427                   <1>         je      short cpass_k       ; (sysexec, '/etc/init') ?  (MainProg)
42612 0000E844 E849FDFFFF             <1>       call  trans_addr_r
42613                                  <1> cpass_1:
42614 0000E849 66FF0D[C4030600]       <1>       dec   word [u.pcount]
42615                                  <1> cpass_2:
42616 0000E850 8B15[C0030600]         <1>       mov   edx, [u.pbase]
42617 0000E856 8A02                   <1>       mov   al, [edx]    ; take the character pointed to
42618                                  <1>                          ; by u.base and put it in r1
42619 0000E858 FF05[8C030600]         <1>       inc   dword [u.nread] ; increment no. of bytes transferred
42620 0000E85E FF05[84030600]         <1>       inc   dword [u.base]  ; increment the buffer address to point to the
42621                                  <1>                            ; next byte
42622 0000E864 FF05[C0030600]         <1>       inc   dword [u.pbase]
42623                                  <1> cpass_3:
42624 0000E86A C3                     <1>       retn
42625                                  <1> cpass_k:
42626                                  <1>       ; 02/07/2015
42627                                  <1>       ; The caller is os kernel
```

```
42628                                  <1>       ; (get sysexec arguments from kernel's memory space)
42629 0000E86B 8B1D[84030600]          <1>       mov    ebx, [u.base]
42630 0000E871 66C705[C4030600]00-     <1>         mov     word [u.pcount], PAGE_SIZE ; 4096
42631 0000E879 10                      <1>
42632 0000E87A 891D[C0030600]          <1>       mov    [u.pbase], ebx
42633 0000E880 EBCE                    <1>       jmp    short cpass_2
42634                                  <1>
42635                                  <1> transfer_to_user_buffer: ; fast transfer
42636                                  <1>       ; 27/05/2016
42637                                  <1>       ; 16/05/2016 - TRDOS 386 (TRDOS v2.0)
42638                                  <1>       ;
42639                                  <1>       ; INPUT ->
42640                                  <1>       ;     ESI = source address in system space
42641                                  <1>       ;     EDI = user's buffer address
42642                                  <1>       ;     ECX = transfer (byte) count
42643                                  <1>       ;     [u.pgdir] = user's page directory
42644                                  <1>       ; OUTPUT ->
42645                                  <1>       ;     ECX = actual transfer count
42646                                  <1>       ;     cf = 1 -> error
42647                                  <1>       ;     [u.count] = remain byte count
42648                                  <1>       ;
42649                                  <1>       ; Modified registers: eax, ecx
42650                                  <1>       ;
42651                                  <1>
42652 0000E882 21C9                    <1>       and    ecx, ecx
42653 0000E884 743B                    <1>       jz     short ttub_4
42654                                  <1>
42655 0000E886 890D[88030600]          <1>       mov    [u.count], ecx
42656                                  <1>
42657 0000E88C 57                      <1>       push   edi
42658 0000E88D 56                      <1>       push   esi
42659 0000E88E 53                      <1>       push   ebx
42660 0000E88F 52                      <1>       push   edx
42661 0000E890 51                      <1>       push   ecx
42662                                  <1>
42663 0000E891 89FB                    <1>       mov    ebx, edi
42664 0000E893 81C300004000            <1>       add    ebx, CORE ; 27/05/2016
42665                                  <1> ttub_1:
42666                                  <1>       ; ebx = virtual (linear) address
42667                                  <1>       ; [u.pgdir] = user's page directory
42668 0000E899 E8F669FFFF              <1>             call   get_physical_addr_x ; get physical address
42669 0000E89E 7222                    <1>       jc     short ttub_5
42670                                  <1>       ; eax = physical address
42671                                  <1>       ; ecx = remain byte count in page (1-4096)
42672 0000E8A0 89C7                    <1>       mov    edi, eax
42673 0000E8A2 A1[88030600]            <1>       mov    eax, [u.count]
42674 0000E8A7 39C1                    <1>       cmp    ecx, eax
42675 0000E8A9 7602                    <1>       jna    short ttub_2
42676 0000E8AB 89C1                    <1>       mov    ecx, eax
42677                                  <1> ttub_2:
42678 0000E8AD 29C8                    <1>       sub    eax, ecx
42679 0000E8AF 01CB                    <1>       add    ebx, ecx
42680 0000E8B1 F3A4                    <1>       rep    movsb
42681 0000E8B3 A3[88030600]            <1>       mov    [u.count], eax
42682 0000E8B8 09C0                    <1>       or     eax, eax
42683 0000E8BA 75DD                    <1>       jnz    short ttub_1
42684                                  <1> ttub_retn:
42685                                  <1> tfub_retn:
42686 0000E8BC 59                      <1>       pop    ecx ; transfer count = actual transfer count
42687                                  <1> ttub_3:
42688 0000E8BD 5A                      <1>       pop    edx
42689 0000E8BE 5B                      <1>       pop    ebx
42690 0000E8BF 5E                      <1>       pop    esi
42691 0000E8C0 5F                      <1>       pop    edi
42692                                  <1> ttub_4:
42693 0000E8C1 C3                      <1>       retn
42694                                  <1> ttub_5:
42695 0000E8C2 59                      <1>       pop    ecx
42696 0000E8C3 2B0D[88030600]          <1>       sub    ecx, [u.count] ; actual transfer count
42697 0000E8C9 F9                      <1>       stc
42698 0000E8CA EBF1                    <1>       jmp    short ttub_3
42699                                  <1>
42700                                  <1> transfer_from_user_buffer: ; fast transfer
42701                                  <1>       ; 27/05/2016
42702                                  <1>       ; 16/05/2016 - TRDOS 386 (TRDOS v2.0)
42703                                  <1>       ;
42704                                  <1>       ; INPUT ->
42705                                  <1>       ;     ESI = user's buffer address
42706                                  <1>       ;     EDI = destination address in system space
42707                                  <1>       ;     ECX = transfer (byte) count
42708                                  <1>       ;     [u.pgdir] = user's page directory
42709                                  <1>       ; OUTPUT ->
42710                                  <1>       ;     ecx = actual transfer count
42711                                  <1>       ;     cf = 1 -> error
42712                                  <1>       ;     [u.count] = remain byte count
42713                                  <1>       ;
42714                                  <1>       ; Modified registers: eax, ecx
42715                                  <1>       ;
42716                                  <1>
42717 0000E8CC 21C9                    <1>       and    ecx, ecx
42718                                  <1>       ;jz    short tfub_4
42719 0000E8CE 74F1                    <1>       jz     short ttub_4
42720                                  <1>
42721 0000E8D0 890D[88030600]          <1>       mov    [u.count], ecx
42722                                  <1>
42723 0000E8D6 57                      <1>       push   edi
42724 0000E8D7 56                      <1>       push   esi
42725 0000E8D8 53                      <1>       push   ebx
42726 0000E8D9 52                      <1>       push   edx
42727 0000E8DA 51                      <1>       push   ecx
42728                                  <1>
42729 0000E8DB 89F3                    <1>       mov    ebx, esi
```

```
42730 0000E8DD 81C300004000        <1>       add    ebx, CORE ; 27/05/2016
42731                              <1> tfub_1:
42732                              <1>       ; ebx = virtual (linear) address
42733                              <1>       ; [u.pgdir] = user's page directory
42734 0000E8E3 E8AC69FFFF         <1>             call   get_physical_addr_x ; get physical address
42735                              <1>       ;jc    short tfub_5
42736 0000E8E8 72D8                <1>       jc     short ttub_5
42737                              <1>       ; eax = physical address
42738                              <1>       ; ecx = remain byte count in page (1-4096)
42739 0000E8EA 89C6                <1>       mov    esi, eax
42740 0000E8EC A1[88030600]        <1>       mov    eax, [u.count]
42741 0000E8F1 39C1                <1>       cmp    ecx, eax
42742 0000E8F3 7602                <1>       jna    short tfub_2
42743 0000E8F5 89C1                <1>       mov    ecx, eax
42744                              <1> tfub_2:
42745 0000E8F7 29C8                <1>       sub    eax, ecx
42746 0000E8F9 01CB                <1>       add    ebx, ecx
42747 0000E8FB F3A4                <1>       rep    movsb
42748 0000E8FD A3[88030600]        <1>       mov    [u.count], eax
42749 0000E902 09C0                <1>       or     eax, eax
42750 0000E904 75DD                <1>       jnz    short tfub_1
42751                              <1>
42752 0000E906 EBB4                <1>       jmp    short tfub_retn
42753                              <1>
42754                              <1> ;tfub_retn:
42755                              <1> ;     pop    ecx ; transfer count = actual transfer count
42756                              <1> ;tfub_3:
42757                              <1> ;     pop    edx
42758                              <1> ;     pop    ebx
42759                              <1> ;     pop    esi
42760                              <1> ;     pop    edi
42761                              <1> ;tfub_4:
42762                              <1> ;     retn
42763                              <1> ;tfub_5:
42764                              <1> ;     pop    ecx
42765                              <1> ;     sub    ecx, [u.count] ; actual transfer count
42766                              <1> ;     stc
42767                              <1> ;     jmp    short tfub_3
42768                              <1>
42769                              <1> sysfff: ; <Find First File>
42770                              <1>       ; 17/10/2016
42771                              <1>       ; 16/10/2016
42772                              <1>       ; 15/10/2016 TRDOS 386 (TRDOS v2.0) feature only !
42773                              <1>       ;           -derived from TRDOS v1.0, INT_21H.ASM-
42774                              <1>       ;            ("loc_INT21h_find_first_file")
42775                              <1>       ; TRDOS 8086 (v1.0)
42776                              <1>         ;   07/08/2011
42777                              <1>         ;   Find First File
42778                              <1>       ;     INPUT:
42779                              <1>         ;       CX= Attributes
42780                              <1>         ;       DS:DX= Pointer to filename
42781                              <1>       ;     MSDOS OUTPUT:
42782                              <1>       ;         DTA: (Default address: PSP offset 80h)
42783                              <1>       ;         Offset  Descrription
42784                              <1>       ;         0      Reserved for use find next file
42785                              <1>       ;         21     Attribute of file found
42786                              <1>       ;         22     Time stamp of file
42787                              <1>       ;         24     Date stamp of file
42788                              <1>       ;         26     File size in bytes
42789                              <1>       ;         30     Filename and extension (zero terminated)
42790                              <1>       ;     If cf = 1:
42791                              <1>       ;         Error Codes: (in AX)
42792                              <1>       ;            2 - File not found
42793                              <1>       ;            18 - No more files
42794                              <1>         ;
42795                              <1>       ; TRDOS 386 (v2.0)
42796                              <1>       ; 15/10/2016
42797                              <1>       ;
42798                              <1>         ; INPUT ->
42799                              <1>         ;      CL = File attributes
42800                              <1>       ;           bit 0 (1) - Read only file (R)
42801                              <1>       ;            bit 1 (1) - Hidden file (H)
42802                              <1>         ;            bit 2 (1) - System file (R)
42803                              <1>       ;            bit 3 (1) - Volume label/name (V)
42804                              <1>         ;            bit 4 (1) - Subdirectory (D)
42805                              <1>       ;           bit 5 (1) - File has been archived (A)
42806                              <1>       ;        CH = 0 -> Return basic parameters (24 bytes)
42807                              <1>       ;        CH > 0 -> Return FindFile structure/table (128 bytes)
42808                              <1>         ;         EBX = Pointer to filename (ASCIIZ) -path-
42809                              <1>       ;        EDX = File parameters buffer address
42810                              <1>       ;           (buffer size = 24 bytes if CH input = 0)
42811                              <1>       ;           (buffer size = 128 bytes if CH input > 0)
42812                              <1>       ;
42813                              <1>       ; OUTPUT ->
42814                              <1>       ;        EAX = 0 if CH input > 0
42815                              <1>       ;        EAX = First cluster number of file if CH input = 0
42816                              <1>       ;        EDX = File parameters table/structure address
42817                              <1>       ;        Basic Parameters:
42818                              <1>       ;           Offset  Description
42819                              <1>       ;           ------ ---------------
42820                              <1>       ;           0      File Attributes
42821                              <1>       ;           1      Ambiguous filename chars are used sign
42822                              <1>       ;                  (0 = filename fits exactly with request)
42823                              <1>       ;                  (>0 = ambiguous filename chars are used)
42824                              <1>       ;           2      Time stamp of file
42825                              <1>       ;           4      Date stamp of file
42826                              <1>       ;           6      File size in bytes
42827                              <1>       ;           10     Short Filename (ASCIIZ, max. 13 bytes)
42828                              <1>       ;           23     Longname Length (1-255) if existing
42829                              <1>       ;
42830                              <1>       ;         cf = 1 -> Error code in AL
42831                              <1>       ;
```

```
42832                                  <1>       ; Modified Registers: EAX (at the return of system call)
42833                                  <1>       ;
42834                                  <1>       ; TR-DOS FindFile (FFF) Structure (128 bytes):
42835                                  <1>       ; 09/10/2011 (DIR.ASM) - 10/02/2016 (trdoskx.s)
42836                                  <1>       ;
42837                                  <1>       ; Offset     Parameter           Size
42838                                  <1>       ; ------     -----------------   --------
42839                                  <1>       ; 0          FindFile_Drv        1 byte
42840                                  <1>       ; 1          FindFile_Directory  65 bytes
42841                                  <1>       ; 66         FindFile_Name       13 bytes
42842                                  <1>       ; 79         FindFile_LongNameEntryLength 1 byte
42843                                  <1>       ;Above 80 bytes form
42844                                  <1>       ;TR-DOS Source/Destination File FullName Format/Structure
42845                                  <1>       ; 80         FindFile_AttributesMask 1 word
42846                                  <1>       ; 82         FindFile_DirEntry   32 bytes (*)
42847                                  <1>       ; 114        FindFile_DirFirstCluster 1 double word
42848                                  <1>       ; 118        FindFile_DirCluster 1 double word
42849                                  <1>       ; 122        FindFile_DirEntryNumber 1 word
42850                                  <1>       ; 124        FindFile_MatchCounter     1 word
42851                                  <1>       ; 126        FindFile_Reserved   1 word
42852                                  <1>       ; (*) MS-DOS, FAT 12-16-32 classic directory entry (32 bytes)
42853                                  <1>
42854                                  <1>       ;mov   [u.namep], ebx
42855                                  <1>       ; 16/10/2016
42856 0000E908 8915[7C5B0100]         <1>       mov    [FFF_UBuffer], edx
42857 0000E90E 66890D[815B0100]       <1>       mov    [FFF_Attrib], cx ; [FFF_RType] = ch
42858                                  <1>                   ; Attributes in CL, return data type in CH
42859 0000E915 89DE                   <1>       mov    esi, ebx
42860                                  <1>       ; file name is forced, change directory as temporary
42861                                  <1>       ;mov   ax, 1
42862                                  <1>       ;mov   [FFF_Valid], ah ; 0 ; reset ; 17/10/2016
42863                                  <1>       ;call  set_working_path
42864 0000E917 E8870C0000             <1>       call   set_working_path_x ; 17/10/2016
42865 0000E91C 731D                   <1>       jnc    short sysfff_0
42866                                  <1>
42867 0000E91E 21C0                   <1>       and    eax, eax  ; 0 -> Bad Path!
42868 0000E920 7505                   <1>       jnz    short sysfff_err
42869                                  <1>
42870                                  <1>       ; eax = 0
42871 0000E922 B80C000000             <1>       mov    eax, ERR_DIR_NOT_FOUND ; Directory not found !
42872                                  <1> sysfff_err:
42873 0000E927 A3[64030600]           <1>       mov    [u.r0], eax
42874 0000E92C A3[C8030600]           <1>       mov    [u.error], eax
42875 0000E931 E8420D0000             <1>         call       reset_working_path
42876 0000E936 E954DBFFFF             <1>       jmp    error
42877                                  <1>
42878                                  <1> sysfff_0:
42879                                  <1>       ;sub   ah, ah ; ah = 0
42880 0000E93B 8A0424                 <1>       mov    al, [esp]
42881 0000E93E 08C0                   <1>       or     al, al
42882 0000E940 7412                   <1>       jz     short sysfff_2
42883 0000E942 B410                   <1>       mov    ah, 10h
42884 0000E944 A808                   <1>       test   al, 08h
42885 0000E946 7503                   <1>       jnz    short sysfff_1
42886 0000E948 80CC08                 <1>       or     ah, 08h
42887                                  <1> sysfff_1:
42888 0000E94B 2410                   <1>       and    al, 10h ; Directory
42889 0000E94D 7405                   <1>       jz     short sysfff_2
42890 0000E94F 80E408                 <1>       and    ah, 08h
42891 0000E952 30C0                   <1>       xor    al, al ; When a directory is searched,
42892                                  <1>                     ; filename will be returned even if
42893                                  <1>                     ; it is not a directory!
42894                                  <1>                     ; Because: (in order to prevent
42895                                  <1>                     ; creating a dir with existing file name)
42896                                  <1>                     ; Dir and file names must not be same!
42897                                  <1>                     ; (return attribute must be checked)
42898                                  <1> sysfff_2:
42899                                  <1>       ; AX = Attributes mask
42900                                  <1>             ; AL = AND mask (result must be equal to AL)
42901                                  <1>             ; AH = Negative AND mask (result must be ZERO)
42902                                  <1>       ; ESI = FindFile_Name address
42903                                  <1>
42904 0000E954 E89696FFFF             <1>       call   find_first_file
42905 0000E959 72CC                   <1>       jc     short sysfff_err ; eax = 2 (File not found !)
42906                                  <1>
42907                                  <1>       ; ESI = Directory Entry (FindFile_DirEntry) Location
42908                                  <1>       ; EDI = Directory Buffer Directory Entry Location
42909                                  <1>       ; EAX = File Size
42910                                  <1>       ;  BL = Attributes of The File/Directory
42911                                  <1>       ;  BH = Long Name Yes/No Status (>0 is YES)
42912                                  <1>       ;  DX > 0 : Ambiguous filename chars are used
42913                                  <1>
42914                                  <1> sysfff_3:
42915                                  <1>       ; 16/10/2016
42916 0000E95B 668B0D[815B0100]       <1>       mov    cx, [FFF_Attrib]
42917                                  <1>       ; Attribs in CL, return data type in CH
42918                                  <1>
42919                                  <1>       ;or    cl, cl
42920                                  <1>       ;jz    short sysfff_4 ; 0 = No filter
42921 0000E962 80F1FF                 <1>       xor    cl, 0FFh
42922 0000E965 20D9                   <1>       and    cl, bl
42923 0000E967 7409                   <1>       jz     short sysfff_4
42924                                  <1>
42925                                  <1>       ;mov   eax, 2 ; 'file not found !' error
42926                                  <1>       ;jmp   short sysfff_err_1
42927                                  <1>
42928                                  <1>       ; 16/10/2016
42929 0000E969 E83097FFFF             <1>       call   find_next_file
42930 0000E96E 72B7                   <1>       jc     short sysfff_err ; eax = 12 (no more files !)
42931 0000E970 EBE9                   <1>       jmp    short sysfff_3
42932                                  <1>
42933                                  <1> sysfff_4:
```

```
42934 0000E972 20ED              <1>       and   ch, ch ; [FFF_RType]
42935 0000E974 7412              <1>       jz    short sysfff_5
42936 0000E976 B980000000        <1>       mov   ecx, 128 ; ; transfer length
42937 0000E97B 880D[805B0100]    <1>       mov   [FFF_Valid], cl
42938                            <1> sysfnf_11:
42939 0000E981 BE[32580100]      <1>       mov   esi, FindFile_Drv
42940 0000E986 EB44              <1>       jmp   short sysfff_6
42941                            <1> sysfff_5:
42942                            <1>       ;mov  esi, FindFile_DirEntry
42943 0000E988 B918000000        <1>       mov   ecx, 24  ; transfer length
42944 0000E98D 880D[805B0100]    <1>       mov   [FFF_Valid], cl
42945                            <1> sysfnf_12:
42946 0000E993 BF[3C600100]      <1>       mov   edi, DTA ; FFF data transfer address
42947                            <1>       ;mov  al, [esi+DirEntry_Attr] ; 11
42948 0000E998 88D8              <1>       mov   al, bl ; File/Dir Attributes
42949 0000E99A 887F17            <1>       mov   [edi+23], bh ; Longname length (0= none)
42950 0000E99D AA                <1>       stosb
42951 0000E99E 88D0              <1>       mov   al, dl ; DL is for '?'
42952 0000E9A0 00F0              <1>       add   al, dh ; DH is for '*'
42953                            <1>       ; AL > 0 if ambiguous file name wildcards are used
42954 0000E9A2 AA                <1>       stosb
42955 0000E9A3 8B4616            <1>       mov   eax, [esi+DirEntry_WrtTime] ; 22
42956 0000E9A6 AB                <1>         stosd     ; DirEntry_WrtTime & DirEntry_WrtDate
42957 0000E9A7 8B461C            <1>         mov  eax, [esi+DirEntry_FileSize] ; 28
42958 0000E9AA AB                <1>         stosd
42959 0000E9AB 668B4614          <1>       mov   ax, [esi+DirEntry_FstClusHI] ; 20
42960 0000E9AF 66C1E010          <1>       shl   ax, 16
42961 0000E9B3 668B461A          <1>       mov   ax, [esi+DirEntry_FstClusLO] ; 26
42962 0000E9B7 A3[64030600]      <1>       mov   [u.r0], eax ; First Cluster
42963                            <1>
42964                            <1>         ;mov esi, FindFile_DirEntry
42965 0000E9BC E8F40C0000        <1>       call  get_file_name
42966                            <1>
42967 0000E9C1 8A0D[805B0100]    <1>       mov   cl, [FFF_Valid]
42968 0000E9C7 BE[3C600100]      <1>             mov   esi, DTA ; FFF data transfer address
42969                            <1> sysfff_6:
42970 0000E9CC 8B3D[7C5B0100]    <1>       mov   edi, [FFF_UBuffer] ; user's buffer address (edx)
42971 0000E9D2 E8ABFEFFFF        <1>       call  transfer_to_user_buffer
42972                            <1>
42973 0000E9D7 890D[64030600]    <1>       mov   [u.r0], ecx ; actual transfer count
42974 0000E9DD E8960C0000        <1>         call       reset_working_path
42975 0000E9E2 E9C8DAFFFF        <1>       jmp   sysret
42976                            <1>
42977                            <1> sysfnf: ; <Find Next File>
42978                            <1>       ; 16/10/2016 TRDOS 386 (TRDOS v2.0) feature only !
42979                            <1>       ;           -derived from TRDOS v1.0, INT_21H.ASM-
42980                            <1>       ;            ("loc_INT21h_find_next_file")
42981                            <1>       ; TRDOS 8086 (v1.0)
42982                            <1>         ;   07/08/2011
42983                            <1>         ;   Find First File
42984                            <1>       ;     INPUT:
42985                            <1>         ;       none
42986                            <1>       ;     MSDOS OUTPUT:
42987                            <1>       ;         DTA: (Default address: PSP offset 80h)
42988                            <1>       ;         Offset  Descrription
42989                            <1>       ;         0     Reserved for use find next file
42990                            <1>       ;         21    Attribute of file found
42991                            <1>       ;         22    Time stamp of file
42992                            <1>       ;         24    Date stamp of file
42993                            <1>       ;         26    File size in bytes
42994                            <1>       ;         30    Filename and extension (zero terminated)
42995                            <1>       ;     If cf = 1:
42996                            <1>       ;         Error Codes: (in AX)
42997                            <1>       ;            18 - No more files
42998                            <1>         ;
42999                            <1>       ; TRDOS 386 (v2.0)
43000                            <1>       ; 16/10/2016
43001                            <1>       ;
43002                            <1>         ; INPUT ->
43003                            <1>             ;        none
43004                            <1>       ; OUTPUT ->
43005                            <1>       ;        EAX = 0 if CH input of 'Find First File' > 0
43006                            <1>       ;        EAX = First cluster number of file
43007                            <1>       ;             if CH input of 'Find First File' = 0
43008                            <1>       ;        EDX = File parameters table/structure address
43009                            <1>       ;
43010                            <1>       ;          cf = 1 -> Error code in AL
43011                            <1>       ;
43012                            <1>       ; Modified Registers: EAX (at the return of system call)
43013                            <1>
43014                            <1>       ;
43015                            <1>       ; Note: If byte [FFF_Valid] = 0
43016                            <1>       ;     'sysfnf' will return with 'no more files' error.
43017                            <1>       ;     If byte [FFF_Valid] = 24
43018                            <1>       ;     'sysfnf' will return with 32 bytes basic parameters
43019                            <1>       ;     at the address which is in EDX.
43020                            <1>       ;     If byte [FFF_Valid] = 128
43021                            <1>       ;     'sysfnf' will return with 128 bytes Find File
43022                            <1>       ;     Structure/Table at the address which is in EDX.
43023                            <1>
43024 0000E9E7 803D[805B0100]00  <1>       cmp   byte [FFF_Valid], 0
43025 0000E9EE 7714              <1>       ja    short stsfnf_0
43026                            <1>       ; 'no more files !' error
43027 0000E9F0 B80C000000        <1>       mov   eax, ERR_NO_MORE_FILES ; 12
43028 0000E9F5 A3[64030600]      <1>       mov   [u.r0], eax
43029 0000E9FA A3[C8030600]      <1>       mov   [u.error], eax
43030 0000E9FF E98BDAFFFF        <1>       jmp   error
43031                            <1> stsfnf_0:
43032                            <1>       ;cmp  byte [FFF_Valid], 128
43033                            <1>       ;je   short stsfnf_1
43034                            <1>       ;cmp  byte [FFF_Valid], 24
43035                            <1>       ;je   short stsfnf_1
```

```
43036                                <1>       ;mov   [FFF_Valid], 24 ; Default
43037                                <1> stsfnf_1:
43038 0000EA04 0FB61D[8E4E0100]      <1>       movzx  ebx, byte [Current_Drv]
43039 0000EA0B 66891D[865B0100]      <1>       mov    [SWP_DRV], bx
43040 0000EA12 8A15[32580100]        <1>       mov    dl, [FindFile_Drv]
43041 0000EA18 38DA                  <1>       cmp    dl, bl
43042 0000EA1A 750B                  <1>       jne    short stsfnf_2
43043 0000EA1C 86FB                  <1>       xchg   bh, bl
43044 0000EA1E BE00010900            <1>       mov    esi, Logical_DOSDisks
43045 0000EA23 01DE                  <1>       add    esi, ebx
43046 0000EA25 EB0D                  <1>       jmp    short sysfnf_3
43047                                <1>
43048                                <1> stsfnf_2:
43049 0000EA27 FE05[875B0100]        <1>       inc    byte [SWP_DRV_chg]
43050                                <1>
43051 0000EA2D E82B82FFFF            <1>       call   change_current_drive
43052 0000EA32 7245                  <1>       jc     short sysfnf_err_1 ; read error !
43053                                <1>                                 ; (do not stop, because
43054                                <1>                                 ; we don't have a
43055                                <1>                                 ; 'no more files'
43056                                <1>                                 ; -file not found- error,
43057                                <1>                                 ; next sysfnf system call
43058                                <1>                                 ; may solve the problem,
43059                                <1>                                 ; after re-placing the disk)
43060                                <1> sysfnf_3:
43061 0000EA34 A1[A8580100]          <1>       mov    eax, [FindFile_DirCluster]
43062 0000EA39 21C0                  <1>       and    eax, eax
43063 0000EA3B 7550                  <1>       jnz    short sysfnf_6
43064                                <1>
43065 0000EA3D 803D[8D4E0100]02      <1>       cmp    byte [Current_FATType], 2
43066 0000EA44 772C                  <1>       ja     short sysfnf_err_0 ; invalid, we neeed to stop !?
43067 0000EA46 803D[8D4E0100]01      <1>       cmp    byte [Current_FATType], 1
43068 0000EA4D 7223                  <1>       jb     short sysfnf_err_0 ; invalid, we neeed to stop !?
43069                                <1>
43070 0000EA4F 3805[B8560100]        <1>       cmp    byte [DirBuff_ValidData], al ; 0
43071 0000EA55 7608                  <1>       jna    short sysfnf_4
43072                                <1>
43073 0000EA57 3B05[BD560100]        <1>       cmp    eax, [DirBuff_Cluster] ; 0 ?
43074 0000EA5D 745E                  <1>       je     short sysfnf_9
43075                                <1>
43076                                <1>       ;cmp   byte [Current_Dir_Level], 0
43077                                <1>         ;ja  short sysfnf_4
43078                                <1>         ;jna short sysfnf_9
43079                                <1>
43080                                <1> sysfnf_4:
43081 0000EA5F FE05[875B0100]        <1>       inc    byte [SWP_DRV_chg]
43082 0000EA65 E80ED0FFFF            <1>       call   load_FAT_root_directory
43083 0000EA6A 7351                  <1>       jnc    short sysfnf_9
43084                                <1>       ; eax = error code (17, 'drv not ready or read error')
43085 0000EA6C EB0B                  <1>       jmp    short sysfnf_err_1 ; read error ! (no FNF stop)
43086                                <1>                                 ; (if you want, try again,
43087                                <1>                                 ;  after re-placing the disk)
43088                                <1> sysfnf_5:
43089 0000EA6E 3C0C                  <1>       cmp    al, 12 ; 'no more files' error
43090 0000EA70 7507                  <1>       jne    short sysfnf_err_1 ; (no FNF stop -sysfnf will try
43091                                <1>                                 ;  to read the directory again,
43092                                <1>                                 ;  if the user calls sysfnf
43093                                <1>                                 ;  just after this error return-)
43094                                <1>       ; (FNF stop -sysfnf will not try
43095                                <1>       ;  to read the directory again-)
43096                                <1>
43097                                <1> sysfnf_err_0:
43098 0000EA72 C605[805B0100]00      <1>       mov    byte [FFF_Valid], 0 ; FNF stop sign
43099                                <1> sysfnf_err_1:
43100 0000EA79 A3[64030600]          <1>       mov    [u.r0], eax
43101 0000EA7E A3[C8030600]          <1>       mov    [u.error], eax
43102 0000EA83 E8F00B0000            <1>       call   reset_working_path
43103 0000EA88 E902DAFFFF            <1>       jmp    error
43104                                <1>
43105                                <1> sysfnf_6:
43106 0000EA8D 803D[B8560100]00      <1>       cmp    byte [DirBuff_ValidData], 0
43107 0000EA94 7608                  <1>       jna    short sysfnf_7
43108                                <1>
43109 0000EA96 3B05[BD560100]        <1>       cmp    eax, [DirBuff_Cluster]
43110 0000EA9C 741F                  <1>       je     short sysfnf_9
43111                                <1>
43112                                <1> sysfnf_7:
43113 0000EA9E FE05[875B0100]        <1>       inc    byte [SWP_DRV_chg]
43114 0000EAA4 803D[8D4E0100]01      <1>       cmp    byte [Current_FATType], 1
43115 0000EAAB 7309                  <1>       jnb    short sysfnf_8
43116                                <1>
43117                                <1>       ; Singlix (TRFS) File System
43118                                <1>       ; (access via compatibility buffer)
43119 0000EAAD E88ED0FFFF            <1>       call   load_FS_sub_directory
43120 0000EAB2 7309                  <1>       jnc    short sysfnf_9
43121                                <1>
43122 0000EAB4 EBC3                  <1>       jmp    short sysfnf_err_1 ; read error (no FNF stop)
43123                                <1>
43124                                <1> sysfnf_8:
43125 0000EAB6 E848D0FFFF            <1>       call   load_FAT_sub_directory
43126 0000EABB 72BC                  <1>       jc     short sysfnf_err_1 ; read error (no FNF stop)
43127                                <1>
43128                                <1> sysfnf_9:
43129 0000EABD E8DC95FFFF            <1>       call   find_next_file
43130 0000EAC2 72AA                  <1>       jc     short sysfnf_5
43131                                <1>
43132 0000EAC4 A0[815B0100]          <1>       mov    al, [FFF_Attrib]
43133                                <1>       ;or    al, al
43134                                <1>       ;jz    short sysfnf_10 ; 0 = No filter
43135 0000EAC9 34FF                  <1>       xor    al, 0FFh
43136 0000EACB 20D8                  <1>       and    al, bl
43137 0000EACD 75EE                  <1>       jnz    short sysfnf_9 ; search for next file until
```

```
43138                                   <1>                             ; an error return from
43139                                   <1>                             ; find_next_file procedure
43140                                   <1> sysfnf_10:
43141 0000EACF 0FB60D[805B0100]         <1>         movzx      ecx, byte [FFF_Valid]
43142 0000EAD6 80F980                   <1>       cmp    cl, 128 ; complete FindFile structure/table
43143 0000EAD9 0F84A2FEFFFF             <1>       je     sysfnf_11
43144                                   <1>       ;cmp   cl, 24  ; basic parameters
43145                                   <1>       ;je    sysfnf_12
43146 0000EADF E9AFFEFFFF               <1>       jmp    sysfnf_12
43147                                   <1>
43148                                   <1> writei:
43149                                   <1>       ; 26/10/2016
43150                                   <1>       ; 25/10/2016
43151                                   <1>       ; 23/10/2016
43152                                   <1>       ; 22/10/2016
43153                                   <1>       ; 19/10/2016 - TRDOS 386 (TRDOS v2.0)
43154                                   <1>       ; 19/05/2015 - 20/05/2015 (Retro UNIX 386 v1)
43155                                   <1>       ; 12/03/2013 - 31/07/2013 (Retro UNIX 8086 v1)
43156                                   <1>       ;
43157                                   <1>       ; Write data to file with first cluster number in EAX
43158                                   <1>       ;
43159                                   <1>       ; INPUTS ->
43160                                   <1>       ;    EAX - First cluster number of the file
43161                                   <1>       ;    EBX - File number (Open file index number)
43162                                   <1>       ;    u.count - byte count to be written
43163                                   <1>       ;    u.base - points to user buffer
43164                                   <1>       ;    u.fofp - points to dword with current file offset
43165                                   <1>       ;    i.size - file size
43166                                   <1>       ;    cdev - logical dos drive number of the file
43167                                   <1>       ; OUTPUTS ->
43168                                   <1>       ;    u.count - cleared
43169                                   <1>       ;    u.nread - accumulates total bytes passed back
43170                                   <1>       ;    i.size - new file size (if file byte offset overs file size)
43171                                   <1>       ;    u.fofp - points to u.off (with new offset value)
43172                                   <1>       ;
43173                                   <1>       ; (Retro UNIX Prototype : 11/11/2012 - 18/11/2012, UNIXCOPY.ASM)
43174                                   <1>       ; ((Modified registers: eax, edx, ebx, ecx, esi, edi, ebp))
43175                                   <1>
43176 0000EAE4 31C9                     <1>       xor    ecx, ecx
43177 0000EAE6 890D[8C030600]           <1>       mov    [u.nread], ecx  ; 0
43178 0000EAEC 66890D[C4030600]         <1>       mov    [u.pcount], cx ; 19/05/2015
43179 0000EAF3 390D[88030600]           <1>       cmp    [u.count], ecx
43180 0000EAF9 7701                     <1>       ja     short writei_1
43181 0000EAFB C3                       <1>       retn
43182                                   <1> writei_1:
43183 0000EAFC 881D[405B0100]           <1>       mov    [writei.ofn], bl ; Open file number
43184 0000EB02 880D[7B5B0100]           <1>       mov    [setfmod], cl ; 0 ; reset 'update lm date&time' sign
43185                                   <1> dskw_0:
43186                                   <1>       ; 26/10/2016
43187                                   <1>       ; 22/10/2016, 23/10/2016, 25/10/2016
43188                                   <1>       ; 19/10/2016 - TRDOS 386 (TRDOS v2.0)
43189                                   <1>       ; 31/05/2015 - 25/07/2015 (Retro UNIX 386 v1)
43190                                   <1>       ; 26/04/2013 - 20/09/2013 (Retro UNIX 8086 v1)
43191                                   <1>       ;
43192                                   <1>       ; 01/08/2013 (mkdir_w check)
43193 0000EB08 E8D7000000               <1>       call   mget_w
43194                                   <1>       ; eax = sector/block number
43195                                   <1>
43196 0000EB0D 8B1D[74030600]           <1>       mov     ebx, [u.fofp]
43197 0000EB13 8B13                     <1>       mov    edx, [ebx]
43198 0000EB15 81E2FF010000             <1>       and    edx, 1FFh  ; / test the lower 9 bits of the file offset
43199 0000EB1B 750C                     <1>       jnz    short dskw_1 ; / if its non-zero, branch
43200                                   <1>                         ; if zero, file offset = 0,
43201                                   <1>                                 ; / 512, 1024,...(i.e., start of new block)
43202 0000EB1D 813D[88030600]0002-      <1>       cmp    dword [u.count], 512
43203 0000EB25 0000                     <1>
43204                                   <1>                           ; / if zero, is there enough data to fill
43205                                   <1>                           ; / an entire block? (i.e., no. of
43206 0000EB27 7337                     <1>       jnb    short dskw_2 ; / bytes to be written greater than 512.?
43207                                   <1>                         ; / Yes, branch. Don't have to read block
43208                                   <1> dskw_1: ; in as no past info. is to be saved
43209                                   <1>       ; (the entire block will be overwritten).
43210                                   <1>       ; 23/10/2016
43211                                   <1>
43212 0000EB29 BB[94070600]             <1>       mov    ebx, writei_buffer
43213                                   <1>       ; esi = logical dos drive description table address
43214                                   <1>       ; eax = sector number
43215                                   <1>       ; ebx = buffer address (in kernel's memory space)
43216                                   <1>       ; ecx = sector count
43217 0000EB2E B901000000               <1>       mov    ecx, 1
43218 0000EB33 E866060000               <1>       call   disk_read
43219                                   <1>       ;call  dskrd  ; / no, must retain old info..
43220                                   <1>                             ; / Hence, read block 'r1' into an I/O buffer
43221 0000EB38 7326                     <1>       jnc    short dskw_2
43222                                   <1>
43223                                   <1>       ; disk read error
43224 0000EB3A B811000000               <1>       mov    eax, 17 ; drive not ready or READ ERROR !
43225                                   <1> dskw_err: ; jump from disk write error
43226 0000EB3F A3[64030600]             <1>       mov    [u.r0], eax
43227 0000EB44 A3[C8030600]             <1>       mov    [u.error], eax
43228                                   <1>
43229 0000EB49 803D[7B5B0100]00         <1>       cmp    byte [setfmod], 0
43230 0000EB50 0F8639D9FFFF             <1>       jna    error
43231                                   <1>
43232 0000EB56 E8AF030000               <1>       call   update_file_lmdt ; update last modif. date&time of the file
43233                                   <1>       ;mov   byte [setfmod], 0
43234                                   <1>
43235 0000EB5B E92FD9FFFF               <1>       jmp    error
43236                                   <1>
43237                                   <1> dskw_2: ; 3:
43238                                   <1>       ; 23/10/2016
43239 0000EB60 C605[1C5B0100]01         <1>       mov    byte [writei.valid], 1 ; writei buffer contains valid data
```

```
43240 0000EB67 56                  <1>       push   esi ; logical dos drive description table address
43241                              <1>       ; EAX (r1) = block/sector number
43242                              <1>       ;call  wslot
43243                              <1>             ; jsr r0,wslot / set write and inhibit bits in I/O queue,
43244                              <1>                       ; / proc. status=0, r5 points to 1st word of data
43245 0000EB68 803D[C6030600]00    <1>       cmp    byte [u.kcall], 0
43246 0000EB6F 770F                <1>       ja     short dskw_4 ; zf=0 -> the caller is 'mkdir'
43247                              <1>       ;
43248 0000EB71 66833D[C4030600]00  <1>       cmp    word [u.pcount], 0
43249 0000EB79 7705                <1>       ja     short dskw_4
43250                              <1> dskw_3:
43251                              <1>       ; [u.base] = virtual address to transfer (as source address)
43252 0000EB7B E812FAFFFF          <1>       call   trans_addr_r ; translate virtual address to physical (r)
43253                              <1> dskw_4:
43254 0000EB80 BB[94070600]        <1>       mov    ebx, writei_buffer
43255                              <1>       ; EBX (r5) = system (I/O) buffer address
43256 0000EB85 E874FAFFFF          <1>       call   sioreg
43257                              <1>       ; ESI = file (user data) offset
43258                              <1>       ; EDI = sector (I/O) buffer offset
43259                              <1>       ; ECX = byte count
43260                              <1>       ;
43261 0000EB8A F3A4                <1>       rep    movsb
43262                              <1>       ; 25/07/2015
43263                              <1>       ; eax = remain bytes in buffer
43264                              <1>         ;       (check if remain bytes in the buffer > [u.pcount])
43265 0000EB8C 09C0                <1>       or     eax, eax
43266 0000EB8E 75EB                <1>       jnz    short dskw_3 ; (page end before system buffer end!)
43267                              <1>
43268                              <1>       ; 23/10/2016
43269 0000EB90 B101                <1>       mov    cl, 1
43270 0000EB92 5E                  <1>       pop    esi
43271 0000EB93 A1[205B0100]        <1>       mov    eax, [writei.sector]
43272                              <1>       ; esi = logical dos drive description table address
43273                              <1>       ; eax = sector number
43274                              <1>       ; ebx = writei buffer address
43275                              <1>       ; ecx = sector count
43276 0000EB98 E8F2050000          <1>       call   disk_write ; / yes, write the block
43277 0000EB9D 7307                <1>       jnc    short dskw_5
43278                              <1>
43279 0000EB9F B812000000          <1>       mov    eax, 18 ; drive not ready or WRITE ERROR !
43280 0000EBA4 EB99                <1>       jmp    short dskw_err
43281                              <1>
43282                              <1> dskw_5:
43283                              <1>       ; 26/10/2016
43284 0000EBA6 0FB61D[405B0100]    <1>       movzx  ebx, byte [writei.ofn] ; open file number
43285 0000EBAD C0E302              <1>       shl    bl, 2 ; *4
43286 0000EBB0 8B83[105F0100]      <1>       mov    eax, [ebx+OF_POINTER]
43287 0000EBB6 3B83[385F0100]      <1>       cmp    eax, [ebx+OF_SIZE]
43288 0000EBBC 7606                <1>       jna    short dskw_6
43289 0000EBBE 8983[385F0100]      <1>       mov    [ebx+OF_SIZE], eax
43290                              <1> dskw_6:
43291                              <1>       ;shr   bl, 2
43292 0000EBC4 833D[88030600]00    <1>         cmp     dword [u.count], 0 ; / any more data to write?
43293 0000EBCB 760A                <1>       jna    short dskw_7
43294 0000EBCD A1[305B0100]        <1>       mov    eax, [writei.fclust]
43295 0000EBD2 E931FFFFFF          <1>       jmp    dskw_0 ; / yes, branch
43296                              <1> dskw_7:
43297                              <1>       ; update last modif. date&time of the file
43298                              <1>       ; (also updates file size as OF_SIZE)
43299 0000EBD7 E82E030000          <1>       call   update_file_lmdt
43300                              <1>       ;mov   byte [setfmod], 0
43301                              <1>
43302                              <1>       ; 03/08/2013
43303 0000EBDC C605[C6030600]00    <1>       mov    byte [u.kcall], 0
43304                              <1>       ; 23/10/2016
43305                              <1>       ;mov   eax, [writei.fclust]
43306 0000EBE3 C3                  <1>       retn
43307                              <1>
43308                              <1> mget_w:
43309                              <1>       ; 02/11/2016
43310                              <1>       ; 01/11/2016
43311                              <1>       ; 23/10/2016, 31/10/2016
43312                              <1>       ; 22/10/2016 - TRDOS 386 (TRDOS v2.0)
43313                              <1>       ; 03/06/2015 (Retro UNIX 386 v1, 'mget', u.5s)
43314                              <1>       ; 22/03/2013 - 31/07/2013 (Retro UNIX 8086 v1)
43315                              <1>       ;
43316                              <1>       ; Get existing or (allocate) a new disk block for file
43317                              <1>       ;
43318                              <1>       ; INPUTS ->
43319                              <1>       ;    [u.fofp] = file offset pointer
43320                              <1>       ;    [i.size] = file size
43321                              <1>       ;    [u.count] = byte count
43322                              <1>       ;    EAX = First cluster
43323                              <1>       ;    [cdev] = Logical dos drive number
43324                              <1>       ;    [writei.ofn] = File Number
43325                              <1>       ;                (Open file index, 0 based)
43326                              <1>       ;    ([u.off] = file offset)
43327                              <1>       ; OUTPUTS ->
43328                              <1>       ;    EAX = logical sector number
43329                              <1>       ;    ESI = Logical Dos Drive Description Table address
43330                              <1>       ;
43331                              <1>       ; Modified registers: EDX, EBX, ECX, ESI, EDI, EBP
43332                              <1>
43333 0000EBE4 8B35[74030600]      <1>         mov     esi, [u.fofp]
43334 0000EBEA 8B2E                <1>       mov    ebp, [esi] ; u.off (or EBX*4+OF_POINTER)
43335                              <1>
43336 0000EBEC 29C9                <1>       sub    ecx, ecx
43337 0000EBEE 8A2D[46030600]      <1>       mov    ch, [cdev]
43338                              <1>
43339 0000EBF4 BE00010900          <1>       mov    esi, Logical_DOSDisks
43340 0000EBF9 01CE                <1>       add    esi, ecx
43341                              <1>
```

```
43342                                  <1>       ; 31/10/2016
43343 0000EBFB 89C3                    <1>       mov    ebx, eax ; First Cluster or FDT address
43344                                  <1>
43345 0000EBFD 807E0300                <1>       cmp    byte [esi+LD_FATType], 0
43346 0000EC01 0F86DD010000            <1>       jna    mget_w_14 ; Singlix FS
43347                                  <1>
43348 0000EC07 0FB74611                <1>       movzx  eax, word [esi+LD_BPB+BytesPerSec]
43349 0000EC0B 0FB65613                <1>       movzx  edx, byte [esi+LD_BPB+SecPerClust]
43350 0000EC0F 8815[1E5B0100]          <1>       mov    [writei.spc], dl  ; sectors per cluster
43351 0000EC15 F7E2                    <1>       mul    edx
43352                                  <1>       ; edx = 0
43353                                  <1>       ; eax = bytes per cluster (<= 65536)
43354                                  <1>
43355                                  <1>       ; 02/11/2016
43356 0000EC17 89C1                    <1>       mov    ecx, eax
43357 0000EC19 48                      <1>       dec    eax
43358 0000EC1A 66A3[245B0100]          <1>       mov    [writei.bpc], ax
43359                                  <1>
43360 0000EC20 89E8                    <1>       mov    eax, ebp
43361 0000EC22 0305[88030600]          <1>       add    eax, [u.count] ; next file position
43362 0000EC28 3B05[55040600]          <1>       cmp    eax, [i.size] ; <= file size ?
43363 0000EC2E 0F86FC000000            <1>       jna    mget_w_4 ; no
43364                                  <1>
43365 0000EC34 F7F1                    <1>       div    ecx
43366 0000EC36 A3[2C5B0100]            <1>       mov    [writei.c_index], eax ; cluster index
43367                                  <1>       ; edx = byte offset in cluster (<= 65535)
43368                                  <1>       ;mov   [writei.offset], dx
43369                                  <1>       ;shr   dx, 9 ; / 512
43370                                  <1>       ;mov   [writei.s_index], dl ; sector index in cluster (0 to spc -1)
43371                                  <1>
43372 0000EC3B 29D2                    <1>       sub    edx, edx ; 01/11/2016
43373 0000EC3D 8915[205B0100]          <1>       mov    [writei.sector], edx ; 0
43374 0000EC43 668915[265B0100]        <1>       mov    [writei.offset], dx  ; byte offset in cluster
43375 0000EC4A 8815[1F5B0100]          <1>       mov    [writei.s_index], dl ; sector index in cluster (0 to spc -1)
43376                                  <1>
43377 0000EC50 89D8                    <1>       mov    eax, ebx ; First Cluster
43378                                  <1>
43379                                  <1>       ; is this the 1st mget_w or a next mget_w call ? (by 'writei')
43380 0000EC52 3815[1C5B0100]          <1>       cmp    byte [writei.valid], dl ; 0
43381 0000EC58 7624                    <1>       jna    short mget_w_0
43382                                  <1>
43383 0000EC5A 8815[1C5B0100]          <1>       mov    byte [writei.valid], dl ; 0 ; reset ('writei' will set it)
43384                                  <1>
43385 0000EC60 3B05[305B0100]          <1>       cmp    eax, [writei.fclust]
43386 0000EC66 7516                    <1>       jne    short mget_w_0
43387                                  <1>
43388 0000EC68 8A0D[46030600]          <1>       mov    cl, [cdev]
43389 0000EC6E 3A0D[1D5B0100]          <1>       cmp    cl, [writei.drv]
43390 0000EC74 7508                    <1>       jne    short mget_w_0
43391                                  <1>       ; [writei.l_clust] & [writei.l_index] are valid,
43392                                  <1>       ;  we don't need to get last cluster & last cluster index
43393 0000EC76 8B0D[3C5B0100]          <1>       mov    ecx, [writei.l_index]
43394 0000EC7C EB64                    <1>       jmp    short mget_w_2
43395                                  <1> mget_w_0:
43396 0000EC7E A3[305B0100]            <1>       mov    [writei.fclust], eax ; first cluster
43397                                  <1>       ; edx = 0
43398 0000EC83 A3[285B0100]            <1>       mov    [writei.cluster], eax ; first cluster ; 01/11/2016
43399 0000EC88 8915[345B0100]          <1>       mov    [writei.fs_index], edx ; 0 ; curret cluster index
43400                                  <1>
43401                                  <1>       ; FAT file system (FAT12, FAT16, FAT32)
43402 0000EC8E E885D4FFFF              <1>       call   get_last_cluster
43403 0000EC93 0F822B010000            <1>       jc     mget_w_err ; eax = error code
43404                                  <1>
43405 0000EC99 A3[385B0100]            <1>       mov    [writei.lclust], eax ; last cluster
43406                                  <1>
43407 0000EC9E 8B0D[5C590100]          <1>       mov    ecx, [glc_index] ; last cluster index
43408 0000ECA4 890D[3C5B0100]          <1>       mov    [writei.l_index], ecx
43409                                  <1>
43410 0000ECAA A0[405B0100]            <1>       mov    al, [writei.ofn]
43411 0000ECAF FEC0                    <1>       inc    al
43412 0000ECB1 A2[7B5B0100]            <1>       mov    [setfmod], al ; update lm date&time sign
43413                                  <1>
43414                                  <1> mget_w_1:
43415 0000ECB6 3B0D[2C5B0100]          <1>       cmp    ecx, [writei.c_index]  ; last cluster index
43416 0000ECBC 7324                    <1>       jnb    short mget_w_2 ; 01/11/2016
43417                                  <1>
43418 0000ECBE A1[385B0100]            <1>       mov    eax, [writei.lclust]
43419                                  <1>       ; EAX = Last cluster
43420 0000ECC3 E85ED5FFFF              <1>       call   add_new_cluster
43421 0000ECC8 0F82F6000000            <1>       jc     mget_w_err ; eax = error code
43422                                  <1>       ; edx = 0
43423 0000ECCE A3[385B0100]            <1>       mov    [writei.lclust], eax ; (new) last cluster
43424 0000ECD3 8B0D[3C5B0100]          <1>       mov    ecx, [writei.l_index]
43425 0000ECD9 41                      <1>       inc    ecx ; add 1 to last cluster index
43426 0000ECDA 890D[3C5B0100]          <1>       mov    [writei.l_index], ecx ; current last cluster index
43427                                  <1>
43428 0000ECE0 EBD4                    <1>       jmp    short mget_w_1
43429                                  <1>
43430                                  <1> mget_w_2:
43431 0000ECE2 89E9                    <1>       mov    ecx, ebp
43432 0000ECE4 030D[88030600]          <1>       add    ecx, [u.count]
43433 0000ECEA 890D[55040600]          <1>       mov    [i.size], ecx ; save new file size
43434                                  <1>       ;sub   edx, edx ; 0
43435                                  <1>
43436 0000ECF0 A0[46030600]            <1>       mov    al, [cdev]
43437 0000ECF5 A2[1D5B0100]            <1>       mov    [writei.drv], al ; physical drive number
43438                                  <1>       ; edx = 0
43439 0000ECFA 89E8                    <1>       mov    eax, ebp ; file offset
43440 0000ECFC 0FB70D[245B0100]        <1>       movzx  ecx, word [writei.bpc] ; bytes per cluster - 1
43441 0000ED03 41                      <1>       inc    ecx ; bytes per cluster
43442 0000ED04 F7F1                    <1>       div    ecx
43443                                  <1>       ; edx = byte offset in cluster (<= 65535)
```

```
43444                                  <1>       ; eax = cluster index
43445 0000ED06 A3[2C5B0100]            <1>       mov    [writei.c_index], eax
43446 0000ED0B 668915[265B0100]        <1>       mov    [writei.offset], dx
43447 0000ED12 66C1EA09                <1>       shr    dx, 9 ; / 512
43448 0000ED16 8815[1F5B0100]          <1>       mov    [writei.s_index], dl ; sector index in cluster (0 to spc -1)
43449                                  <1>
43450                                  <1> mget_w_3:
43451 0000ED1C 3B05[3C5B0100]          <1>       cmp    eax, [writei.l_index] ; last cluster index
43452 0000ED22 752A                    <1>       jne    short mget_w_5
43453                                  <1>
43454 0000ED24 A3[345B0100]            <1>       mov    [writei.fs_index], eax ; cluster index (for next check)
43455 0000ED29 A1[385B0100]            <1>       mov    eax, [writei.lclust] ; last cluster
43456 0000ED2E EB60                    <1>       jmp    short mget_w_10
43457                                  <1>
43458                                  <1> mget_w_4: ; 02/11/2016
43459                                  <1>       ; eax = next file position
43460 0000ED30 2B05[88030600]          <1>       sub    eax, [u.count] ; current file position
43461                                  <1>       ; edx = 0
43462                                  <1>       ; ecx = bytes per cluster
43463 0000ED36 F7F1                    <1>       div    ecx
43464 0000ED38 A3[2C5B0100]            <1>       mov    [writei.c_index], eax ; cluster index
43465 0000ED3D 668915[265B0100]        <1>       mov    [writei.offset], dx
43466 0000ED44 66C1EA09                <1>       shr    dx, 9 ; / 512
43467 0000ED48 8815[1F5B0100]          <1>       mov    [writei.s_index], dl ; sector index in cluster (0 to spc -1)
43468                                  <1>
43469                                  <1> mget_w_5:
43470 0000ED4E 21C0                    <1>       and    eax, eax ; 0 = First Cluster's index number
43471 0000ED50 750C                    <1>       jnz    short mget_w_6
43472                                  <1>
43473 0000ED52 A3[345B0100]            <1>       mov    [writei.fs_index], eax ; cluster index (for next check)
43474 0000ED57 A1[305B0100]            <1>       mov    eax, [writei.fclust] ; first cluster
43475 0000ED5C EB32                    <1>       jmp    short mget_w_10
43476                                  <1>
43477                                  <1> mget_w_6:
43478 0000ED5E 3B05[345B0100]          <1>       cmp    eax, [writei.fs_index] ; current cluster index (>0)
43479 0000ED64 7507                    <1>       jne    short mget_w_7
43480 0000ED66 A1[285B0100]            <1>       mov    eax, [writei.cluster] ; current cluster
43481 0000ED6B EB3A                    <1>       jmp    short mget_w_11
43482                                  <1>
43483                                  <1> mget_w_7:
43484 0000ED6D 89C1                    <1>       mov    ecx, eax
43485 0000ED6F 2B0D[345B0100]          <1>       sub    ecx, [writei.fs_index]
43486 0000ED75 730D                    <1>       jnc    short mget_w_8
43487                                  <1>       ; get cluster by index from the first cluster
43488 0000ED77 A1[305B0100]            <1>       mov    eax, [writei.fclust]
43489 0000ED7C 8B0D[2C5B0100]          <1>       mov    ecx, [writei.c_index]
43490 0000ED82 EB05                    <1>       jmp    short mget_w_9
43491                                  <1>
43492                                  <1> mget_w_8:
43493 0000ED84 A1[285B0100]            <1>       mov    eax, [writei.cluster] ; beginning cluster
43494                                  <1>       ; ecx = cluster sequence number after the beginning cluster
43495                                  <1>       ; sub  edx, edx ; 0
43496                                  <1>
43497                                  <1> mget_w_9:
43498                                  <1>       ; EAX = Beginning cluster
43499                                  <1>       ; EDX = Sector index in disk/file section
43500                                  <1>       ;     (Only for SINGLIX file system!)
43501                                  <1>       ; ECX = Cluster sequence number after the beginning cluster
43502                                  <1>       ; ESI = Logical DOS Drive Description Table address
43503 0000ED89 E89ED5FFFF              <1>       call   get_cluster_by_index
43504 0000ED8E 7234                    <1>       jc     short mget_w_err ; error code in EAX
43505                                  <1>       ; EAX = Cluster number
43506                                  <1> mget_w_10:
43507 0000ED90 A3[285B0100]            <1>       mov    [writei.cluster], eax ; FDT number for Singlix File System
43508                                  <1>
43509 0000ED95 807E0300                <1>       cmp    byte [esi+LD_FATType], 0
43510 0000ED99 7638                    <1>       jna    short mget_w_13
43511                                  <1>       ; 01/11/2016
43512 0000ED9B 8B15[2C5B0100]          <1>       mov    edx, [writei.c_index]
43513 0000EDA1 8915[345B0100]          <1>       mov    [writei.fs_index], edx
43514                                  <1> mget_w_11:
43515 0000EDA7 83E802                  <1>       sub    eax, 2
43516 0000EDAA 0FB615[1E5B0100]        <1>       movzx  edx, byte [writei.spc]
43517 0000EDB1 F7E2                    <1>       mul    edx
43518                                  <1>
43519 0000EDB3 034668                  <1>       add    eax, [esi+LD_DATABegin]
43520 0000EDB6 8A15[1F5B0100]          <1>       mov    dl, [writei.s_index]
43521 0000EDBC 01D0                    <1>       add    eax, edx
43522                                  <1> mget_w_12:
43523 0000EDBE A3[205B0100]            <1>       mov    [writei.sector], eax
43524                                  <1>       ;; buffer validation must be done in writei
43525                                  <1>       ;;mov  byte [writei.valid], 1
43526 0000EDC3 C3                      <1>       retn
43527                                  <1>
43528                                  <1> mget_w_err:
43529 0000EDC4 A3[C8030600]            <1>       mov    [u.error], eax
43530 0000EDC9 A3[64030600]            <1>       mov    [u.r0], eax
43531 0000EDCE E9BCD6FFFF              <1>       jmp    error
43532                                  <1>
43533                                  <1> mget_w_13:
43534                                  <1>       ; EAX = FDT number (Current Section)
43535                                  <1>       ; EDX = Sector index from the first section (0,1,2,3,4...)
43536 0000EDD3 2B15[345B0100]          <1>       sub    edx, [writei.fs_index]
43537                                  <1>       ; EDX = Sector index from current section
43538 0000EDD9 8915[345B0100]          <1>       mov    [writei.fs_index], edx
43539 0000EDDF 40                      <1>       inc    eax ; the first data sector in FS disk section
43540 0000EDE0 01D0                    <1>       add    eax, edx
43541 0000EDE2 EBDA                    <1>       jmp    short mget_w_12
43542                                  <1>
43543                                  <1> mget_w_14:
43544 0000EDE4 8A4E12                  <1>       mov    cl, [esi+LD_FS_BytesPerSec+1]
43545 0000EDE7 D0E9                    <1>       shr    cl, 1 ;  ; 1 for 512 bytes, 4 for 2048 bytes
```

```
43546 0000EDE9 880D[1E5B0100]      <1>       mov    [writei.spc], cl  ; sectors per cluster
43547                              <1>       ; NOTE: writei bytes per sector value is always 512 !
43548 0000EDEF 66C705[245B0100]00- <1>       mov    word [writei.bpc], 512
43549 0000EDF7 02                  <1>
43550                              <1>
43551 0000EDF8 89E9                <1>       mov    ecx, ebp
43552 0000EDFA 030D[88030600]      <1>       add    ecx, [u.count] ; next file position
43553 0000EE00 3B0D[55040600]      <1>       cmp    ecx, [i.size] ; <= file size ?
43554 0000EE06 0F86C8000000        <1>       jna    mget_w_19 ; no
43555                              <1>
43556 0000EE0C 29D2                <1>       sub    edx, edx ; 0
43557 0000EE0E 8915[205B0100]      <1>       mov    [writei.sector], edx ; 0
43558 0000EE14 668915[265B0100]    <1>       mov    [writei.offset], dx  ; byte offset in cluster
43559 0000EE1B 8815[1F5B0100]      <1>       mov    [writei.s_index], dl ; sector index in cluster (0 to spc -1)
43560                              <1>
43561 0000EE21 C1E909              <1>       shr    ecx, 9 ; 1 cluster = 512 bytes
43562 0000EE24 890D[2C5B0100]      <1>       mov    [writei.c_index], ecx ; section/cluster index
43563                              <1>
43564 0000EE2A 89D8                <1>       mov    eax, ebx ; FDT number (First FDT address)
43565                              <1>
43566                              <1>       ; is this the 1st mget_w or a next mget_w call ? (by 'writei')
43567 0000EE2C 3815[1C5B0100]      <1>       cmp    byte [writei.valid], dl ; 0
43568 0000EE32 7624                <1>       jna    short mget_w_15
43569                              <1>
43570 0000EE34 8815[1C5B0100]      <1>       mov    byte [writei.valid], dl ; 0 ; reset ('writei' will set it)
43571                              <1>
43572 0000EE3A 3B05[305B0100]      <1>       cmp    eax, [writei.fclust]
43573 0000EE40 7516                <1>       jne    short mget_w_15
43574                              <1>
43575 0000EE42 8A0D[46030600]      <1>       mov    cl, [cdev]
43576 0000EE48 3A0D[1D5B0100]      <1>       cmp    cl, [writei.drv]
43577 0000EE4E 7508                <1>       jne    short mget_w_15
43578                              <1>       ; [writei.l_clust] & [writei.l_index] are valid,
43579                              <1>       ;  we don't need to get last cluster & last cluster index
43580 0000EE50 8B0D[3C5B0100]      <1>       mov    ecx, [writei.l_index]
43581 0000EE56 EB49                <1>       jmp    short mget_w_17
43582                              <1> mget_w_15:
43583 0000EE58 A3[305B0100]        <1>       mov    [writei.fclust], eax ; first section (FDT number)
43584                              <1>       ; edx = 0
43585 0000EE5D 8915[285B0100]      <1>       mov    [writei.cluster], edx ; 0 ; current section
43586 0000EE63 8915[345B0100]      <1>       mov    [writei.fs_index], edx ; 0 ; curret section index
43587                              <1>
43588                              <1>       ; eax = FDT number (section 0 header address)
43589 0000EE69 E8E8D4FFFF          <1>       call   get_last_section
43590 0000EE6E 0F8250FFFFFF        <1>       jc     mget_w_err ; eax = error code
43591                              <1>
43592 0000EE74 8915[345B0100]      <1>       mov    [writei.fs_index], edx ; sector index in last section
43593                              <1>
43594 0000EE7A A3[385B0100]        <1>       mov    [writei.lclust], eax ; last section address
43595                              <1>
43596 0000EE7F 8B0D[5C590100]      <1>       mov    ecx, [glc_index] ; last section index
43597 0000EE85 890D[3C5B0100]      <1>       mov    [writei.l_index], ecx
43598                              <1>
43599 0000EE8B A0[405B0100]        <1>       mov    al, [writei.ofn]
43600 0000EE90 FEC0                <1>       inc    al
43601 0000EE92 A2[7B5B0100]        <1>       mov    [setfmod], al ; update lm date&time sign
43602                              <1>
43603                              <1> mget_w_16:
43604                              <1>       ; edx = (existing) last section (sector) index
43605 0000EE97 8B0D[2C5B0100]      <1>       mov    ecx, [writei.c_index] ; final section (sector) index
43606 0000EE9D 29D1                <1>       sub    ecx, edx
43607 0000EE9F 7633                <1>       jna    short mget_w_19
43608                              <1>        ; ecx = sector count
43609                              <1> mget_w_17:
43610 0000EEA1 A1[385B0100]        <1>       mov    eax, [writei.lclust]
43611                              <1>       ; ESI = Logical dos drv desc. table address
43612                              <1>         ; EAX = Last section
43613                              <1>         ; (ECX = 0 for directory)
43614                              <1>         ; ECX = sector count (except FDT)
43615 0000EEA6 E86ECAFFFF          <1>       call   add_new_fs_section
43616 0000EEAB 7312                <1>       jnc    short mget_w_18
43617                              <1>
43618                              <1>       ; If error number = 27h (insufficient disk space)
43619                              <1>       ; it is needed to check free consequent sectors
43620                              <1>       ; (1 data sector at least and +1 section header sector)
43621                              <1>
43622 0000EEAD 83F827              <1>       cmp    eax, 27h
43623 0000EEB0 0F850EFFFFFF        <1>       jne    mget_w_err ; eax = error code
43624                              <1>
43625                              <1>       ; ecx = count of free consequent sectors
43626                              <1>       ; ecx must be > 1 (1 data + 1 header sector)
43627 0000EEB6 49                  <1>       dec    ecx
43628 0000EEB7 0F8407FFFFFF        <1>       jz     mget_w_err
43629 0000EEBD EBE2                <1>       jmp    short mget_w_17
43630                              <1>
43631                              <1> mget_w_18:
43632 0000EEBF A3[385B0100]        <1>       mov    [writei.lclust], eax ; (new) last section
43633                              <1>       ; ecx = sector count (except section header)
43634 0000EEC4 8B15[3C5B0100]      <1>       mov    edx, [writei.l_index]
43635 0000EECA 01CA                <1>       add    edx, ecx ; add sector count to index
43636 0000EECC 8915[3C5B0100]      <1>       mov    [writei.l_index], edx
43637 0000EED2 EBC3                <1>       jmp    short mget_w_16
43638                              <1>
43639                              <1> mget_w_19:
43640 0000EED4 89E9                <1>       mov    ecx, ebp
43641 0000EED6 030D[88030600]      <1>       add    ecx, [u.count]
43642 0000EEDC 890D[55040600]      <1>       mov    [i.size], ecx ; save new file size
43643                              <1>       ;sub   edx, edx ; 0
43644                              <1>
43645 0000EEE2 A0[46030600]        <1>       mov    al, [cdev]
43646 0000EEE7 A2[1D5B0100]        <1>       mov    [writei.drv], al ; physical drive number
43647                              <1>       ; edx = 0
```

```
43648 0000EEEC 89E8              <1>       mov    eax, ebp ; file offset
43649 0000EEEE 89C2              <1>       mov    edx, eax
43650                            <1>       ; 1 cluster = 512 bytes (for Singlix FS)
43651 0000EEF0 C1E809            <1>       shr    eax, 9  ; / 512
43652 0000EEF3 81E2FF010000      <1>       and    edx, 1FFh
43653                            <1>       ; edx = byte offset in cluster/sector (<= 511)
43654                            <1>       ; eax = section (sector/cluster) index
43655 0000EEF9 A3[2C5B0100]      <1>       mov    [writei.c_index], eax
43656 0000EEFE 668915[265B0100]  <1>       mov    [writei.offset], dx
43657                            <1>       ;mov   byte [writei.s_index], 0 ; sector index in cluster
43658 0000EF05 E912FEFFFF        <1>       jmp    mget_w_3
43659                            <1>
43660                            <1> update_file_lmdt: ; & update file size
43661                            <1>       ; 26/10/2016
43662                            <1>       ; 24/10/2016
43663                            <1>       ; 23/10/2016
43664                            <1>       ; 22/10/2016 - TRDOS 386 (TRDOS v2.0)
43665                            <1>       ;
43666                            <1>       ; Update last modification date&time of file
43667                            <1>       ; (call from syswrite -> writei)
43668                            <1>       ; ((also updates file size)) // 26/10/2016
43669                            <1>       ;
43670                            <1>       ; INPUT:
43671                            <1>       ;      byte [setfmod] = open file number
43672                            <1>       ; OUTPUT:
43673                            <1>       ;      cf = 0 -> success !
43674                            <1>       ;      cf = 1 -> lmdt update has been failed!
43675                            <1>       ;
43676                            <1>       ; Modified registers: eax, ebx, ecx, edx, esi, edi
43677                            <1>       ;
43678                            <1>
43679                            <1>       ;cmp   byte [setfmod], 0
43680                            <1>       ;jna   short uflmdt_2 ; nothing to do
43681                            <1>
43682 0000EF0A 31C0              <1>       xor    eax, eax
43683                            <1>
43684 0000EF0C 0FB61D[7B5B0100]  <1>       movzx  ebx, byte [setfmod]
43685 0000EF13 FECB              <1>       dec    bl ; open file index number (0 based)
43686                            <1>
43687 0000EF15 8AA3[E85E0100]    <1>       mov    ah, [ebx+OF_DRIVE]
43688 0000EF1B BE00010900        <1>       mov    esi, Logical_DOSDisks
43689 0000EF20 01C6              <1>       add    esi, eax
43690 0000EF22 C0E302            <1>       shl    bl, 2 ; *4
43691 0000EF25 8B8B[C05E0100]    <1>       mov    ecx, [ebx+OF_FCLUSTER] ; first cluster
43692 0000EF2B 8B93[885F0100]    <1>       mov    edx, [ebx+OF_DIRCLUSTER] ; dir cluster
43693                            <1>
43694 0000EF31 D0EB              <1>       shr    bl, 1 ; /2
43695 0000EF33 0FB7BB[28600100]  <1>       movzx  edi, word [ebx+OF_DIRENTRY]
43696                            <1>
43697 0000EF3A 803D[B8560100]01  <1>       cmp    byte [DirBuff_ValidData], 1
43698 0000EF41 726E              <1>       jb     short uflmdt_4
43699                            <1>
43700 0000EF43 A0[B6560100]      <1>       mov    al, [DirBuff_DRV]
43701 0000EF48 2C41              <1>       sub    al, 'A'
43702 0000EF4A 38E0              <1>       cmp    al, ah
43703 0000EF4C 7563              <1>       jne    short uflmdt_4 ; different drive
43704 0000EF4E 8A4603            <1>       mov    al, [esi+LD_FATType]
43705 0000EF51 3A05[B7560100]    <1>       cmp    al, [DirBuff_FATType]
43706 0000EF57 755B              <1>       jne    short uflmdt_5 ; different FS type
43707 0000EF59 3B15[BD560100]    <1>       cmp    edx, [DirBuff_Cluster]
43708 0000EF5F 7553              <1>       jne    short uflmdt_5 ; different cluster
43709                            <1>
43710                            <1> uflmdt_1:
43711                            <1>       ; Directory buffer is ready here!
43712                            <1>       ; OF_FCLUSTER must be compared/verified
43713 0000EF61 BE00000800        <1>       mov    esi, Directory_Buffer
43714 0000EF66 66C1E705          <1>       shl    di, 5 ; dir entry index * 32
43715 0000EF6A 01FE              <1>       add    esi, edi ; offset
43716                            <1>       ;
43717 0000EF6C F6460B18          <1>       test   byte [esi+DirEntry_Attr], 18h ; Vol & Dir
43718 0000EF70 750F              <1>       jnz    short uflmdt_2 ; not a valid file !
43719 0000EF72 668B4614          <1>       mov    ax, [esi+DirEntry_FstClusHI]
43720 0000EF76 C1E010            <1>       shl    eax, 16
43721 0000EF79 668B461A          <1>       mov    ax, [esi+DirEntry_FstClusLO]
43722 0000EF7D 39C8              <1>       cmp    eax, ecx ; same first cluster ?
43723 0000EF7F 7407              <1>       je     short uflmdt_3 ; yes, it is OK !!!
43724                            <1>
43725                            <1> uflmdt_2:
43726                            <1>       ; save directory buffer if has modified/changed sign
43727                            <1>       ; (It is good to save dir buff even if the searched
43728                            <1>       ; directory entry is not found !?)
43729 0000EF81 E8E4B6FFFF        <1>       call   save_directory_buffer
43730 0000EF86 F9                <1>       stc    ; update failed
43731 0000EF87 C3                <1>       retn
43732                            <1>
43733                            <1> uflmdt_3:
43734                            <1>       ; Update directory entry
43735                            <1>       ; 26/10/2016
43736 0000EF88 D0E3              <1>       shl    bl, 1 ; *2
43737 0000EF8A 8B83[385F0100]    <1>       mov    eax, [ebx+OF_SIZE] ; file size
43738 0000EF90 89461C            <1>       mov    [esi+DirEntry_FileSize], eax
43739                            <1>       ;
43740 0000EF93 E834B6FFFF        <1>       call   convert_current_date_time
43741                            <1>       ; OUTPUT -> DX = Date in dos dir entry format
43742                            <1>         ;       AX = Time in dos dir entry format
43743 0000EF98 66894616          <1>       mov    [esi+DirEntry_WrtTime], ax
43744 0000EF9C 66895618          <1>       mov    [esi+DirEntry_WrtDate], dx
43745 0000EFA0 66895612          <1>       mov    [esi+DirEntry_LastAccDate], dx
43746 0000EFA4 C605[B8560100]02  <1>       mov    byte [DirBuff_ValidData], 2
43747 0000EFAB E8BAB6FFFF        <1>       call   save_directory_buffer
43748 0000EFB0 C3                <1>       retn
43749                            <1>
```

```
                            <1> uflmdt_4:
                            <1>       ; Directory buffer sector read&write
                            <1>       ; 23/10/2016
                            <1>       ;
0000EFB1 8A4603             <1>       mov    al, [esi+LD_FATType]
                            <1> uflmdt_5:
0000EFB4 BB[9C090600]       <1>       mov    ebx, rw_buffer ; Common r/w sector buffer addr
                            <1>
0000EFB9 20C0               <1>       and    al, al ; 0 = Singlix FS
0000EFBB 0F8492000000       <1>       jz     uflmdt_11
                            <1>
0000EFC1 21D2               <1>       and    edx, edx
0000EFC3 7521               <1>       jnz    short uflmdt_9
                            <1>
0000EFC5 3C02               <1>       cmp    al, 2  ; 3 = FAT32
0000EFC7 771A               <1>       ja     short uflmdt_8
                            <1>
0000EFC9 89F8               <1>       mov    eax, edi ; directory entry index number
0000EFCB 66C1E804           <1>       shr    ax, 4 ; 16 entries per sector
0000EFCF 034664             <1>       add    eax, [esi+LD_ROOTBegin]
                            <1>       ; eax = root directory sector
                            <1> uflmdt_6:
0000EFD2 50                 <1>       push   eax ; * ; disk sector address
0000EFD3 51                 <1>       push   ecx ; first cluster
0000EFD4 B901000000         <1>       mov    ecx, 1
                            <1>       ; ecx = sector count
0000EFD9 E8C0010000         <1>       call   disk_read
0000EFDE 59                 <1>       pop    ecx
0000EFDF 731A               <1>       jnc    short uflmdt_10
0000EFE1 58                 <1>       pop    eax ; *
                            <1> uflmdt_7:
0000EFE2 C3                 <1>       retn
                            <1>
                            <1> uflmdt_8:
0000EFE3 8B5632             <1>       mov    edx, [esi+LD_BPB+FAT32_RootFClust]
                            <1> uflmdt_9:
0000EFE6 83FA02             <1>       cmp    edx, 2
0000EFE9 72F7               <1>       jb     short uflmdt_7 ; invalid, nothing to do
                            <1>
0000EFEB 83EA02             <1>       sub    edx, 2
0000EFEE 89D0               <1>       mov    eax, edx
0000EFF0 0FB65613           <1>       movzx  edx, byte [esi+LD_BPB+SecPerClust]
0000EFF4 F7E2               <1>       mul    edx
0000EFF6 034668             <1>       add    eax, [esi+LD_DATABegin]
                            <1>       ; eax = sub directory (data) sector
0000EFF9 EBD7               <1>       jmp    short uflmdt_6
                            <1>
                            <1> uflmdt_10:
                            <1>       ; Directory sector buffer is ready here!
                            <1>       ; OF_FCLUSTER must be compared/verified
                            <1>       ; edi = dir entry index number (<= 2047)
0000EFFB 6683E70F           <1>       and    di, 0Fh ; 16 entries per sector
0000EFFF 66C1E705           <1>       shl    di, 5 ; dir entry index * 32
0000F003 81C7[9C090600]     <1>       add    edi, rw_buffer
                            <1>       ;
0000F009 F6470B18           <1>       test   byte [edi+DirEntry_Attr], 18h ; Vol & Dir
0000F00D 0F856EFFFFFF       <1>       jnz    uflmdt_2 ; not a valid file !
0000F013 668B5714           <1>       mov    dx, [edi+DirEntry_FstClusHI]
0000F017 C1E210             <1>       shl    edx, 16
0000F01A 668B571A           <1>       mov    dx, [edi+DirEntry_FstClusLO]
0000F01E 39CA               <1>       cmp    edx, ecx ; same first cluster ?
0000F020 0F855BFFFFFF       <1>       jne    uflmdt_2 ; no !?
                            <1>
                            <1>       ; Update directory entry
0000F026 E8A1B5FFFF         <1>       call   convert_current_date_time
                            <1>       ; OUTPUT -> DX = Date in dos dir entry format
                            <1>         ;       AX = Time in dos dir entry format
0000F02B 66894716           <1>       mov    [edi+DirEntry_WrtTime], ax
0000F02F 66895718           <1>       mov    [edi+DirEntry_WrtDate], dx
0000F033 66895712           <1>       mov    [edi+DirEntry_LastAccDate], dx
                            <1>
0000F037 58                 <1>       pop    eax ; *
                            <1>
0000F038 BB[9C090600]       <1>       mov    ebx, rw_buffer ; Common r/w sector buffer addr
0000F03D B901000000         <1>       mov    ecx, 1
                            <1>       ; esi = logical dos description table address
                            <1>       ; eax = disk sector number/address (LBA)
                            <1>       ; ecx = sector count
                            <1>       ; ebx = buffer address
0000F042 E848010000         <1>       call   disk_write
0000F047 0F8234FFFFFF       <1>       jc     uflmdt_2
                            <1>
                            <1>       ; save directory buffer if has modified/changed sign
0000F04D E818B6FFFF         <1>       call   save_directory_buffer
0000F052 C3                 <1>       retn
                            <1>
                            <1> uflmdt_11:
                            <1>       ; 24/10/2016
                            <1>       ; Update last modification date & time of a file
                            <1>       ; on a disk with Singlix File System.
                            <1>       ;
                            <1>       ; (Method: Read the FDT -File Description Table-
                            <1>       ; sector of the file and update the lmdt data fields,
                            <1>       ; then write FDT sector to the disk.
                            <1>       ; /// It is easy but there is compatibility buffer
                            <1>       ; method also for changing directory entry data and
                            <1>       ; also there are some programming issues for Singlix
                            <1>       ; file system (TRFS), which are not completed yet!)
                            <1>       ;
                            <1>       ; Not ready yet ! (24/10/2016)
                            <1>       ; /// Temporary code for error return ! ///
0000F053 31C0               <1>       xor    eax, eax
```

```
43852 0000F055 F9                <1>       stc
43853 0000F056 C3                <1>       retn
43854                            <1>
43855                            <1> sysalloc:
43856                            <1>       ; 15/05/2017
43857                            <1>       ; 04/03/2017
43858                            <1>       ; 20/02/2017
43859                            <1>       ; 19/02/2017 - TRDOS 386 (TRDOS v2.0)
43860                            <1>       ; (TRDOS 386 feature only!)
43861                            <1>       ;
43862                            <1>       ; Allocate Contiguous Memory Block/Pages (for user)
43863                            <1>       ; (System call for DMA Buffer allocation etc.)
43864                            <1>       ;
43865                            <1>       ; INPUT ->
43866                            <1>       ;     EBX = Virtual address (for user)
43867                            <1>       ;          (Physical memory block/aperture
43868                            <1>       ;          will be mapped to this virtual address)
43869                            <1>       ;     ECX = Byte Count
43870                            <1>       ;          (will be rounded up to page border)
43871                            <1>       ;     If ECX = 0
43872                            <1>       ;         System call will return with an error (cf=1)
43873                            <1>       ;         but ECX will contain maximum size of
43874                            <1>       ;         available memory aperture and physical
43875                            <1>       ;         (beginning) address of that aperture
43876                            <1>       ;         (which have maximum size) will be in EAX.
43877                            <1>       ;     EDX = Upper limit of the requested physical memory
43878                            <1>       ;           block/pages.
43879                            <1>       ;          (The last byte address of the memory aperture
43880                            <1>       ;           must not be equal to or above this limit.)
43881                            <1>       ;     If EDX = 0
43882                            <1>       ;        there is NOLIMIT !
43883                            <1>       ;     If EDX = 0FFFFFFFFh (-1)
43884                            <1>       ;        ESI = Lower Limit !
43885                            <1>       ;            (Beginning of the block must not be 'less'
43886                            <1>       ;            than this.) (Must be equal to or above...)
43887                            <1>       ;        EDI = Upper Limit !
43888                            <1>       ;            (End of the block must be !less! than this)
43889                            <1>       ;            (The last byte addr of the memory aperture
43890                            <1>       ;            must not be equal to or above this limit.)
43891                            <1>       ;
43892                            <1>       ; OUTPUT ->
43893                            <1>       ;     If CF = 0
43894                            <1>       ;     EAX = Physical address of the allocated memory block
43895                            <1>       ;     ECX = Allocated bytes (as rounded up to page borders)
43896                            <1>       ;     EBX = Virtual address (as rounded up)
43897                            <1>       ;     IF CF = 1
43898                            <1>       ;         Requested (size of) Memory block could not be
43899                            <1>       ;         allocated to the user!
43900                            <1>       ;     IF CF = 1 & EAX = 0 (Insufficient memory error!)
43901                            <1>       ;        ECX = Total number of free bytes
43902                            <1>       ;              (not size of available contiguous bytes!)
43903                            <1>       ;     If CF = 1 & EAX > 0
43904                            <1>       ;        there is not a memory aperture with requested size
43905                            <1>       ;        but total free mem is not less than requested size.
43906                            <1>       ;        EAX = Physical addr of available memory aperture
43907                            <1>       ;             with max size
43908                            <1>       ;              (but it doesn't fit to the conditions!)
43909                            <1>       ;        ECX = Size of available memory aperture in bytes.
43910                            <1>       ;     If CF = 1 -> EAX = 0FFFFFFFFh
43911                            <1>       ;        Conditions/Parameters are wrong !
43912                            <1>       ;        ECX is same with input value.
43913                            <1>       ;
43914                            <1>       ; Note:      Previously allocated pages will be deallocated if
43915                            <1>       ;       new allocation conditions are met.
43916                            <1>       ;
43917                            <1>       ; Note: u.break control may be included in future versions
43918                            <1>       ;
43919                            <1>
43920 0000F057 31C0             <1>       xor    eax, eax ; 0
43921 0000F059 09D2             <1>       or     edx, edx
43922 0000F05B 7418             <1>       jz     short sysalloc_2 ; NOLIMIT
43923 0000F05D 48               <1>       dec    eax ; 0FFFFFFFFh  (-1)
43924 0000F05E 39D1             <1>       cmp    ecx, edx
43925 0000F060 7742             <1>       ja     short sysalloc_wrong
43926 0000F062 39C2             <1>       cmp    edx, eax ; 0FFFFFFFFh (-1)
43927 0000F064 750E             <1>       jne    short sysalloc_1
43928 0000F066 89FA             <1>       mov    edx, edi ; upper limit
43929 0000F068 29F7             <1>       sub    edi, esi
43930 0000F06A 7638             <1>       jna    short sysalloc_wrong
43931 0000F06C 39CF             <1>       cmp    edi, ecx ; byte count
43932 0000F06E 7234             <1>       jb     short sysalloc_wrong
43933 0000F070 89F0             <1>       mov    eax, esi ; lower limit, beginning address
43934 0000F072 EB01             <1>       jmp    short sysalloc_2
43935                            <1> sysalloc_1:
43936 0000F074 40               <1>       inc    eax ; 0
43937                            <1> sysalloc_2:
43938                            <1>       ; EAX = Beginning address (physical)
43939                            <1>       ; EAX = 0 -> Allocate mem block from the 1st proper aperture
43940                            <1>       ; ECX = Number of bytes to be allocated
43941 0000F075 E8A963FFFF       <1>       call   allocate_memory_block
43942 0000F07A 721F             <1>       jc     short sysalloc_err
43943                            <1>       ; EAX = Beginning (physical) addr of the allocated mem block
43944                            <1>       ; ECX = Num of allocated bytes (rounded up to page borders)
43945 0000F07C 50               <1>       push   eax ; * ; 04/03/2017
43946 0000F07D 09D2             <1>       or     edx, edx ; is there a limit ?
43947 0000F07F 7514             <1>       jnz    short sysalloc_4 ; yes
43948                            <1> sysalloc_3:
43949                            <1>       ; Here, requested contiquous memory pages have been allocated
43950                            <1>       ; on Memory Allocation Table but user's page directory
43951                            <1>       ; and page tables have not been updated yet!
43952 0000F081 51               <1>       push   ecx ; **
43953                            <1>       ; ebx = virtual address (will be rounded up to page border)
```

```
43954                                  <1>       ; ecx = number of bytes to be deallocated
43955                                  <1>       ;     will be adjusted to ebx+ecx round down - ebx round up
43956 0000F082 E8F766FFFF              <1>       call  deallocate_user_pages
43957 0000F087 7325                    <1>       jnc   short sysalloc_5 ; EAX = Deallocated memory bytes
43958 0000F089 59                      <1>       pop   ecx ; **
43959 0000F08A 58                      <1>       pop   eax ; *
43960                                  <1>       ; error
43961                                  <1>       ; restore Memory Allocation Table Content
43962 0000F08B E8A065FFFF              <1>       call  deallocate_memory_block
43963 0000F090 31C0                    <1>       xor   eax, eax ; 0
43964 0000F092 48                      <1>       dec   eax ; 0FFFFFFFFh ; 15/05/2017
43965 0000F093 EB0F                    <1>       jmp   short sysalloc_wrong
43966                                  <1> sysalloc_4:
43967                                  <1>       ; check upper limit (edx) overlap
43968                                  <1>       ; ecx = byte count
43969                                  <1>       ; eax = beginning address (physical)
43970 0000F095 01C8                    <1>       add   eax, ecx
43971 0000F097 39D0                    <1>       cmp   eax, edx
43972 0000F099 72E6                    <1>       jb    short sysalloc_3
43973                                  <1> sysalloc_err:
43974 0000F09B 8B2D[60030600]          <1>       mov   ebp, [u.usp]  ; ebp points to user's registers
43975 0000F0A1 894D18                  <1>       mov   [ebp+24], ecx ; return to user with ecx value
43976                                  <1> sysalloc_wrong:
43977                                  <1>       ; eax = 0FFFFFFFFh
43978 0000F0A4 A3[64030600]            <1>       mov   [u.r0], eax
43979 0000F0A9 E9E1D3FFFF              <1>       jmp   error
43980                                  <1> sysalloc_5:
43981 0000F0AE 8B2D[60030600]          <1>       mov   ebp, [u.usp]  ; ebp points to user's registers
43982 0000F0B4 894518                  <1>       mov   [ebp+24], eax ; return to user with ecx value
43983 0000F0B7 895D10                  <1>       mov   [ebp+16], ebx ; new value of ebx (rounded up)
43984 0000F0BA 89C1                    <1>       mov   ecx, eax ; byte count (from 'deallocate_user_pages')
43985 0000F0BC 5A                      <1>       pop   edx ; ** ; discard (another) byte count
43986 0000F0BD 58                      <1>       pop   eax ; *
43987 0000F0BE A3[64030600]            <1>       mov   [u.r0], eax ; physical address
43988                                  <1>       ;
43989                                  <1>       ; Write newly allocated contiguous (physical) pages
43990                                  <1>       ; on page dir and page tables of current user/process
43991                                  <1>       ; as PRESENT, USER, WRITABLE
43992                                  <1>       ; (then clear allocated pages)
43993 0000F0C3 E8AB67FFFF              <1>       call  allocate_user_pages
43994 0000F0C8 0F83E1D3FFFF            <1>       jnc   sysret ; OK! return to process with success...
43995                                  <1>       ;
43996                                  <1>       ; unexpected error ! insufficient memory !? conflict !?
43997                                  <1>       ; (!!?there is not a free page for a new page table?!!)
43998                                  <1>       ; We need to terminate process with error message !!!
43999                                  <1>       ;
44000 0000F0CE 8B2D[60030600]          <1>       mov   ebp, [u.usp]  ; ebp points to user's registers
44001 0000F0D4 8B4D18                  <1>       mov   ecx, [ebp+24] ; byte count
44002 0000F0D7 A1[64030600]            <1>       mov   eax, [u.r0]   ; physical address (of the block)
44003                                  <1>       ;
44004                                  <1>       ; restore Memory Allocation Table Content
44005 0000F0DC E84F65FFFF              <1>       call  deallocate_memory_block
44006                                  <1>       ;
44007 0000F0E1 803D[C25E0000]03        <1>       cmp   byte [CRT_MODE], 3 ; 80x25 text mode?
44008 0000F0E8 7407                    <1>       je    short sysalloc_6 ; yes
44009                                  <1>       ; Current mode is VGA (or CGA graphics) mode,
44010                                  <1>       ; We need to return to text mode for displaying
44011                                  <1>       ; error message just before 'sysexit'.
44012 0000F0EA B003                    <1>       mov   al, 3
44013 0000F0EC E87424FFFF              <1>       call  _set_mode
44014                                  <1> sysalloc_6:
44015 0000F0F1 BE[F9050100]            <1>       mov   esi, beep_Insufficient_Memory ; error message
44016 0000F0F6 E86272FFFF              <1>       call  print_msg ; print/display the message
44017 0000F0FB B801000000              <1>       mov   eax, 1 ; ax=1 is needed for 'sysexit' procedure
44018 0000F100 E931D5FFFF              <1>       jmp   sysexit ; and terminate the process !
44019                                  <1>
44020                                  <1> sysdalloc:
44021                                  <1>       ; 19/02/2017 - TRDOS 386 (TRDOS v2.0)
44022                                  <1>       ; (TRDOS 386 feature only!)
44023                                  <1>       ;
44024                                  <1>       ; Deallocate Memory Block/Pages (for user)
44025                                  <1>       ; (Complementary call for sysalloc.)
44026                                  <1>       ;
44027                                  <1>       ; INPUT ->
44028                                  <1>       ;     EBX = Virtual address (for user)
44029                                  <1>       ;           (will be rounded up to page border)
44030                                  <1>       ;     ECX = Byte Count
44031                                  <1>       ;          (will be adjusted to page borders)
44032                                  <1>       ;     If ICX = 0
44033                                  <1>       ;        nothing to do
44034                                  <1>       ;     If EBX + ECX > User's ESP
44035                                  <1>       ;        nothing to do
44036                                  <1>       ;
44037                                  <1>       ; Note: u.break control may be included in future versions
44038                                  <1>       ;
44039                                  <1>       ; OUTPUT ->
44040                                  <1>       ;     If CF = 0
44041                                  <1>       ;        EAX = Deallocated memory bytes
44042                                  <1>       ;        EBX = Virtual address (as rounded up)
44043                                  <1>       ;     IF CF = 1
44044                                  <1>       ;        EAX = 0
44045                                  <1>       ;
44046                                  <1>       ; Note:      Main purpose of this call is to deallocate/release
44047                                  <1>       ;     previously allocated (physically) contiguous memory
44048                                  <1>       ;     pages but beginning (virtual) address may not be
44049                                  <1>       ;     followed by physically contiguous pages. So, this
44050                                  <1>       ;     system call will deallocate user's virtually
44051                                  <1>       ;     contiguous memory pages. Also, there is not any
44052                                  <1>       ;     objections to use this system call without sysalloc
44053                                  <1>       ;     system call; only possible objection is to lost data
44054                                  <1>       ;     within user's memory space, if the beginning address
44055                                  <1>       ;     and size is not proper.
```

```
44056                                  <1>       ;
44057                                  <1>       ; Note: Empty page tables will not be deallocated!!!
44058                                  <1>       ;       (they will be deallocated at process termination)
44059                                  <1>       ;
44060                                  <1>       ; Note: When the program terminates itself or when it is
44061                                  <1>       ;     terminated by operating system kernel, all allocated
44062                                  <1>       ;     memory pages will be deallocated during termination
44063                                  <1>       ;     stage. So, 'sysdalloc' is not necessary except
44064                                  <1>       ;     forgiving memory block to other programs/processes.
44065                                  <1>       ;
44066 0000F105 8B15[5C030600]          <1>       mov   edx, [u.sp]
44067 0000F10B 8B420C                 <1>       mov   eax, [edx+12] ; user's stack pointer
44068 0000F10E 29C8                   <1>       sub   eax, ecx ; esp - byte count
44069 0000F110 24FC                   <1>       and   al, 0FCh ; dword alignment
44070 0000F112 39D8                   <1>       cmp   eax, ebx
44071 0000F114 7220                   <1>       jb    short sysdalloc_err ; deallocation overlaps with stack
44072                                  <1>
44073 0000F116 31C0                   <1>       xor   eax, eax
44074 0000F118 21C9                   <1>       and   ecx, ecx
44075 0000F11A 7407                   <1>       jz    short sysdalloc_2
44076                                  <1>
44077 0000F11C E85D66FFFF             <1>       call  deallocate_user_pages
44078 0000F121 7213                   <1>       jc    short sysdalloc_err
44079                                  <1>
44080                                  <1> sysdalloc_2:
44081 0000F123 A3[64030600]           <1>       mov   [u.r0], eax
44082 0000F128 8B2D[60030600]         <1>       mov   ebp, [u.usp]
44083 0000F12E 895D10                 <1>       mov   [ebp+16], ebx ; new value of ebx
44084 0000F131 E979D3FFFF             <1>       jmp   sysret
44085                                  <1>
44086                                  <1> sysdalloc_err:
44087 0000F136 A3[64030600]           <1>       mov   [u.r0], eax ; 0
44088 0000F13B E94FD3FFFF             <1>       jmp   error
44089                                  <1>
44090                                  <1> syscalbac:
44091                                  <1>       ; SYS CALLBACK
44092                                  <1>       ; 16/04/2017
44093                                  <1>       ; 14/04/2017
44094                                  <1>       ; 13/04/2017
44095                                  <1>       ; 28/02/2017
44096                                  <1>       ; 26/02/2017
44097                                  <1>       ; 24/02/2017
44098                                  <1>       ; 21/02/2017 - TRDOS 386 (TRDOS v2.0)
44099                                  <1>       ; (TRDOS 386 feature only!)
44100                                  <1>       ;
44101                                  <1>       ; Link or unlink IRQ callback service to/from user (ring 3)
44102                                  <1>       ;
44103                                  <1>       ; INPUT ->
44104                                  <1>       ;     BL = IRQ number (Hardware interrupt request number)
44105                                  <1>       ;          (0 t0 15 but IRQ 0,1,2,6,8,14,15 are prohibited)
44106                                  <1>       ;          IRQ numbers 3,4,5,7,9,10,11,12,13 are valid
44107                                  <1>       ;          (numbers >15 are invalid)
44108                                  <1>       ;
44109                                  <1>       ;     BH = 0 = Unlink IRQ (in BL) from user (ring 3) service
44110                                  <1>       ;          1 = Link IRQ by using Signal Response Byte method
44111                                  <1>       ;          2 = Link IRQ by using Callback service method
44112                                  <1>       ;          3 = Link IRQ by using Auto Increment S.R.B. method
44113                                  <1>       ;         >3 = invalid
44114                                  <1>       ;
44115                                  <1>       ;     CL = Signal Return/Response Byte value
44116                                  <1>       ;
44117                                  <1>       ;     If BH = 2, kernel will put a counter value
44118                                  <1>       ;               (into the S.R.B. addr)
44119                                  <1>       ;              between 0 to 255. (start value = CL+1)
44120                                  <1>       ;
44121                                  <1>       ;     NOTE: counter value, for example: even and odd numbers
44122                                  <1>       ;           may be used for -audio- DMA buffer switch
44123                                  <1>       ;           within double buffer method, etc.
44124                                  <1>       ;
44125                                  <1>       ;     EDX = Signal return (Response) byte address
44126                                  <1>       ;                           - or -
44127                                  <1>       ;           Interrupt/Callback service/routine address
44128                                  <1>       ;
44129                                  <1>       ;           (virtual address in user's memory space)
44130                                  <1>       ;
44131                                  <1>       ; OUTPUT ->
44132                                  <1>       ;     CF = 0 & EAX = 0 -> Successful setting
44133                                  <1>       ;     CF = 1 & EAX > 0 -> IRQ is prohibited or locked
44134                                  <1>       ;                         by another process
44135                                  <1>       ;             eax = ERR_PERM_DENIED -> prohibited or locked
44136                                  <1>       ;             eax = ERR_INV_PARAMETER ->
44137                                  <1>       ;                   invalid parameter/option or bad address
44138                                  <1>       ;
44139                                  <1>       ;     NOTE: Timer callbacks are set by using 'systimer'
44140                                  <1>       ;           system call (IRQ 0, PIT and IRQ 8, RTC)
44141                                  <1>       ;
44142                                  <1>       ;           Direct keyboard access is performed by using
44143                                  <1>       ;           Keyboard Interrupt (INT 32h)
44144                                  <1>       ;
44145                                  <1>       ;           It is prohibited here because:
44146                                  <1>       ;            1) Signal Response Byte method has not advantage
44147                                  <1>       ;               against INT 32h, function AH = 1. Also,
44148                                  <1>       ;               keyboard service interrupt will return with
44149                                  <1>       ;               ascii and scan codes (AL, AH) while
44150                                  <1>       ;               SRB method has only 1 byte space for ascii code
44151                                  <1>       ;               or scan code. One byte signal response is used
44152                                  <1>       ;               for ensuring very simple and very fast
44153                                  <1>       ;               virtual to physical memory address conversion
44154                                  <1>       ;               without any memory page crossover risk.
44155                                  <1>       ;               (Otherwise double page conversion or word
44156                                  <1>       ;               alignment would be needed.)
44157                                  <1>       ;            2) Badly written user code (callback code)
```

```
44158                                  <1>       ;               can prevent keyboard and timesharing functions
44159                                  <1>       ;               of the operating system via continuous and long
44160                                  <1>       ;               keyboard event handling by callback service.
44161                                  <1>       ;               (It can cause to lose immediate keystroke
44162                                  <1>       ;               response from hardware to user.)
44163                                  <1>       ;            3) If user will check any keyboard events, 'getkey'
44164                                  <1>       ;               (or 'getchar') must have more priority than other
44165                                  <1>       ;               (video etc.) events because only control ability
44166                                  <1>       ;               on a procedural infinite loop is a keyboard or
44167                                  <1>       ;               mouse event. So user can use keyboard function
44168                                  <1>       ;               at the end or at the beginning of a loop.
44169                                  <1>       ;               In this case, INT 32h is used for that purpose
44170                                  <1>       ;               and timer interrupt etc. callbacks can be used
44171                                  <1>       ;               for dynamic and synchronized data refresh/transfer
44172                                  <1>       ;               while cpu is in a static loop (without polling).
44173                                  <1>       ;               Keyboard Int callback is not more useful because
44174                                  <1>       ;               already a manual check (a key is pressed or not)
44175                                  <1>       ;               can be performed (via INT 32h, AH = 1) efficiently
44176                                  <1>       ;               in a loop to prevent a locked infinitive loop.
44177                                  <1>       ;
44178                                  <1>       ;          Disk IRQs (6,14,15) have been phohibited from ring 3
44179                                  <1>       ;          callback because, disk operations (file system services
44180                                  <1>       ;          etc.) are independent from user program, for fast disk r/w.
44181                                  <1>       ;          They are not more useful at ring 3 while they are in use
44182                                  <1>       ;          by standard diskio functions which are mandatory part of
44183                                  <1>       ;          (monolithic) OS kernel and mainprog command interpreter.
44184                                  <1>       ;          INT 33h diskio functions are enough for user level disk
44185                                  <1>       ;          r/w.
44186                                  <1>       ;
44187                                  <1>       ; TRDOS 386 - IRQ CALLBACK structures (parameters):
44188                                  <1>       ;
44189                                  <1>       ;         [u.irqlock] = 1 word, IRQ flags (0-15) that indicates
44190                                  <1>       ;                       which IRQs are locked by (that) user.
44191                                  <1>       ;                         Lock and unlock (by user) will change
44192                                  <1>       ;                       these flags or 'terminate process' (sysexit)
44193                                  <1>       ;                       will clear these flags and unlock those IRQs.
44194                                  <1>       ;
44195                                  <1>       ;                       Bit 0 is for IRQ 0 and Bit 15 is for IRQ 15
44196                                  <1>       ;
44197                                  <1>       ;          IRQ(x).owner      : 1 byte, user, [u.uno], 0 = free (unlocked)
44198                                  <1>       ;
44199                                  <1>       ;          IRQ(x).method : 1 byte for callback method & status
44200                                  <1>       ;                          0 = Signal Response Byte method
44201                                  <1>       ;                          1 = Callback service method
44202                                  <1>       ;                          >1 = invalid for current 'syscalback'.
44203                                  <1>       ;                       or(+) 80h = IRQ is in use by system (ring 0)
44204                                  <1>       ;                                 function (audio etc.) or
44205                                  <1>       ;                                a device driver.
44206                                  <1>       ;                       (system function will ignore the lock/owner)
44207                                  <1>       ;
44208                                  <1>       ;          IRQ(x).srb: 1 byte, Signal Return/Response byte value
44209                                  <1>       ;                       (a fixed value by user or a counter value
44210                                  <1>       ;                        from 0 to 255, which is increased by every
44211                                  <1>       ;                        interrupt just before putting it into
44212                                  <1>       ;                        the Signal Response byte address
44213                                  <1>       ;                        (This is not used in callback serv method)
44214                                  <1>       ;
44215                                  <1>       ;          IRQ(x).addr       : 1 dword
44216                                  <1>       ;                        Signal Response Byte address (physical)
44217                                  <1>       ;                                   -or-
44218                                  <1>       ;                        Callback service address (virtual)
44219                                  <1>       ;
44220                                  <1>       ;          IRQ(x).dev: 1 byte
44221                                  <1>       ;                        0 = Default device or kernel function
44222                                  <1>       ;                                  -or-
44223                                  <1>       ;                        1-255 = Assigned device driver number
44224                                  <1>       ;
44225                                  <1>       ;          (x) = 3,4,5,7,9,10,11,12,13
44226                                  <1>       ;
44227                                  <1>       ;
44228                                  <1>       ;      NOTE: If user's process/program calls the kernel (INT 40h)
44229                                  <1>       ;            while it is already running in a (ring 3) callback
44230                                  <1>       ;            service, kernel will force (convert) system call to
44231                                  <1>       ;            'sysrele' (sys release). So, this feature provides
44232                                  <1>       ;            easy and simple usage of callback services without
44233                                  <1>       ;            falling into deepless <please 'callback me' then
44234                                  <1>       ;            let me 'callback you'> cycles! (User must return
44235                                  <1>       ;            from callback service by using 'sysrele' system
44236                                  <1>       ;            call, without a significant delay. Otherwise user
44237                                  <1>       ;            process/program may be late to catch the next event
44238                                  <1>       ;            within same callback purpose.
44239                                  <1>       ;
44240                                  <1>
44241 0000F140 30C0                   <1>       xor    al, al ; the caller is 'syscalbac' sign/flag
44242 0000F142 E88A100000             <1>       call   set_irq_callback_service
44243                                  <1>       ; 16/04/2017
44244 0000F147 A3[64030600]           <1>       mov    [u.r0], eax
44245 0000F14C 0F835DD3FFFF           <1>       jnc    sysret
44246 0000F152 A3[C8030600]           <1>       mov    dword [u.error], eax
44247 0000F157 E933D3FFFF             <1>       jmp    error
44248                                  <1>
44249                                  <1> sysfpstat:
44250                                  <1>       ; 28/02/2017 - TRDOS 386 (TRDOS v2.0)
44251                                  <1>       ; (TRDOS 386 feature only!)
44252                                  <1>       ;
44253                                  <1>       ; Set or reset FPU registers save/restore option (for user)
44254                                  <1>       ;             (during software task switching, wswap-rswap)
44255                                  <1>       ;
44256                                  <1>       ; INPUT ->
44257                                  <1>       ;     BL = 0 -> reset
44258                                  <1>       ;     BL = 1 -> set (FPU register will be saved and restored)
44259                                  <1>       ;
```

```
44260                                  <1>       ; OUTPUT ->
44261                                  <1>       ;      cf = 0 -> no error, FPU is ready...
44262                                  <1>       ;                (EAX = 0)
44263                                  <1>       ;      Cf = 1 -> error, 80387 FPU is not ready !
44264                                  <1>       ;                (EAX = 0FFFFFFFFh)
44265                                  <1>
44266 0000F15C 31C0                   <1>       xor    eax, eax
44267 0000F15E 803D[885B0100]00       <1>       cmp    byte [fpready], 0
44268 0000F165 7613                   <1>       jna    short sysfpstat_err
44269                                  <1>
44270 0000F167 80E301                 <1>       and    bl, 1 ; use BIT 0 only !
44271 0000F16A 881D[DA030600]         <1>       mov    [u.fpsave], bl
44272 0000F170 A3[64030600]           <1>       mov    [u.r0], eax ; 0
44273 0000F175 E935D3FFFF             <1>       jmp    sysret
44274                                  <1>
44275                                  <1> sysfpstat_err:
44276 0000F17A 48                     <1>       dec    eax ; 0FFFFFFFFh
44277 0000F17B A3[64030600]           <1>       mov    [u.r0], eax ; -1
44278 0000F180 E90AD3FFFF             <1>       jmp    error
44279                                  <1>
44280                                  <1> ;maknod:
44281                                  <1>       ; 26/10/2016
44282                                  <1>       ; temporary
44283                                  <1> ;     retn
44284                                  <1>
44285                                  <1> ; temporary - 24/01/2016
44286                                  <1>
44287                                  <1> iget:
44288 0000F185 C3                     <1>       retn
44289                                  <1> isintr:
44290 0000F186 C3                     <1>       retn
44291                                  <1> iopen:
44292 0000F187 C3                     <1>       retn
44293                                  <1> iclose:
44294 0000F188 C3                     <1>       retn
44295                                  <1> setimod:
44296 0000F189 C3                     <1>       retn
44297                                  <1> sndc:
44298 0000F18A C3                     <1>       retn
44299                                  <1> access:
44300 0000F18B C3                     <1>       retn
44301                                  <1> epoch:
44302 0000F18C C3                     <1>       retn
44303                                  <1> sleep:
44304 0000F18D C3                     <1>       retn
44305                                  <1> set_date_time:
44306 0000F18E C3                     <1>       retn
44307                                      %include 'trdosk7.s' ; 24/01/2016
44308                                  <1> ; ****************************************************************************
44309                                  <1> ; TRDOS386.ASM (TRDOS 386 Kernel - v2.0.0) - DISK READ&WRITE : trdosk7.s
44310                                  <1> ; ----------------------------------------------------------------------------
44311                                  <1> ; Last Update: 25/02/2016
44312                                  <1> ; ----------------------------------------------------------------------------
44313                                  <1> ; Beginning: 24/01/2016
44314                                  <1> ; ----------------------------------------------------------------------------
44315                                  <1> ; Assembler: NASM version 2.11 (trdos386.s)
44316                                  <1> ; ----------------------------------------------------------------------------
44317                                  <1> ; Derived from TRDOS Operating System v1.0 (8086) source code by Erdogan Tan
44318                                  <1> ; DISK_IO.ASM (20/07/2011)
44319                                  <1> ; ****************************************************************************
44320                                  <1> ; DISK_IO.ASM (c) 2009-2011 Erdogan TAN [ 04/07/2009 ] Last Update: 20/07/2011
44321                                  <1>
44322                                  <1> disk_write:
44323                                  <1>       ; 25/02/2016
44324                                  <1>       ; 24/02/2016
44325                                  <1>       ; 23/02/2016
44326 0000F18F 807E0500               <1>       cmp    byte [esi+LD_LBAYes], 0
44327 0000F193 777B                   <1>         ja      short lba_write
44328                                  <1>
44329                                  <1> chs_write:
44330                                  <1>       ; 25/02/2016
44331                                  <1>       ; 23/02/2016
44332 0000F195 C605[81570100]03       <1>       mov    byte [disk_rw_op], 3 ; CHS write
44333 0000F19C EB0D                   <1>       jmp    short chs_rw
44334                                  <1>
44335                                  <1> disk_read:
44336                                  <1>       ; 25/02/2016
44337                                  <1>       ; 24/02/2016
44338                                  <1>       ; 23/02/2016
44339                                  <1>       ; 17/02/2016
44340                                  <1>       ; 14/02/2016
44341                                  <1>       ; 31/01/2016 (TRDOS 386 =  TRDOS v2.0)
44342                                  <1>       ; 17/10/2010
44343                                  <1>       ; 18/04/2010
44344                                  <1>       ;
44345                                  <1>       ; INPUT -> EAX = Logical Block Address
44346                                  <1>       ;         ESI = Logical Dos Disk Table Offset (DRV)
44347                                  <1>       ;         ECX = Sector Count
44348                                  <1>       ;         EBX = Destination Buffer
44349                                  <1>       ; OUTPUT ->
44350                                  <1>       ;         cf = 0 or cf = 1
44351                                  <1>       ; (Modified registers: EAX, EBX, ECX, EDX)
44352                                  <1>
44353 0000F19E 807E0500               <1>       cmp    byte [esi+LD_LBAYes], 0
44354 0000F1A2 7775                   <1>         ja      short lba_read
44355                                  <1>
44356                                  <1> chs_read:
44357                                  <1>       ; 25/02/2016
44358                                  <1>       ; 24/02/2016
44359                                  <1>       ; 23/02/2016
44360                                  <1>       ; 31/01/2016 (TRDOS 386 =  TRDOS v2.0)
44361                                  <1>       ; 20/07/2011
```

```
44362                                  <1>       ; 04/07/2009
44363                                  <1>       ;
44364                                  <1>       ; INPUT -> EAX = Logical Block Address
44365                                  <1>       ;         ECX = Number of sectors to read
44366                                  <1>       ;         ESI = Logical Dos Disk Table Offset (DRV)
44367                                  <1>       ;         EBX = Destination Buffer
44368                                  <1>       ; OUTPUT ->
44369                                  <1>       ;         cf = 0 or cf = 1
44370                                  <1>       ; (Modified registers: EAX; EBX, ECX, EDX)
44371                                  <1>
44372                                  <1>       ; 23/02/2016
44373 0000F1A4 C605[81570100]02       <1>       mov    byte [disk_rw_op], 2 ; CHS read
44374                                  <1>
44375                                  <1> chs_rw:
44376                                  <1>       ;;movzx      edx, word [esi+LD_BPB+SecPerTrack]
44377                                  <1>       ;movzx edx, byte [esi+LD_BPB+SecPerTrack] ; <= 63
44378                                  <1>       ;mov   [disk_rw_spt], dl
44379                                  <1>
44380                                  <1> chs_read_next_sector:
44381 0000F1AB C605[82570100]04       <1>       mov    byte [retry_count], 4
44382                                  <1>
44383                                  <1> chs_read_retry:
44384                                  <1>       ;mov   [sector_count], ecx ; 23/02/2016
44385                                  <1>
44386 0000F1B2 50                     <1>       push   eax                ; Linear sector #
44387 0000F1B3 51                     <1>       push   ecx                ; # of FAT/FILE/DIR sectors
44388                                  <1>
44389 0000F1B4 0FB74E1E               <1>       movzx  ecx, word [esi+LD_BPB+SecPerTrack]
44390                                  <1>       ;movzx ecx, byte [disk_rw_spt] ; 23/02/2016
44391 0000F1B8 29D2                   <1>       sub    edx, edx
44392 0000F1BA F7F1                   <1>       div    ecx
44393                                  <1>       ; eax = track, dx (dl ) = sector (on track)
44394                                  <1>       ;sub   cl, dl ; 24/02/2016 (spt - sec)
44395                                  <1>       ;push  ecx ; *
44396 0000F1BC 6689D1                 <1>       mov    cx, dx             ; Sector (zero based)
44397 0000F1BF 6641                   <1>       inc    cx                 ; To make it 1 based
44398 0000F1C1 6651                   <1>       push   cx
44399 0000F1C3 668B4E20               <1>       mov    cx, [esi+LD_BPB+Heads]
44400 0000F1C7 6629D2                 <1>       sub    dx, dx
44401 0000F1CA F7F1                   <1>       div    ecx                ; Convert track to head & cyl
44402                                  <1>       ; eax (ax) = cylinder, dx (dl) = head (max. FFh)
44403 0000F1CC 88D6                   <1>       mov    dh, dl
44404 0000F1CE 6659                   <1>       pop    cx                 ; AX=Cyl, DH=Head, CX=Sector
44405 0000F1D0 8A5602                 <1>       mov    dl, [esi+LD_PhyDrvNo]
44406                                  <1>
44407 0000F1D3 88C5                   <1>       mov    ch, al             ; NOTE: max. 1023 cylinders !
44408 0000F1D5 C0CC02                 <1>       ror    ah, 2              ; Rotate 2 bits right
44409 0000F1D8 08E1                   <1>       or     cl, ah
44410                                  <1>
44411                                  <1>       ; 24/02/2016
44412                                  <1>       ;pop   eax ; * (spt - sec) (example: 63 - 0 = 63)
44413                                  <1>       ;cmp   eax, [sector_count]
44414                                  <1>       ;jb    short chs_write_sectors
44415                                  <1>       ;je    short chs_read_sectors
44416                                  <1>       ;; (# of sectors to read is more than remaining sectors on the track)
44417                                  <1>       ;mov   al, [sector_count]
44418                                  <1> ;chs_read_sectors: ; read or write !
44419 0000F1DA B001                   <1>       mov    al, 1 ; 25/02/2016
44420 0000F1DC 8A25[81570100]         <1>       mov    ah, [disk_rw_op]  ; 02h = chs read, 03h = chs write
44421                                  <1>       ;
44422 0000F1E2 E81F50FFFF             <1>       call   int13h             ; BIOS Service func ( ah ) = 2
44423                                  <1>                                 ; Read disk sectors
44424                                  <1>                                 ; AL-sec num CH-track CL-sec
44425                                  <1>                                 ; DH-head DL-drive ES:BX-buffer
44426                                  <1>                                 ; CF-flag AH-stat AL-sec read
44427                                  <1>                                ; If CF = 1 then (If AH > 0)
44428 0000F1E7 8825[83570100]         <1>       mov    [disk_rw_err], ah
44429                                  <1>
44430 0000F1ED 59                     <1>       pop    ecx
44431 0000F1EE 58                     <1>       pop    eax
44432 0000F1EF 7314                   <1>       jnc    short chs_read_ok
44433                                  <1>
44434 0000F1F1 803D[83570100]09       <1>       cmp    byte [disk_rw_err], 09h ; DMA crossed 64K segment boundary
44435 0000F1F8 7408                   <1>       je     short chs_read_error_retn
44436                                  <1>
44437 0000F1FA FE0D[82570100]         <1>       dec    byte [retry_count]
44438 0000F200 75B0                   <1>       jnz    short chs_read_retry
44439                                  <1>
44440                                  <1> chs_read_error_retn:
44441 0000F202 F9                     <1>       stc
44442                                  <1>       ;retn
44443 0000F203 EB69                   <1>       jmp    short update_drv_error_byte
44444                                  <1>
44445                                  <1> ;chs_write_sectors: ; read or write
44446                                  <1>       ;; (# of sectors to read is less than remaining sectors on the track)
44447                                  <1>       ;mov   [sector_count], al
44448                                  <1>       ;jmp   short chs_read_sectors
44449                                  <1>
44450                                  <1> chs_read_ok:
44451                                  <1>       ;; 23/02/2016
44452                                  <1>       ;movzx edx, byte [sector_count] ; sector count (<= spt)
44453                                  <1>         ;sub    ecx, edx  ; remaining sector count
44454                                  <1>       ;jna   short update_drv_error_byte
44455                                  <1>       ;add   eax, edx ; next disk sector
44456                                  <1>       ;shl   edx, 9 ; 512 * sector count
44457                                  <1>       ;add   ebx, edx ; next buffer byte address
44458                                  <1>         ;jmp     chs_read_next_sector
44459                                  <1>       ; 25/02/2016
44460 0000F205 40                     <1>       inc    eax ; next sector
44461 0000F206 81C300020000           <1>       add    ebx, 512
44462 0000F20C E29D                   <1>       loop   chs_read_next_sector
44463 0000F20E EB5E                   <1>       jmp    short update_drv_error_byte
```

```
44464                                  <1>
44465                                  <1> lba_write:
44466                                  <1>       ; 23/02/2016
44467 0000F210 C605[81570100]1C        <1>       mov    byte [disk_rw_op], 1Ch ; LBA write
44468 0000F217 EB07                    <1>       jmp    short lba_rw
44469                                  <1>
44470                                  <1> lba_read:
44471                                  <1>       ; 23/02/2016
44472                                  <1>       ; 17/02/2016
44473                                  <1>       ; 14/02/2016
44474                                  <1>       ; 13/02/2016
44475                                  <1>       ; 31/01/2016 (TRDOS 386 =  TRDOS v2.0)
44476                                  <1>       ; 10/07/2015 (Retro UNIX 386 v1)
44477                                  <1>       ;
44478                                  <1>       ; INPUT -> EAX = Logical Block Address
44479                                  <1>       ;         ESI = Logical Dos Disk Table Offset (DRV)
44480                                  <1>       ;         ECX = Sector Count
44481                                  <1>       ;         EBX = Destination Buffer
44482                                  <1>       ; OUTPUT ->
44483                                  <1>       ;         cf = 0 or cf = 1
44484                                  <1>       ; (Modified registers: EAX, EBX, ECX, EDX)
44485                                  <1>
44486                                  <1>       ; LBA read/write (with private LBA function)
44487                                  <1>       ;((Retro UNIX 386 v1 - DISK I/O code by Erdogan Tan))
44488                                  <1>
44489                                  <1>
44490                                  <1>       ; 23/02/2016
44491 0000F219 C605[81570100]1B        <1>       mov    byte [disk_rw_op], 1Bh ; LBA read
44492                                  <1>
44493                                  <1> lba_rw:
44494                                  <1>       ; 17/02/2016
44495 0000F220 57                      <1>       push   edi
44496                                  <1>
44497 0000F221 890D[84570100]          <1>       mov    [sector_count], ecx ; total sector (read) count
44498                                  <1>
44499 0000F227 8A5602                  <1>       mov    dl, [esi+LD_PhyDrvNo]
44500                                  <1>       ; dl = physical drive number (0,1, 80h, 81h, 82h, 83h)
44501                                  <1>
44502                                  <1> lba_read_next:
44503 0000F22A 81F900010000            <1>       cmp    ecx, 256
44504 0000F230 7605                    <1>       jna    short lba_read_rsc
44505 0000F232 B900010000              <1>       mov    ecx, 256 ; 17/02/2016
44506                                  <1> lba_read_rsc:
44507 0000F237 290D[84570100]          <1>       sub    [sector_count], ecx ; remain sectors
44508                                  <1>
44509 0000F23D 89CF                    <1>       mov    edi, ecx
44510 0000F23F 89C1                    <1>       mov    ecx, eax ; sector number/address
44511                                  <1>
44512 0000F241 C605[82570100]04        <1>       mov    byte [retry_count], 4
44513                                  <1> lba_read_retry:
44514 0000F248 89F8                    <1>       mov    eax, edi
44515                                  <1>       ;
44516                                  <1>       ; ecx = sector number
44517                                  <1>       ; al = sector count (0 - 255) /// (0 = 256)
44518                                  <1>       ; dl = drive number
44519                                  <1>       ; ebx = buffer offset
44520                                  <1>       ;
44521                                  <1>       ; Function 1Bh = LBA read, 1Ch = LBA write
44522                                  <1>       ; 23/02/2016
44523 0000F24A 8A25[81570100]          <1>       mov    ah, [disk_rw_op] ; 1Bh = LBA read, 1Ch = LBA write
44524 0000F250 E8B14FFFFF              <1>       call   int13h
44525                                  <1>       ; al = ? (changed)
44526                                  <1>       ; ah = error code
44527 0000F255 8825[83570100]          <1>       mov    [disk_rw_err], ah
44528 0000F25B 7334                    <1>       jnc    short lba_read_ok
44529 0000F25D 80FC80                  <1>       cmp    ah, 80h ; time out?
44530 0000F260 740A                    <1>         je     short lba_read_stc_retn
44531 0000F262 FE0D[82570100]          <1>       dec    byte [retry_count]
44532 0000F268 7FDE                    <1>       jg     short lba_read_retry
44533 0000F26A 743A                    <1>       jz     short lba_read_reset
44534                                  <1>       ; sf =  1
44535                                  <1>
44536                                  <1> lba_read_stc_retn:
44537 0000F26C F9                      <1>       stc
44538                                  <1> lba_read_retn:
44539 0000F26D 5F                      <1>       pop    edi
44540                                  <1>
44541                                  <1> update_drv_error_byte:
44542 0000F26E 9C                      <1>       pushf
44543 0000F26F 53                      <1>       push   ebx
44544 0000F270 6651                    <1>       push   cx
44545                                  <1>       ;or    ecx, ecx
44546                                  <1>       ;jz    short udrv_errb0
44547 0000F272 8A0D[83570100]          <1>       mov    cl, [disk_rw_err]
44548                                  <1> udrv_errb0:
44549 0000F278 0FB65E02                <1>       movzx  ebx, byte [esi+LD_PhyDrvNo]
44550 0000F27C 80FB02                  <1>       cmp    bl, 2
44551 0000F27F 7203                    <1>         jb     short udrv_errb1
44552 0000F281 80EB7E                  <1>       sub    bl, 7Eh
44553                                  <1>       ;cmp   bl, 5
44554                                  <1>       ;ja    short udrv_errb2
44555                                  <1> udrv_errb1:
44556 0000F284 81C3[495D0000]          <1>         add    ebx, drv.error ; 13/02/2016
44557 0000F28A 880B                    <1>       mov    [ebx], cl ; error code
44558                                  <1> udrv_errb2:
44559 0000F28C 6659                    <1>       pop    cx
44560 0000F28E 5B                      <1>       pop    ebx
44561 0000F28F 9D                      <1>       popf
44562 0000F290 C3                      <1>       retn
44563                                  <1>
44564                                  <1> lba_read_ok:
44565 0000F291 89C8                    <1>       mov    eax, ecx ; sector number
```

```
44566 0000F293 01F8              <1>       add    eax, edi ; sector number (next)
44567 0000F295 C1E709            <1>       shl    edi, 9 ; sector count * 512
44568 0000F298 01FB              <1>       add    ebx, edi ; next buffer offset
44569                            <1>
44570 0000F29A 8B0D[84570100]    <1>       mov    ecx, [sector_count] ; remaining sectors
44571 0000F2A0 09C9              <1>       or     ecx, ecx
44572 0000F2A2 7586              <1>       jnz    short lba_read_next
44573 0000F2A4 EBC7              <1>       jmp    short lba_read_retn
44574                            <1>
44575                            <1> lba_read_reset:
44576 0000F2A6 B40D              <1>       mov    ah, 0Dh ; Alternate reset
44577 0000F2A8 E8594FFFFF        <1>         call int13h
44578                            <1>       ; al = ? (changed)
44579                            <1>       ; ah = error code
44580 0000F2AD 7399              <1>       jnc    short lba_read_retry
44581 0000F2AF EBBC              <1>       jmp    short lba_read_retn
44582                                %include 'trdosk8.s' ; 24/01/2016
44583                            <1> ; ****************************************************************************
44584                            <1> ; TRDOS386.ASM (TRDOS 386 Kernel - v2.0.0) - MAIN PROGRAM : trdosk8.s
44585                            <1> ; ----------------------------------------------------------------------------
44586                            <1> ; Last Update: 22/06/2017
44587                            <1> ; ----------------------------------------------------------------------------
44588                            <1> ; Beginning: 24/01/2016
44589                            <1> ; ----------------------------------------------------------------------------
44590                            <1> ; Assembler: NASM version 2.11 (trdos386.s)
44591                            <1> ; ----------------------------------------------------------------------------
44592                            <1> ; Derived from 'Retro UNIX 386 Kernel - v0.2.1.0' source code by Erdogan Tan
44593                            <1> ; u0.s (20/11/2015), u4.s (14/10/2015)
44594                            <1> ; ****************************************************************************
44595                            <1> ; Derived from TRDOS Operating System v1.0 (8086) source code by Erdogan Tan
44596                            <1> ; TRDOS2.ASM (09/11/2011)
44597                            <1> ; ----------------------------------------------------------------------------
44598                            <1> ; DIR.ASM (c) 2004-2011 Erdogan TAN  [07/01/2004] Last Update: 09/10/2011
44599                            <1>
44600                            <1> set_run_sequence:
44601                            <1>       ; 23/12/2016
44602                            <1>       ; 10/06/2016
44603                            <1>       ; 22/05/2016
44604                            <1>       ; 20/05/2016
44605                            <1>       ; 19/05/2016 - TRDOS 386 (TRDOS v2.0)
44606                            <1>       ; TRDOS 386 feature only !
44607                            <1>       ;
44608                            <1>       ; INPUT ->
44609                            <1>       ;     AL = process number (next process)
44610                            <1>       ;
44611                            <1>       ;     this process must be added to run sequence
44612                            <1>       ;
44613                            <1>       ;     [u.pri] = priority of present process
44614                            <1>       ;
44615                            <1>       ;     DL = priority (queue)
44616                            <1>       ;          0 = background (low) ; run on background
44617                            <1>       ;          1 = regular (normal) ; run as regular
44618                            <1>       ;          2 = event (high)     ; run for event
44619                            <1>       ;
44620                            <1>       ;     1) If the requested process is already running:
44621                            <1>       ;        a) If present priority is high ([u.pri]=2)
44622                            <1>       ;           and requested priority is also high,
44623                            <1>       ;           there is nothing to do! Because it has been
44624                            <1>       ;           done already (before this attempt).
44625                            <1>       ;        b) If present priority is high ([u.pri]=2)
44626                            <1>       ;           and requested priority is not high, there is
44627                            <1>       ;           nothing to do! Because, it's current
44628                            <1>       ;           run queue is unspecified, here. (It may be in
44629                            <1>       ;           a waiting list or in a run queue; if the new
44630                            <1>       ;           priority would be used to add it to relavant
44631                            <1>         ;             run queue, this would be wrong, unnecessary
44632                            <1>       ;           and destabilizing duplication!)
44633                            <1>       ;        c) If present priority is not high ([u.pri]<2)
44634                            <1>         ;             and requested priority is high (event),
44635                            <1>       ;           process will be added to present priority's
44636                            <1>       ;           run queue and then, priority will be changed
44637                            <1>       ;           to high ([u.pri]=2).
44638                            <1>       ;        d) If present priority is not high ([u.pri]<2)
44639                            <1>       ;           and requested priority is not high, [u.pri]
44640                            <1>       ;           value will be changed. There is nothing to do
44641                            <1>       ;           in addition. (The new priority value will be
44642                            <1>       ;           used by 'tswap/tswitch' procedure at 'sysret'
44643                            <1>       ;           or 'sysrele' stage.)
44644                            <1>       ;
44645                            <1>       ;     2) If the requested process is not running:
44646                            <1>       ;        a) If requested priority of the requested
44647                            <1>       ;           (next) process is high (event) and priority
44648                            <1>       ;           of present process is not high, the requested
44649                            <1>       ;           process will be added to ('runq_event') high
44650                            <1>       ;           priority run queue and then present (running)
44651                            <1>       ;           process will be stopped (swapped/switched out)
44652                            <1>       ;           immediately if it is in user mode, or it's
44653                            <1>       ;           [u.quant] value will be reset to 0 and (then)
44654                            <1>       ;           it will be stopped at 'sysret' stage.
44655                            <1>       ;        b) If requested priority of the requested
44656                            <1>       ;           (next) process is high (event) and priority
44657                            <1>       ;           of present process is also high, the requested
44658                            <1>       ;           process will be added to ('runq_event') high
44659                            <1>       ;           priority run queue and present (running)
44660                            <1>       ;           process will be allowed to run until it's
44661                            <1>       ;           time quantum will be elapsed ([u.quant]=0).
44662                            <1>       ;        c) If requested priority of the requested
44663                            <1>       ;           (next) process is not high ('run for event'),
44664                            <1>       ;           there is nothing to do. Because, it's current
44665                            <1>       ;           run queue is unspecified, here. (It may be in
44666                            <1>       ;           a waiting list or in a run queue; if the new
44667                            <1>       ;           priority would be used to add it to relavant
```

```
44668                                  <1>         ;             run queue, this would be wrong, unnecessary
44669                                  <1>       ;            and destabilizing duplication!)
44670                                  <1>       ;
44671                                  <1>       ; OUTPUT ->
44672                                  <1>       ;     none
44673                                  <1>       ;
44674                                  <1>       ;     [u.pri] = priority of present process
44675                                  <1>       ;
44676                                  <1>       ;     cf = 1, if the request could not be fulfilled.
44677                                  <1>       ;
44678                                  <1>       ;     NOTE:
44679                                  <1>         ;          * Processes in 'run as regular' queue can run
44680                                  <1>       ;          if there is no process in 'run for event' queue
44681                                  <1>       ;          ('run for event' processes have higher priority)
44682                                  <1>       ;        * When [u.quant] time quantum of a process is
44683                                  <1>       ;          elapsed, it's high priority ('run for event')
44684                                  <1>       ;          status will be disabled, it can be run in sequence
44685                                  <1>       ;          of it's actual run queue.
44686                                  <1>       ;        * A 'run on background' process will always be
44687                                  <1>       ;          sequenced in 'run on background' (low priority)
44688                                  <1>       ;          queue, it can run only when other priority queues
44689                                  <1>       ;          are empty. (idle time processes, e.g. printing)
44690                                  <1>       ;
44691                                  <1>       ; Modified registers: eax, ebx, edx
44692                                  <1>       ;
44693                                  <1>
44694                                  <1> srunseq_0:
44695 0000F2B1 3A05[B3030600]          <1>         cmp     al, [u.uno]     ; same process ?
44696 0000F2B7 750C                    <1>       jne    short srunseq_2 ; no
44697                                  <1>
44698 0000F2B9 8A25[A9030600]          <1>       mov    ah, [u.pri]  ; present/current priority
44699 0000F2BF 80FC02                  <1>       cmp    ah, 2        ; 'run for event' priority level
44700 0000F2C2 7221                    <1>       jb     short srunseq_6 ; no
44701                                  <1>
44702                                  <1> srunseq_1:
44703                                  <1>       ; there is nothing to do!
44704 0000F2C4 C3                      <1>       retn
44705                                  <1>
44706                                  <1> srunseq_2:
44707                                  <1>       ;;this not necessary ! 23/12/2016
44708                                  <1>         ;;cmp      al, nproc    ; number of processes = 16
44709                                  <1>       ;;jnb  short srunseq_5     ; error ! invalid process number
44710                                  <1>
44711                                  <1>       ; dl = priority
44712 0000F2C5 80FA02                  <1>       cmp    dl, 2        ; event queue
44713 0000F2C8 72FA                    <1>       jb     short srunseq_1 ; requested process is not present
44714                                  <1>                            ; process and priority of requested
44715                                  <1>                            ; process is not high (event),
44716                                  <1>                            ; there is nothing to do!
44717                                  <1>
44718                                  <1>       ; requested process is not present process
44719                                  <1>       ; & priority of requested process is high
44720 0000F2CA 3A15[A9030600]          <1>       cmp    dl, [u.pri]  ; priority of present process
44721 0000F2D0 7606                    <1>       jna    short srunseq_3 ; is high, also
44722                                  <1>       ;
44723                                  <1>       ; present process will be swapped/switched out
44724 0000F2D2 FE05[5D5B0100]          <1>       inc    byte [p_change] ; 1
44725                                  <1>
44726                                  <1> srunseq_3:
44727                                  <1>       ; add process to 'runq_event' queue for new event
44728 0000F2D8 BB[52030600]            <1>       mov    ebx, runq_event ; high priority run queue
44729                                  <1>
44730                                  <1> srunseq_4:
44731                                  <1>       ; al = process number
44732                                  <1>       ; ebx = run queue
44733 0000F2DD E8B5F4FFFF              <1>       call   putlu
44734 0000F2E2 C3                      <1>       retn
44735                                  <1>
44736                                  <1> srunseq_5:
44737 0000F2E3 F5                      <1>       cmc
44738 0000F2E4 C3                      <1>       retn
44739                                  <1>
44740                                  <1> srunseq_6:
44741                                  <1>       ; present priority of the process is not high
44742                                  <1>
44743 0000F2E5 8815[A9030600]          <1>       mov    [u.pri], dl ; new priority
44744                                  <1>                          ; (will be used by 'tswap')
44745                                  <1>
44746 0000F2EB 80FA02                  <1>       cmp    dl, 2        ; high priority ?
44747 0000F2EE 72F3                    <1>       jb     short srunseq_5 ; no, there is nothing to do
44748                                  <1>                            ; in addition
44749                                  <1>
44750                                  <1>       ; process must be added to relevant run queue, here!
44751                                  <1>       ; (new priority is high/event priority and process
44752                                  <1>       ; will not be added to a run queue by 'tswap')
44753                                  <1>
44754 0000F2F0 BB[54030600]            <1>       mov    ebx, runq_normal ; 'run as regular' queue
44755                                  <1>
44756 0000F2F5 20E4                    <1>       and    ah, ah  ;  previous value of [u.pri]
44757 0000F2F7 75E4                    <1>       jnz    short srunseq_4
44758                                  <1>
44759 0000F2F9 43                      <1>       inc    ebx
44760 0000F2FA 43                      <1>       inc    ebx
44761                                  <1>       ; ebx = runq_background ; 'run on backgroud' queue
44762                                  <1>
44763 0000F2FB EBE0                    <1>       jmp    short srunseq_4
44764                                  <1> clock:
44765                                  <1>       ; 23/05/2016
44766                                  <1>       ; 22/05/2016
44767                                  <1>       ; 20/05/2016
44768                                  <1>       ; 19/05/2016 - TRDOS 386 (TRDOS v2.0)
44769                                  <1>       ; 14/05/2015 - 14/10/2015 (Retro UNIX 386 v1)
```

```
44770                                  <1>       ; 07/12/2013 - 10/04/2014 (Retro UNIX 8086 v1)
44771                                  <1>
44772 0000F2FD 803D[A8030600]00        <1>       cmp    byte [u.quant], 0
44773 0000F304 772C                    <1>       ja     short clk_1
44774                                  <1>       ;
44775 0000F306 803D[B3030600]01        <1>       cmp     byte [u.uno], 1 ; /etc/init ? (for Retro UNIX 8086 & 386 v1)
44776                                  <1>                       ; MainProg (Kernel's Command Interpreter)
44777                                  <1>                       ; for TRDOS 386.
44778 0000F30D 7623                    <1>       jna    short clk_1 ; yes, do not swap out
44779                                  <1>       ;
44780 0000F30F 803D[5B030600]FF        <1>       cmp     byte [sysflg], 0FFh ; user or system space ?
44781 0000F316 7520                    <1>       jne    short clk_2      ; system space (sysflg <> 0FFh)
44782                                  <1>       ;
44783 0000F318 66833D[AA030600]00      <1>       cmp    word [u.intr], 0
44784 0000F320 7616                    <1>       jna    short clk_2
44785                                  <1>       ;
44786                                  <1>       ; 23/05/2016
44787 0000F322 803D[5E5B0100]00        <1>       cmp    byte [multi_tasking], 0
44788 0000F329 760D                    <1>       jna    short clk_2
44789                                  <1>       ;
44790 0000F32B FE05[5D5B0100]          <1>       inc    byte [p_change] ; it is time to change running process
44791 0000F331 C3                      <1>       retn
44792                                  <1> clk_1:
44793 0000F332 FE0D[A8030600]          <1>       dec    byte [u.quant]
44794                                  <1> clk_2:
44795 0000F338 C3                      <1>       retn   ; return to (hardware) timer interrupt routine
44796                                  <1>
44797                                  <1> ; 15/01/2017
44798                                  <1> ; 14/01/2017
44799                                  <1> ; 07/01/2017
44800                                  <1> ; 02/01/2017
44801                                  <1> ; 17/08/2016
44802                                  <1> ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
44803                                  <1> int34h:  ; #IOCTL# (I/O port access support for ring 3)
44804                                  <1> ; 23/05/2016
44805                                  <1>       ; 20/06/2016
44806                                  <1>       ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
44807                                  <1>       ;
44808                                  <1>       ; INPUT ->
44809                                  <1>       ;     AH = 0 -> read port  (physical IO port) -byte-
44810                                  <1>       ;     AH = 1 -> write port (physical IO port) -byte-
44811                                  <1>       ;           AL = data byte
44812                                  <1>       ;     AH = 2 -> read port  (physical IO port) -word-
44813                                  <1>       ;     AH = 3 -> write port (physical IO port) -word-
44814                                  <1>       ;           BX = data word
44815                                  <1>       ;     AH = 4 -> read port  (physical IO port) -dword-
44816                                  <1>       ;     AH = 5 -> write port (physical IO port) -dword-
44817                                  <1>       ;           EBX = data dword
44818                                  <1>       ;
44819                                  <1>       ;     DX = Port number (<= 0FFFFh)
44820                                  <1>       ;
44821                                  <1>       ; OUTPUT ->
44822                                  <1>       ;     AL = data byte (in al, dx)
44823                                  <1>       ;     AX = data word (in ax, dx)
44824                                  <1>       ;     EAX = data dword (in eax, dx)
44825                                  <1>       ;
44826                                  <1>       ;     (ECX = actual TRANSFER COUNT for string functions)
44827                                  <1>       ;
44828                                  <1>       ;
44829                                  <1>       ; Modified registers: EAX
44830                                  <1>       ;
44831                                  <1>
44832 0000F339 80FC05                  <1>       cmp    ah, 5
44833 0000F33C 7739                    <1>       ja     short int34h_5 ; invalid function !
44834                                  <1>
44835                                  <1>       ;; 15/01/2017
44836                                  <1>       ; 14/01/2017
44837                                  <1>       ; 02/01/2017
44838                                  <1>       ;;mov  byte [ss:intflg], 34h     ; IOCTL interrupt
44839 0000F33E FB                      <1>       sti
44840                                  <1>
44841                                  <1>       ;sti   ; enable interrupts
44842 0000F33F 80642408FE              <1>       and    byte [esp+8], 11111110b   ; clear carry bit of eflags register
44843                                  <1>
44844 0000F344 80FC01                  <1>       cmp    ah, 1
44845 0000F347 7205                    <1>       jb     short int34h_0
44846 0000F349 7705                    <1>       ja     short int34h_1
44847                                  <1>
44848 0000F34B EE                      <1>       out    dx, al
44849                                  <1>       ;iretd
44850 0000F34C EB01                    <1>       jmp short int34h_iret
44851                                  <1>
44852                                  <1> int34h_0:
44853 0000F34E EC                      <1>       in     al, dx
44854                                  <1>       ;iretd
44855                                  <1> int34h_iret:
44856                                  <1>       ;cli   ; 07/01/2017
44857                                  <1>       ;; 15/01/2017
44858                                  <1>       ;;mov  byte [ss:intflg], 0 ; reset
44859 0000F34F CF                      <1>       iretd
44860                                  <1>
44861                                  <1> int34h_1:
44862 0000F350 F6C401                  <1>       test   ah, 1
44863 0000F353 7511                    <1>       jnz    short int34h_3 ; out
44864                                  <1>
44865                                  <1>       ; in
44866 0000F355 80FC02                  <1>       cmp    ah, 2
44867 0000F358 7707                    <1>       ja     short int34h_2
44868                                  <1>
44869 0000F35A 6689D8                  <1>       mov    ax, bx
44870 0000F35D 66ED                    <1>       in     ax, dx
44871                                  <1>       ;iretd
```

```
44872 0000F35F EBEE              <1>       jmp    short int34h_iret
44873                            <1>
44874                            <1> int34h_2:
44875                            <1>       ; ah = 4
44876 0000F361 89D8              <1>       mov    eax, ebx
44877 0000F363 ED                <1>       in     eax, dx
44878                            <1>       ;iretd
44879 0000F364 EBE9              <1>       jmp    short int34h_iret
44880                            <1>
44881                            <1> int34h_3:
44882 0000F366 80FC03            <1>       cmp    ah, 3
44883 0000F369 7707              <1>       ja     short int34h_4
44884                            <1>
44885 0000F36B 6689D8            <1>       mov    ax, bx
44886 0000F36E 66EF              <1>       out    dx, ax
44887                            <1>       ;iretd
44888 0000F370 EBDD              <1>       jmp    short int34h_iret
44889                            <1>
44890                            <1> int34h_4:
44891                            <1>       ; ah = 5
44892 0000F372 89D8              <1>       mov    eax, ebx
44893 0000F374 EF                <1>       out    dx, eax
44894                            <1>       ;iretd
44895 0000F375 EBD8              <1>       jmp    short int34h_iret
44896                            <1>
44897                            <1> int34h_5:
44898 0000F377 804C240801        <1>       or     byte [esp+8], 1     ; set carry bit of eflags register
44899 0000F37C CF                <1>       iretd
44900                            <1>
44901                            <1> INT4Ah:
44902                            <1>       ; 24/01/2016
44903                            <1>       ; this procedure will be called by 'RTC_INT' (in 'timer.s')
44904 0000F37D C3                <1>       retn
44905                            <1>
44906                            <1> ; u0.s
44907                            <1> ; Retro UNIX 386 v1 Kernel (v0.2) - SYS0.INC
44908                            <1> ; Last Modification: 20/11/2015
44909                            <1>
44910                            <1> com2_int:
44911                            <1>       ; 07/11/2015
44912                            <1>       ; 24/10/2015
44913                            <1>       ; 23/10/2015
44914                            <1>       ; 14/03/2015 (Retro UNIX 386 v1 - Beginning)
44915                            <1>       ; 28/07/2014 (Retro UNIX 8086 v1)
44916                            <1>       ; < serial port 2 interrupt handler >
44917                            <1>       ;
44918 0000F37E 890424            <1>       mov    [esp], eax ; overwrite call return address
44919                            <1>       ;push  eax
44920 0000F381 66B80900          <1>       mov    ax, 9
44921 0000F385 EB07              <1>       jmp    short comm_int
44922                            <1> com1_int:
44923                            <1>       ; 07/11/2015
44924                            <1>       ; 24/10/2015
44925 0000F387 890424            <1>       mov    [esp], eax ; overwrite call return address
44926                            <1>       ; 23/10/2015
44927                            <1>       ;push  eax
44928 0000F38A 66B80800          <1>       mov    ax, 8
44929                            <1> comm_int:
44930                            <1>       ; 20/11/2015
44931                            <1>       ; 18/11/2015
44932                            <1>       ; 17/11/2015
44933                            <1>       ; 16/11/2015
44934                            <1>       ; 09/11/2015
44935                            <1>       ; 08/11/2015
44936                            <1>       ; 07/11/2015
44937                            <1>       ; 06/11/2015 (serial4.asm, 'serial')
44938                            <1>       ; 01/11/2015
44939                            <1>       ; 26/10/2015
44940                            <1>       ; 23/10/2015
44941 0000F38E 53                <1>       push   ebx
44942 0000F38F 56                <1>       push   esi
44943 0000F390 57                <1>       push   edi
44944 0000F391 1E                <1>       push   ds
44945 0000F392 06                <1>       push   es
44946                            <1>       ; 18/11/2015
44947 0000F393 0F20DB            <1>       mov    ebx, cr3
44948 0000F396 53                <1>       push   ebx ; ****
44949                            <1>       ;
44950 0000F397 51                <1>       push   ecx ; ***
44951 0000F398 52                <1>       push   edx ; **
44952                            <1>       ;
44953 0000F399 BB10000000        <1>       mov    ebx, KDATA
44954 0000F39E 8EDB              <1>       mov    ds, bx
44955 0000F3A0 8EC3              <1>       mov    es, bx
44956                            <1>       ;
44957 0000F3A2 8B0D[C84D0100]    <1>       mov    ecx, [k_page_dir]
44958 0000F3A8 0F22D9            <1>       mov    cr3, ecx
44959                            <1>       ; 20/11/2015
44960                            <1>       ; Interrupt identification register
44961 0000F3AB 66BAFA02          <1>       mov    dx, 2FAh ; COM2
44962                            <1>       ;
44963 0000F3AF 3C08              <1>       cmp    al, 8
44964 0000F3B1 7702              <1>       ja     short com_i0
44965                            <1>       ;
44966                            <1>       ; 20/11/2015
44967                            <1>       ; 17/11/2015
44968                            <1>       ; 16/11/2015
44969                            <1>       ; 15/11/2015
44970                            <1>       ; 24/10/2015
44971                            <1>       ; 14/03/2015 (Retro UNIX 386 v1 - Beginning)
44972                            <1>       ; 28/07/2014 (Retro UNIX 8086 v1)
44973                            <1>       ; < serial port 1 interrupt handler >
```

```
                                <1>       ;
0000F3B3 FEC6                   <1>       inc    dh ; 3FAh ; COM1 Interrupt id. register
                                <1> com_i0:
                                <1>       ;push  eax ; *
                                <1>       ; 07/11/2015
0000F3B5 A2[324E0100]           <1>       mov    byte [ccomport], al
                                <1>       ; 09/11/2015
0000F3BA 0FB7D8                 <1>       movzx  ebx, ax ; 8 or 9
                                <1>       ; 17/11/2015
                                <1>       ; reset request for response status
0000F3BD 88A3[284E0100]         <1>       mov    [ebx+req_resp-8], ah ; 0
                                <1>       ;
                                <1>       ; 20/11/2015
0000F3C3 EC                     <1>       in     al, dx       ; read interrupt id. register
0000F3C4 EB00                   <1>       JMP    $+2          ; I/O DELAY
0000F3C6 2404                   <1>       and    al, 4        ; received data available?
0000F3C8 7470                   <1>       jz     short com_eoi; (transmit. holding reg. empty)
                                <1>       ;
                                <1>       ; 20/11/2015
0000F3CA 80EA02                 <1>       sub    dl, 3FAh-3F8h; data register (3F8h, 2F8h)
0000F3CD EC                     <1>       in     al, dx       ; read character
                                <1>       ;JMP   $+2          ; I/O DELAY
                                <1>       ; 08/11/2015
                                <1>       ; 07/11/2015
0000F3CE 89DE                   <1>       mov    esi, ebx
0000F3D0 89DF                   <1>       mov    edi, ebx
0000F3D2 81C6[2C4E0100]         <1>       add    esi, rchar - 8 ; points to last received char
0000F3D8 81C7[2E4E0100]         <1>       add    edi, schar - 8 ; points to last sent char
0000F3DE 8806                   <1>       mov    [esi], al ; received char (current char)
                                <1>       ; query
0000F3E0 20C0                   <1>       and    al, al
0000F3E2 7527                   <1>       jnz    short com_i2
                                <1>       ; response
                                <1>       ; 17/11/2015
                                <1>       ; set request for response status
0000F3E4 FE83[284E0100]         <1>         inc     byte [ebx+req_resp-8] ; 1
                                <1>       ;
0000F3EA 6683C205               <1>       add    dx, 3FDh-3F8h; (3FDh, 2FDh)
0000F3EE EC                     <1>       in     al, dx       ; read line status register
0000F3EF EB00                   <1>       JMP    $+2          ; I/O DELAY
0000F3F1 2420                   <1>       and    al, 20h            ; transmitter holding reg. empty?
0000F3F3 7445                   <1>       jz     short com_eoi      ; no
0000F3F5 B0FF                   <1>       mov    al, 0FFh     ; response
0000F3F7 6683EA05               <1>       sub    dx, 3FDh-3F8h      ; data port (3F8h, 2F8h)
0000F3FB EE                     <1>       out    dx, al       ; send on serial port
                                <1>       ; 17/11/2015
0000F3FC 803F00                 <1>       cmp    byte [edi], 0   ; query ? (schar)
0000F3FF 7502                   <1>       jne    short com_i1    ; no
0000F401 8807                   <1>       mov    [edi], al    ; 0FFh (responded)
                                <1> com_i1:
                                <1>       ; 17/11/2015
                                <1>       ; reset request for response status (again)
0000F403 FE8B[284E0100]         <1>         dec     byte [ebx+req_resp-8] ; 0
0000F409 EB2F                   <1>       jmp    short com_eoi
                                <1> com_i2:
                                <1>       ; 08/11/2015
0000F40B 3CFF                   <1>       cmp    al, 0FFh     ; (response ?)
0000F40D 7417                   <1>       je     short com_i3 ; (check for response signal)
                                <1>       ; 07/11/2015
0000F40F 3C04                   <1>       cmp    al, 04h      ; EOT
0000F411 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)
0000F413 861D[F64D0100]         <1>       xchg   bl, [ptty]  ; tty number (8 or 9)
0000F419 E8B86FFFFF             <1>       call   ctrlbrk
0000F41E 861D[F64D0100]         <1>       xchg   [ptty], bl ; (restore ptty value and BL value)
                                <1>       ;mov   al, 04h ; EOT
                                <1>       ; 08/11/2015
0000F424 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
0000F426 803F00                 <1>         cmp     byte [edi], 0 ; query ? (schar)
0000F429 7704                   <1>       ja     short com_i4 ; no
                                <1>       ; reset query status (schar)
0000F42B 8807                   <1>       mov    [edi], al ; 0FFh
0000F42D FEC0                   <1>       inc    al ; 0
                                <1> com_i4:
                                <1>       ; 27/07/2014
                                <1>       ; 09/07/2014
0000F42F D0E3                   <1>       shl    bl, 1
0000F431 81C3[F84D0100]         <1>       add    ebx, ttychr
                                <1>       ; 23/07/2014 (always overwrite)
                                <1>       ;;cmp  word [ebx], 0
                                <1>       ;;ja   short com_eoi
                                <1>       ;
0000F437 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 ; *
0000F43A A0[324E0100]           <1>       mov    al, byte [ccomport] ; current COM port
                                <1>        ; al = tty number (8 or 9)
0000F43F E85E010000             <1>         call wakeup
                                <1> com_iret:
```

```
45076                                  <1>       ; 23/10/2015
45077 0000F444 5A                      <1>       pop    edx ; **
45078 0000F445 59                      <1>       pop    ecx ; ***
45079                                  <1>       ; 18/11/2015
45080                                  <1>       ;pop   eax ; ****
45081                                  <1>       ;mov   cr3, eax
45082                                  <1>       ;jmp   iiret
45083 0000F446 E99516FFFF              <1>       jmp    iiretp
45084                                  <1>
45085                                  <1> ;iiretp: ; 01/09/2015
45086                                  <1> ;     ; 28/08/2015
45087                                  <1> ;     pop    eax ; (*) page directory
45088                                  <1> ;     mov    cr3, eax
45089                                  <1> ;iiret:
45090                                  <1> ;     ; 22/08/2014
45091                                  <1> ;     mov    al, 20h ; END OF INTERRUPT COMMAND TO 8259
45092                                  <1> ;     out    20h, al      ; 8259 PORT
45093                                  <1> ;     ;
45094                                  <1> ;     pop    es
45095                                  <1> ;     pop    ds
45096                                  <1> ;     pop    edi
45097                                  <1> ;     pop    esi
45098                                  <1> ;     pop    ebx ; 29/08/2014
45099                                  <1> ;     pop    eax
45100                                  <1> ;     iretd
45101                                  <1>
45102                                  <1> sp_init:
45103                                  <1>       ; 07/11/2015
45104                                  <1>       ; 29/10/2015
45105                                  <1>       ; 26/10/2015
45106                                  <1>       ; 23/10/2015
45107                                  <1>       ; 29/06/2015
45108                                  <1>       ; 14/03/2015 (Retro UNIX 386 v1 - 115200 baud)
45109                                  <1>       ; 28/07/2014 (Retro UNIX 8086 v1 - 9600 baud)
45110                                  <1>       ; Initialization of Serial Port Communication Parameters
45111                                  <1>       ; (COM1 base port address = 3F8h, COM1 Interrupt = IRQ 4)
45112                                  <1>       ; (COM2 base port address = 2F8h, COM1 Interrupt = IRQ 3)
45113                                  <1>       ;
45114                                  <1>       ; ((Modified registers: EAX, ECX, EDX, EBX))
45115                                  <1>       ;
45116                                  <1>       ; INPUT:  (29/06/2015)
45117                                  <1>       ;     AL = 0 for COM1
45118                                  <1>       ;          1 for COM2
45119                                  <1>       ;     AH = Communication parameters
45120                                  <1>       ;
45121                                  <1>       ;  (*) Communication parameters (except BAUD RATE):
45122                                  <1>       ;     Bit    4      3      2      1      0
45123                                  <1>       ;            -PARITY--   STOP BIT  -WORD LENGTH-
45124                                  <1>       ;  this one -->     00 = none    0 = 1 bit  11 = 8 bits
45125                                  <1>       ;            01 = odd     1 = 2 bits   10 = 7 bits
45126                                  <1>       ;            11 = even
45127                                  <1>       ;  Baud rate setting bits: (29/06/2015)
45128                                  <1>       ;            Retro UNIX 386 v1 feature only !
45129                                  <1>       ;     Bit    7    6    5  | Baud rate
45130                                  <1>       ;            ------------------------
45131                                  <1>       ;     value  0    0    0  | Default (Divisor = 1)
45132                                  <1>       ;            0    0    1  | 9600 (12)
45133                                  <1>       ;            0    1    0  | 19200 (6)
45134                                  <1>       ;            0    1   1  | 38400 (3)
45135                                  <1>       ;            1    0   0  | 14400 (8)
45136                                  <1>       ;            1    0   1  | 28800 (4)
45137                                  <1>       ;            1    1    0  | 57600 (2)
45138                                  <1>       ;            1    1    1  | 115200 (1)
45139                                  <1>
45140                                  <1>       ; References:
45141                                  <1>       ; (1) IBM PC-XT Model 286 BIOS Source Code
45142                                  <1>       ;     RS232.ASM --- 10/06/1985 COMMUNICATIONS BIOS (RS232)
45143                                  <1>       ; (2) Award BIOS 1999 - ATORGS.ASM
45144                                  <1>       ; (3) http://wiki.osdev.org/Serial_Ports
45145                                  <1>       ;
45146                                  <1>       ; Set communication parameters for COM1 (= 03h)
45147                                  <1>       ;
45148 0000F44B BB[2E4E0100]            <1>       mov    ebx, com1p          ; COM1 parameters
45149 0000F450 66BAF803                <1>       mov    dx, 3F8h            ; COM1
45150                                  <1>        ; 29/10/2015
45151 0000F454 66B90103                <1>       mov    cx, 301h  ; divisor = 1 (115200 baud)
45152 0000F458 E86F000000              <1>       call   sp_i3  ; call A4
45153 0000F45D A880                    <1>       test   al, 80h
45154 0000F45F 7410                    <1>       jz     short sp_i0 ; OK..
45155                                  <1>              ; Error !
45156                                  <1>       ;mov   dx, 3F8h
45157 0000F461 80EA05                  <1>       sub    dl, 5 ; 3FDh -> 3F8h
45158 0000F464 66B90E03                <1>       mov    cx, 30Eh  ; divisor = 12 (9600 baud)
45159 0000F468 E85F000000              <1>       call   sp_i3  ; call A4
45160 0000F46D A880                    <1>       test   al, 80h
45161 0000F46F 7508                    <1>       jnz    short sp_i1
45162                                  <1> sp_i0:
45163                                  <1>         ; (Note: Serial port interrupts will be disabled here...)
45164                                  <1>         ; (INT 14h initialization code disables interrupts.)
45165                                  <1>       ;
45166 0000F471 C603E3                  <1>       mov    byte [ebx], 0E3h ; 11100011b
45167 0000F474 E8DC000000              <1>       call   sp_i5 ; 29/06/2015
45168                                  <1> sp_i1:
45169 0000F479 43                      <1>       inc    ebx
45170 0000F47A 66BAF802                <1>       mov    dx, 2F8h            ; COM2
45171                                  <1>        ; 29/10/2015
45172 0000F47E 66B90103                <1>       mov    cx, 301h  ; divisor = 1 (115200 baud)
45173 0000F482 E845000000              <1>       call   sp_i3  ; call A4
45174 0000F487 A880                    <1>       test   al, 80h
45175 0000F489 7410                    <1>       jz     short sp_i2 ; OK..
45176                                  <1>              ; Error !
45177                                  <1>       ;mov   dx, 2F8h
```

```
45178 0000F48B 80EA05              <1>       sub   dl, 5 ; 2FDh -> 2F8h
45179 0000F48E 66B90E03            <1>       mov   cx, 30Eh  ; divisor = 12 (9600 baud)
45180 0000F492 E835000000          <1>       call  sp_i3  ; call A4
45181 0000F497 A880                <1>       test  al, 80h
45182 0000F499 7530                <1>       jnz   short sp_i7
45183                              <1> sp_i2:
45184 0000F49B C603E3              <1>       mov   byte [ebx], 0E3h ; 11100011b
45185                              <1> sp_i6:
45186                              <1>       ;; COM2 - enabling IRQ 3
45187                              <1>       ; 07/11/2015
45188                              <1>       ; 26/10/2015
45189 0000F49E 9C                  <1>       pushf
45190 0000F49F FA                  <1>       cli
45191                              <1>       ;
45192 0000F4A0 66BAFC02            <1>       mov   dx, 2FCh          ; modem control register
45193 0000F4A4 EC                  <1>       in    al, dx                  ; read register
45194 0000F4A5 EB00                <1>       JMP   $+2               ; I/O DELAY
45195 0000F4A7 0C08                <1>       or    al, 8             ; enable bit 3 (OUT2)
45196 0000F4A9 EE                  <1>       out   dx, al            ; write back to register
45197 0000F4AA EB00                <1>       JMP   $+2               ; I/O DELAY
45198 0000F4AC 66BAF902            <1>       mov   dx, 2F9h          ; interrupt enable register
45199 0000F4B0 EC                  <1>       in    al, dx            ; read register
45200 0000F4B1 EB00                <1>       JMP   $+2               ; I/O DELAY
45201                              <1>       ;or   al, 1             ; receiver data interrupt enable and
45202 0000F4B3 0C03                <1>       or    al, 3             ; transmitter empty interrupt enable
45203 0000F4B5 EE                  <1>       out   dx, al                  ; write back to register
45204 0000F4B6 EB00                <1>       JMP   $+2               ; I/O DELAY
45205 0000F4B8 E421                <1>       in    al, 21h           ; read interrupt mask register
45206 0000F4BA EB00                <1>       JMP   $+2               ; I/O DELAY
45207 0000F4BC 24F7                <1>       and   al, 0F7h          ; enable IRQ 3 (COM2)
45208 0000F4BE E621                <1>       out   21h, al           ; write back to register
45209                              <1>       ;
45210                              <1>       ; 23/10/2015
45211 0000F4C0 B8[7EF30000]        <1>       mov   eax, com2_int
45212 0000F4C5 A3[9DF50000]        <1>       mov   [com2_irq3], eax
45213                              <1>       ; 26/10/2015
45214 0000F4CA 9D                  <1>       popf
45215                              <1> sp_i7:
45216 0000F4CB C3                  <1>       retn
45217                              <1>
45218                              <1> sp_i3:
45219                              <1> ;A4:        ;----- INITIALIZE THE COMMUNICATIONS PORT
45220                              <1>       ; 28/10/2015
45221 0000F4CC FEC2                <1>       inc   dl    ; 3F9h (2F9h); 3F9h, COM1 Interrupt enable register
45222 0000F4CE B000                <1>       mov   al, 0
45223 0000F4D0 EE                  <1>       out   dx, al            ; disable serial port interrupt
45224 0000F4D1 EB00                <1>       JMP   $+2               ; I/O DELAY
45225 0000F4D3 80C202              <1>       add   dl, 2  ; 3FBh (2FBh); COM1 Line control register (3FBh)
45226 0000F4D6 B080                <1>       mov   al, 80h
45227 0000F4D8 EE                  <1>       out   dx, al            ; SET DLAB=1 ; divisor latch access bit
45228                              <1>       ;----- SET BAUD RATE DIVISOR
45229                              <1>       ; 26/10/2015
45230 0000F4D9 80EA03              <1>       sub   dl, 3   ; 3F8h (2F8h)     ; register for least significant byte
45231                              <1>                                 ; of the divisor value
45232 0000F4DC 88C8                <1>       mov   al, cl ; 1
45233 0000F4DE EE                  <1>       out   dx, al            ; 1 = 115200 baud (Retro UNIX 386 v1)
45234                              <1>                                 ; 2 = 57600 baud
45235                              <1>                                 ; 3 = 38400 baud
45236                              <1>                                 ; 6 = 19200 baud
45237                              <1>                                 ; 12 = 9600 baud (Retro UNIX 8086 v1)
45238 0000F4DF EB00                <1>       JMP   $+2               ; I/O DELAY
45239 0000F4E1 28C0                <1>       sub   al, al
45240 0000F4E3 FEC2                <1>       inc   dl      ; 3F9h (2F9h)     ; register for most significant byte
45241                              <1>                                 ; of the divisor value
45242 0000F4E5 EE                  <1>       out   dx, al ; 0
45243 0000F4E6 EB00                <1>       JMP   $+2               ; I/O DELAY
45244                              <1>       ;
45245 0000F4E8 88E8                <1>       mov   al, ch ; 3        ; 8 data bits, 1 stop bit, no parity
45246                              <1>       ;and  al, 1Fh ; Bits 0,1,2,3,4
45247 0000F4EA 80C202              <1>       add   dl, 2  ; 3FBh (2FBh); Line control register
45248 0000F4ED EE                  <1>       out   dx, al
45249 0000F4EE EB00                <1>       JMP   $+2               ; I/O DELAY
45250                              <1>       ; 29/10/2015
45251 0000F4F0 FECA                <1>       dec   dl    ; 3FAh (2FAh); FIFO Control register (16550/16750)
45252 0000F4F2 30C0                <1>       xor   al, al            ; 0
45253 0000F4F4 EE                  <1>       out   dx, al            ; Disable FIFOs (reset to 8250 mode)
45254 0000F4F5 EB00                <1>       JMP   $+2
45255                              <1> sp_i4:
45256                              <1> ;A18: ;----- COMM PORT STATUS ROUTINE
45257                              <1>       ; 29/06/2015 (line status after modem status)
45258 0000F4F7 80C204              <1>       add   dl, 4  ; 3FEh (2FEh); Modem status register
45259                              <1> sp_i4s:
45260 0000F4FA EC                  <1>       in    al, dx            ; GET MODEM CONTROL STATUS
45261 0000F4FB EB00                <1>       JMP   $+2               ; I/O DELAY
45262 0000F4FD 88C4                <1>       mov   ah, al            ; PUT IN (AH) FOR RETURN
45263 0000F4FF FECA                <1>       dec   dl    ; 3FDh (2FDh); POINT TO LINE STATUS REGISTER
45264                              <1>                                 ; dx = 3FDh for COM1, 2FDh for COM2
45265 0000F501 EC                  <1>       in    al, dx            ; GET LINE CONTROL STATUS
45266                              <1>       ; AL = Line status, AH = Modem status
45267 0000F502 C3                  <1>       retn
45268                              <1>
45269                              <1> sp_status:
45270                              <1>       ; 29/06/2015
45271                              <1>       ; 27/06/2015 (Retro UNIX 386 v1)
45272                              <1>       ; Get serial port status
45273 0000F503 66BAFE03            <1>       mov   dx, 3FEh          ; Modem status register (COM1)
45274 0000F507 28C6                <1>       sub   dh, al            ; dh = 2 for COM2 (al = 1)
45275                              <1>                                 ; dx = 2FEh for COM2
45276 0000F509 EBEF                <1>       jmp   short sp_i4s
45277                              <1>
45278                              <1> sp_setp: ; Set serial port communication parameters
45279                              <1>       ; 07/11/2015
```

```
45280                                  <1>       ; 29/10/2015
45281                                  <1>       ; 29/06/2015
45282                                  <1>       ; Retro UNIX 386 v1 feature only !
45283                                  <1>       ;
45284                                  <1>       ; INPUT:
45285                                  <1>       ;     AL = 0 for COM1
45286                                  <1>       ;          1 for COM2
45287                                  <1>       ;     AH = Communication parameters (*)
45288                                  <1>       ; OUTPUT:
45289                                  <1>       ;     CL = Line status
45290                                  <1>       ;     CH = Modem status
45291                                  <1>       ;   If cf = 1 -> Error code in [u.error]
45292                                  <1>       ;             'invalid parameter !'
45293                                  <1>       ;                   or
45294                                  <1>       ;             'device not ready !' error
45295                                  <1>       ;
45296                                  <1>       ;  (*) Communication parameters (except BAUD RATE):
45297                                  <1>       ;     Bit    4     3     2     1     0
45298                                  <1>       ;             -PARITY--   STOP BIT  -WORD LENGTH-
45299                                  <1>       ;  this one -->    00 = none    0 = 1 bit  11 = 8 bits
45300                                  <1>       ;             01 = odd     1 = 2 bits   10 = 7 bits
45301                                  <1>       ;             11 = even
45302                                  <1>       ;  Baud rate setting bits: (29/06/2015)
45303                                  <1>       ;             Retro UNIX 386 v1 feature only !
45304                                  <1>       ;     Bit    7    6    5  | Baud rate
45305                                  <1>       ;             ------------------------
45306                                  <1>       ;     value  0    0    0  | Default (Divisor = 1)
45307                                  <1>       ;            0    0    1  | 9600 (12)
45308                                  <1>       ;            0    1    0  | 19200 (6)
45309                                  <1>       ;            0    1   1  | 38400 (3)
45310                                  <1>       ;            1    0   0  | 14400 (8)
45311                                  <1>       ;            1    0   1  | 28800 (4)
45312                                  <1>       ;            1    1    0  | 57600 (2)
45313                                  <1>       ;            1    1    1  | 115200 (1)
45314                                  <1>       ;
45315                                  <1>       ; (COM1 base port address = 3F8h, COM1 Interrupt = IRQ 4)
45316                                  <1>       ; (COM2 base port address = 2F8h, COM1 Interrupt = IRQ 3)
45317                                  <1>       ;
45318                                  <1>       ; ((Modified registers: EAX, ECX, EDX, EBX))
45319                                  <1>       ;
45320 0000F50B 66BAF803                <1>       mov   dx, 3F8h
45321 0000F50F BB[2E4E0100]            <1>       mov   ebx, com1p ; COM1 control byte offset
45322 0000F514 3C01                    <1>       cmp   al, 1
45323 0000F516 776B                    <1>       ja    short sp_invp_err
45324 0000F518 7203                    <1>       jb    short sp_setp1 ;  COM1 (AL = 0)
45325 0000F51A FECE                    <1>       dec   dh ; 2F8h
45326 0000F51C 43                      <1>       inc   ebx ; COM2 control byte offset
45327                                  <1> sp_setp1:
45328                                  <1>       ; 29/10/2015
45329 0000F51D 8823                    <1>       mov   [ebx], ah
45330 0000F51F 0FB6CC                  <1>       movzx ecx, ah
45331 0000F522 C0E905                  <1>       shr   cl, 5 ; -> baud rate index
45332 0000F525 80E41F                  <1>       and   ah, 1Fh ; communication parameters except baud rate
45333 0000F528 8A81[92F50000]          <1>       mov   al, [ecx+b_div_tbl]
45334 0000F52E 6689C1                  <1>       mov   cx, ax
45335 0000F531 E896FFFFFF              <1>       call  sp_i3
45336 0000F536 6689C1                  <1>       mov   cx, ax ; CL = Line status, CH = Modem status
45337 0000F539 A880                    <1>       test  al, 80h
45338 0000F53B 740F                    <1>       jz    short sp_setp2
45339 0000F53D C603E3                  <1>         mov     byte [ebx], 0E3h ; Reset to initial value (11100011b)
45340                                  <1> stp_dnr_err:
45341 0000F540 C705[C8030600]0F00-     <1>       mov   dword [u.error], ERR_DEV_NOT_RDY ; 'device not ready !'
45342 0000F548 0000                    <1>
45343                                  <1>       ; CL = Line status, CH = Modem status
45344 0000F54A F9                      <1>       stc
45345 0000F54B C3                      <1>       retn
45346                                  <1> sp_setp2:
45347 0000F54C 80FE02                  <1>       cmp   dh, 2 ; COM2 (2F?h)
45348 0000F54F 0F8649FFFFFF            <1>         jna     sp_i6
45349                                  <1>                     ; COM1 (3F?h)
45350                                  <1> sp_i5:
45351                                  <1>       ; 07/11/2015
45352                                  <1>       ; 26/10/2015
45353                                  <1>       ; 29/06/2015
45354                                  <1>       ;
45355                                  <1>       ;; COM1 - enabling IRQ 4
45356 0000F555 9C                      <1>       pushf
45357 0000F556 FA                      <1>       cli
45358 0000F557 66BAFC03                <1>       mov   dx, 3FCh            ; modem control register
45359 0000F55B EC                      <1>       in    al, dx                    ; read register
45360 0000F55C EB00                    <1>       JMP   $+2                 ; I/O DELAY
45361 0000F55E 0C08                    <1>       or    al, 8               ; enable bit 3 (OUT2)
45362 0000F560 EE                      <1>       out   dx, al              ; write back to register
45363 0000F561 EB00                    <1>       JMP   $+2                 ; I/O DELAY
45364 0000F563 66BAF903                <1>       mov   dx, 3F9h            ; interrupt enable register
45365 0000F567 EC                      <1>       in    al, dx              ; read register
45366 0000F568 EB00                    <1>       JMP   $+2                 ; I/O DELAY
45367                                  <1>       ;or   al, 1               ; receiver data interrupt enable and
45368 0000F56A 0C03                    <1>       or    al, 3               ; transmitter empty interrupt enable
45369 0000F56C EE                      <1>       out   dx, al                    ; write back to register
45370 0000F56D EB00                    <1>       JMP   $+2                 ; I/O DELAY
45371 0000F56F E421                    <1>       in    al, 21h             ; read interrupt mask register
45372 0000F571 EB00                    <1>       JMP   $+2                 ; I/O DELAY
45373 0000F573 24EF                    <1>       and   al, 0EFh            ; enable IRQ 4 (COM1)
45374 0000F575 E621                    <1>       out   21h, al             ; write back to register
45375                                  <1>       ;
45376                                  <1>       ; 23/10/2015
45377 0000F577 B8[87F30000]            <1>       mov   eax, com1_int
45378 0000F57C A3[99F50000]            <1>       mov   [com1_irq4], eax
45379                                  <1>       ; 26/10/2015
45380 0000F581 9D                      <1>       popf
45381 0000F582 C3                      <1>       retn
```

```
45382                                  <1>
45383                                  <1> sp_invp_err:
45384 0000F583 C705[C8030600]1700-      <1>       mov    dword [u.error], ERR_INV_PARAMETER ; 'invalid parameter !'
45385 0000F58B 0000                    <1>
45386 0000F58D 31C9                    <1>       xor    ecx, ecx
45387 0000F58F 49                      <1>       dec    ecx ; 0FFFFh
45388 0000F590 F9                      <1>       stc
45389 0000F591 C3                      <1>       retn
45390                                  <1>
45391                                  <1> ; 29/10/2015
45392                                  <1> b_div_tbl: ; Baud rate divisor table (115200/divisor)
45393 0000F592 010C0603080401          <1>       db 1, 12, 6, 3, 8, 4, 1
45394                                  <1>
45395                                  <1>
45396                                  <1> ; 23/10/2015
45397                                  <1> com1_irq4:
45398 0000F599 [A1F50000]              <1>       dd dummy_retn
45399                                  <1> com2_irq3:
45400 0000F59D [A1F50000]              <1>       dd dummy_retn
45401                                  <1>
45402                                  <1> dummy_retn:
45403 0000F5A1 C3                      <1>       retn
45404                                  <1>
45405                                  <1> wakeup:
45406                                  <1>       ; 24/01/2016
45407 0000F5A2 C3                      <1>       retn
45408                                  <1>
45409                                  <1> set_working_path_x:
45410                                  <1>             ; 17/10/2016 (TRDOS 386 - FFF & FNF)
45411 0000F5A3 66B80100                <1>             mov    ax, 1
45412                                  <1>                    ; File name is needed/forced (AL=1)
45413                                  <1>                    ; Change directory as temprorary (AH=0)
45414                                  <1>
45415                                  <1>             ; This is needed for preventing wrong Find Next File
45416                                  <1>             ; system call after sysopen, syscreate, sysmkdir etc.
45417                                  <1>             ; Find Next File must immmediate follow Find First file)
45418                                  <1>
45419 0000F5A7 8825[805B0100]          <1>             mov    [FFF_Valid], ah ; 0 ; reset ; 17/10/2016
45420                                  <1>
45421                                  <1> set_working_path:
45422                                  <1>             ; 16/10/2016
45423                                  <1>             ; 12/10/2016
45424                                  <1>             ; 10/10/2016
45425                                  <1>             ; 05/10/2016 - TRDOS 386 (TRDOS v2.0)
45426                                  <1>             ;
45427                                  <1>             ; TRDOS v1.0 (DIR.ASM, "proc_set_working_path")
45428                                  <1>                ; 27/01/2011 - 08/02/2011
45429                                  <1>             ; Set/Changes current drive, directory and file
45430                                  <1>             ; depending on command tail
45431                                  <1>             ; (procedure is derivated from CMD_INTR.ASM
45432                                  <1>             ; file or dir locating code of internal commands)
45433                                  <1>             ; (This procedure is prepared for INT 21H file/dir
45434                                  <1>             ; functions and also to get compact code for
45435                                  <1>             ; internal mainprog -command interpreter- commands)
45436                                  <1>             ;
45437                                  <1>             ; INPUT: DS:SI -> Command tail (ASCIIZ string)
45438                                  <1>             ; AL= 0 -> any, AL > 0 -> file name is forced
45439                                  <1>             ; AH= CD -> Change directory permanently
45440                                  <1>             ; AH <> CD -> Change directory as temporary
45441                                  <1>             ;
45442                                  <1>             ; OUTPUT: ES=DS, FindFile structure has been set
45443                                  <1>             ;       RUN_CDRV points previous current drive
45444                                  <1>             ;       DS:SI = FindFile structure address
45445                                  <1>             ;       (DS=CS)
45446                                  <1>             ;       AX, BX, CX, DX, DI will be changed
45447                                  <1>             ;   cf = 1 -> Error code in AX (AL)
45448                                  <1>             ;       stc & AX = 0 -> Bad command or path name
45449                                  <1>             ; -----------------------------------------------
45450                                  <1>             ;
45451                                  <1>             ; TRDOS 386 (05/10/2016)
45452                                  <1>             ; INPUT:
45453                                  <1>             ;     ESI = File/Directory Path (ASCIIZ string)
45454                                  <1>             ;             address in user's memory space
45455                                  <1>             ;      Al = 0 -> any
45456                                  <1>             ;      AL > 0 -> file name is forced
45457                                  <1>             ;      AH = CD -> change directory as permanent
45458                                  <1>             ;      AH <> CD -> change directory as temporary
45459                                  <1>             ;
45460                                  <1>             ; OUTPUT:
45461                                  <1>             ;     FindFile structure has been set
45462                                  <1>             ;      RUN_CDRV points previous current drive
45463                                  <1>             ;      ESI = FindFile_Name address ; 12/10/2016
45464                                  <1>             ;
45465                                  <1>             ;      cf = 1 -> Error code in EAX (AL)
45466                                  <1>             ;      stc & EAX = 0 -> Bad command or path name
45467                                  <1>             ;
45468                                  <1>             ; Modified registers: EAX, EBX, ECX, EDX, ESI, EDI
45469                                  <1>
45470 0000F5AD 66A3[845B0100]          <1>             mov    [SWP_Mode], ax
45471 0000F5B3 A0[8E4E0100]            <1>             mov    al, [Current_Drv]
45472 0000F5B8 30E4                    <1>             xor    ah, ah
45473 0000F5BA 66A3[865B0100]          <1>             mov    word [SWP_DRV], ax
45474                                  <1>
45475                                  <1>             ; TRDOS 386 ring 3 (user's page directory)
45476                                  <1>             ; to ring 0 (kernel's page directory)
45477                                  <1>             ; transfer modifications (05/10/2016).
45478                                  <1>
45479 0000F5C0 55                      <1>             push   ebp
45480 0000F5C1 89E5                    <1>             mov    ebp, esp
45481                                  <1>
45482 0000F5C3 B980000000              <1>             mov    ecx, 128 ; maximum path length = 128 bytes
45483 0000F5C8 29CC                    <1>             sub    esp, ecx ; reserve 128 bytes (buffer) on stack
```

```
45484 0000F5CA 89E7              <1>             mov    edi, esp ; destination address (kernel space)
45485                            <1>             ; esi = source address (virtual, in user's memory space)
45486 0000F5CC E8FBF2FFFF        <1>             call   transfer_from_user_buffer
45487 0000F5D1 720A              <1>             jc     short loc_swp_xor_retn
45488                            <1>
45489 0000F5D3 89E6              <1>             mov    esi, esp ; temporary buffer (the path) on stack
45490                            <1> loc_swp_fchar:
45491 0000F5D5 8A06              <1>             mov    al, [esi]
45492 0000F5D7 3C20              <1>             cmp    al, 20h
45493 0000F5D9 7711              <1>             ja     short loc_swp_parse_path_name
45494 0000F5DB 740C              <1>             je     short loc_swp_fchar_next
45495                            <1>
45496                            <1> loc_swp_xor_retn:
45497 0000F5DD 31C0              <1>             xor    eax, eax
45498 0000F5DF F9                <1>             stc
45499                            <1> loc_swp_retn:
45500 0000F5E0 89EC              <1>             mov    esp, ebp
45501 0000F5E2 5D                <1>             pop    ebp
45502                            <1>
45503                            <1>             ;mov   esi, FindFile_Drv
45504 0000F5E3 BE[74580100]      <1>             mov    esi, FindFile_Name ; 12/10/2016
45505 0000F5E8 C3                <1>             retn
45506                            <1>
45507                            <1> loc_swp_fchar_next:
45508 0000F5E9 46                <1>             inc    esi
45509 0000F5EA EBE9              <1>             jmp    short loc_swp_fchar
45510                            <1>
45511                            <1> loc_swp_parse_path_name:
45512 0000F5EC BF[32580100]      <1>             mov    edi, FindFile_Drv
45513 0000F5F1 E810ACFFFF        <1>             call   parse_path_name
45514 0000F5F6 72E8              <1>             jc     short loc_swp_retn
45515                            <1>
45516                            <1> loc_swp_checkfile_name:
45517 0000F5F8 803D[845B0100]00  <1>             cmp    byte [SWP_Mode], 0
45518 0000F5FF 761E              <1>             jna    short loc_swp_drv
45519                            <1>
45520                            <1>             ; 10/10/2016 (valid file name checking)
45521 0000F601 BE[74580100]      <1>             mov    esi, FindFile_Name
45522 0000F606 803E20            <1>             cmp    byte [esi], 20h
45523 0000F609 76D2              <1>             jna    short loc_swp_xor_retn
45524                            <1>
45525                            <1>             ; 16/10/2016
45526 0000F60B C605[835B0100]00  <1>             mov    byte [SWP_inv_fname], 0 ; reset
45527                            <1>             ; esi = file name address (ASCIIZ)
45528 0000F612 E8988DFFFF        <1>             call   check_filename
45529 0000F617 7306              <1>             jnc    short loc_swp_drv
45530                            <1>
45531 0000F619 FE05[835B0100]    <1>             inc    byte [SWP_inv_fname] ; set
45532                            <1> loc_swp_drv:
45533 0000F61F 8A35[8E4E0100]    <1>             mov    dh, [Current_Drv]
45534                            <1>                ;mov      [RUN_CDRV], dh
45535                            <1>
45536 0000F625 8A15[32580100]    <1>             mov    dl, [FindFile_Drv]
45537                            <1>                ;cmp      dl, dh
45538 0000F62B 3A15[8E4E0100]    <1>             cmp    dl, [Current_Drv]
45539 0000F631 740D              <1>             je     short loc_swp_change_directory
45540                            <1>
45541 0000F633 FE05[875B0100]    <1>             inc    byte [SWP_DRV_chg]
45542 0000F639 E81F76FFFF        <1>             call   change_current_drive
45543 0000F63E 72A0              <1>             jc     short loc_swp_retn ; eax = error code
45544                            <1>             ; eax = 0
45545                            <1>
45546                            <1> loc_swp_change_directory:
45547 0000F640 803D[33580100]21  <1>             cmp    byte [FindFile_Directory], 21h
45548 0000F647 F5                <1>             cmc
45549 0000F648 7396              <1>             jnc    short loc_swp_retn
45550                            <1>
45551 0000F64A FE05[875B0100]    <1>             inc    byte [SWP_DRV_chg]
45552 0000F650 FE05[98020100]    <1>             inc    byte [Restore_CDIR]
45553 0000F656 BE[33580100]      <1>             mov    esi, FindFile_Directory
45554 0000F65B 8A25[855B0100]    <1>             mov    ah, [SWP_Mode+1]
45555 0000F661 E88CA5FFFF        <1>             call   change_current_directory
45556 0000F666 0F8274FFFFFF      <1>             jc     loc_swp_retn ; eax = error code
45557                            <1>
45558                            <1> loc_swp_change_prompt_dir_string:
45559                            <1>             ; esi = PATH_Array
45560                            <1>             ; eax = Current Directory First Cluster
45561                            <1>             ; edi = Logical DOS Drive Description Table
45562 0000F66C E8A6A4FFFF        <1>             call   change_prompt_dir_str
45563 0000F671 29C0              <1>             sub    eax, eax ; 0
45564 0000F673 E968FFFFFF        <1>             jmp    loc_swp_retn
45565                            <1>
45566                            <1> reset_working_path:
45567                            <1>             ; 06/10/2016 - TRDOS 386 (TRDOS v2.0)
45568                            <1>             ;
45569                            <1>             ; TRDOS v1.0 (DIR.ASM, "proc_reset_working_path")
45570                            <1>             ; 05/02/2011 - 08/02/2011
45571                            <1>             ;
45572                            <1>             ; Restores current drive and directory
45573                            <1>             ;
45574                            <1>             ; INPUT: none
45575                            <1>             ; OUTPUT: DL = SWP_DRV, EAX = 0 -> OK
45576                            <1>             ;
45577                            <1>             ;    AX = 0 -> ESI = Logical Dos Drv Desc. Table
45578                            <1>             ;
45579                            <1>             ;    EAX, EBX, ECX, EDX, ESI, EDI will be changed
45580                            <1>             ;
45581                            <1>
45582                            <1>
45583 0000F678 31C0              <1>             xor    eax, eax
45584 0000F67A 48                <1>             dec    eax
45585                            <1>
```

```
45586 0000F67B 668B15[865B0100]    <1>             mov    dx, [SWP_DRV]
45587 0000F682 08F6                <1>             or     dh, dh
45588 0000F684 742E                <1>             jz     short loc_rwp_return
45589                              <1>
45590 0000F686 3A15[8E4E0100]      <1>             cmp    dl, [Current_Drv]
45591 0000F68C 7407                <1>             je     short loc_rwp_restore_cdir
45592                              <1> loc_rwp_restore_cdrv:
45593 0000F68E E8CA75FFFF          <1>             call   change_current_drive
45594 0000F693 EB10                <1>             jmp    short loc_rwp_restore_ok
45595                              <1> loc_rwp_restore_cdir:
45596 0000F695 31DB                <1>             xor    ebx, ebx
45597 0000F697 88D7                <1>             mov    bh, dl
45598 0000F699 BE00010900          <1>             mov    esi, Logical_DOSDisks
45599 0000F69E 01DE                <1>             add    esi, ebx
45600                              <1>
45601 0000F6A0 E86F76FFFF          <1>             call   restore_current_directory
45602                              <1>
45603                              <1> loc_rwp_restore_ok:
45604 0000F6A5 668B15[865B0100]    <1>             mov    dx, [SWP_DRV]
45605 0000F6AC 31C0                <1>             xor    eax, eax
45606 0000F6AE 66A3[875B0100]      <1>             mov    [SWP_DRV_chg], ax
45607                              <1> loc_rwp_return:
45608 0000F6B4 C3                  <1>             retn
45609                              <1>
45610                              <1> get_file_name:
45611                              <1>             ; 15/10/2016 - TRDOS 386 (TRDOS v2.0)
45612                              <1>             ; Convert file name
45613                              <1>             ;      from directory entry format
45614                              <1>                ;  to (8.3) dot file name format
45615                              <1>             ;
45616                              <1>             ; TRDOS v1.0 (DIR.ASM, "get_file_name")
45617                              <1>                ; 2005 - 09/10/2011
45618                              <1>             ; INPUT:
45619                              <1>             ;      DS:SI -> Directory Entry Format File Name
45620                              <1>             ;       ES:DI -> DOS Dot File Name Address
45621                              <1>             ; OUTPUT:
45622                              <1>             ;      DS:SI -> DOS Dot File Name Address
45623                              <1>                ;  ES:DI -> Directory Entry Format File Name
45624                              <1>             ;
45625                              <1>             ; TRDOS 386 (15/10/2016)
45626                              <1>             ; INPUT:
45627                              <1>             ;      ESI = File name addr in dir entry format
45628                              <1>             ;      EDI = Dot file name address (destination)
45629                              <1>             ; OUTPUT:
45630                              <1>             ;      File name is converted and moved
45631                              <1>             ;      to destination (as 8.3 dot filename)
45632                              <1>             ;
45633                              <1>             ; Modified registers: EAX, ECX
45634                              <1>
45635                              <1>                ; 2005 (TRDOS 8086) - 2016 (TRDOS 386)
45636                              <1>
45637 0000F6B5 57                  <1>             push   edi
45638 0000F6B6 56                  <1>             push   esi
45639 0000F6B7 AC                  <1>             lodsb
45640 0000F6B8 3C20                <1>             cmp    al, 20h
45641 0000F6BA 762A                <1>             jna    short pass_gfn_ext
45642 0000F6BC 56                  <1>             push   esi
45643 0000F6BD AA                  <1>             stosb
45644 0000F6BE B907000000          <1>             mov    ecx, 7
45645                              <1> loc_gfn_next_char:
45646 0000F6C3 AC                  <1>             lodsb
45647 0000F6C4 3C20                <1>             cmp    al, 20h
45648 0000F6C6 7603                <1>             jna    short pass_gfn_fn
45649 0000F6C8 AA                  <1>             stosb
45650 0000F6C9 E2F8                <1>             loop   loc_gfn_next_char
45651                              <1> pass_gfn_fn:
45652 0000F6CB 5E                  <1>             pop    esi
45653 0000F6CC 83C607              <1>             add    esi, 7
45654 0000F6CF AC                  <1>             lodsb
45655 0000F6D0 3C20                <1>             cmp    al, 20h
45656 0000F6D2 7612                <1>             jna    short pass_gfn_ext
45657 0000F6D4 B42E                <1>             mov    ah, '.'
45658 0000F6D6 86E0                <1>             xchg   ah, al
45659 0000F6D8 66AB                <1>             stosw
45660 0000F6DA AC                  <1>             lodsb
45661 0000F6DB 3C20                <1>             cmp    al, 20h
45662 0000F6DD 7607                <1>             jna    short pass_gfn_ext
45663 0000F6DF AA                  <1>             stosb
45664 0000F6E0 AC                  <1>             lodsb
45665 0000F6E1 3C20                <1>             cmp    al, 20h
45666 0000F6E3 7601                <1>             jna    short pass_gfn_ext
45667 0000F6E5 AA                  <1>             stosb
45668                              <1> pass_gfn_ext:
45669 0000F6E6 30C0                <1>             xor    al, al
45670 0000F6E8 AA                  <1>             stosb
45671 0000F6E9 5E                  <1>             pop    esi
45672 0000F6EA 5F                  <1>             pop    edi
45673 0000F6EB C3                  <1>             retn
45674                              <1>
45675                              <1> set_hardware_int_vector:
45676                              <1>             ; 18/03/2017
45677                              <1>             ; 03/03/2017
45678                              <1>             ; 28/02/2017 - TRDOS 386 (TRDOS v2.0)
45679                              <1>             ;
45680                              <1>             ; SET/RESET HARDWARE INTERRUPT GATE
45681                              <1>             ;
45682                              <1>             ; Changes interrupt gate descriptor table
45683                              <1>             ; (without changing default interrupt list)
45684                              <1>             ;
45685                              <1>             ; INPUT:
45686                              <1>             ;      AL = IRQ number (0 to 15)
45687                              <1>             ;      AH > 0 -> set
```

```
45688                                  <1>             ;      AH = 0 -> reset
45689                                  <1>             ;
45690                                  <1>             ; Modified registers: eax, ebx, edx, edi
45691                                  <1>             ;
45692                                  <1>
45693 0000F6EC C0E002                  <1>             shl    al, 2 ; IRQ number * 4
45694 0000F6EF 0FB6D8                  <1>             movzx  ebx, al
45695                                  <1>
45696 0000F6F2 08E4                    <1>             or     ah, ah
45697 0000F6F4 7508                    <1>             jnz    short shintv_1 ; set (for user call service)
45698                                  <1>
45699                                  <1>             ; 18/03/2017
45700 0000F6F6 81C3[640C0100]          <1>             add    ebx, IRQ_list ; reset to default interrupt list
45701 0000F6FC EB06                    <1>             jmp    short shintv_2
45702                                  <1> shintv_1:
45703 0000F6FE 81C3[25F70000]          <1>             add    ebx, IRQ_u_list
45704                                  <1> shintv_2:
45705 0000F704 8B13                    <1>             mov    edx, [ebx] ; IRQ handler address
45706                                  <1>
45707                                  <1>             ; 03/03/2017
45708 0000F706 D0E0                    <1>             shl    al, 1 ; IRQ number * 8
45709                                  <1>             ; 18/03/2017
45710 0000F708 0FB6F8                  <1>             movzx  edi, al
45711 0000F70B 81C7[E04B0100]          <1>             add    edi, idt + (8*32) ; IRQ 0 offset = idt + 256
45712                                  <1>
45713 0000F711 89D0                    <1>             mov    eax, edx ; IRQ handler address
45714 0000F713 BB00000800              <1>             mov    ebx, 80000h
45715                                  <1>
45716                                  <1>             ;mov   edx, eax
45717 0000F718 66BA008E                <1>             mov    dx, 8E00h
45718 0000F71C 6689C3                  <1>             mov    bx, ax
45719 0000F71F 89D8                    <1>             mov    eax, ebx ; /* selector = 0x0008 = cs */
45720                                  <1>                                     ; /* interrupt gate - dpl=0, present */
45721 0000F721 AB                      <1>             stosd  ; selector & offset bits 0-15
45722 0000F722 8917                    <1>             mov    [edi], edx ; attributes & offset bits 16-23
45723                                  <1>
45724 0000F724 C3                      <1>             retn
45725                                  <1> IRQ_u_list:
45726                                  <1>             ; 28/02/2017
45727 0000F725 [8B060000]              <1>             dd     timer_int
45728 0000F729 [FF0D0000]              <1>             dd     kb_int
45729 0000F72D [6D080000]              <1>             dd     irq2
45730 0000F731 [65F70000]              <1>             dd     IRQ_service3
45731 0000F735 [6FF70000]              <1>             dd     IRQ_service4
45732 0000F739 [79F70000]              <1>             dd     IRQ_service5
45733 0000F73D [B0410000]              <1>             dd     fdc_int
45734 0000F741 [83F70000]              <1>             dd     IRQ_service7
45735 0000F745 [F6070000]              <1>             dd     rtc_int
45736 0000F749 [8DF70000]              <1>             dd     IRQ_service9
45737 0000F74D [97F70000]              <1>             dd     IRQ_service10
45738 0000F751 [A1F70000]              <1>             dd     IRQ_service11
45739 0000F755 [ABF70000]              <1>             dd     IRQ_service12
45740 0000F759 [B5F70000]              <1>             dd     IRQ_service13
45741 0000F75D [2C4B0000]              <1>             dd     hdc1_int
45742 0000F761 [534B0000]              <1>             dd     hdc2_int
45743                                  <1>
45744                                  <1>             ; 03/03/2017
45745                                  <1>             ; 27/02/2017
45746                                  <1> IRQ_service3:
45747 0000F765 36C605[4A610100]03      <1>             mov    byte [ss:IRQnum], 3
45748 0000F76D EB4E                    <1>             jmp    short IRQ_service
45749                                  <1> IRQ_service4:
45750 0000F76F 36C605[4A610100]04      <1>             mov    byte [ss:IRQnum], 4
45751 0000F777 EB44                    <1>             jmp    short IRQ_service
45752                                  <1> IRQ_service5:
45753 0000F779 36C605[4A610100]05      <1>             mov    byte [ss:IRQnum], 5
45754 0000F781 EB3A                    <1>             jmp    short IRQ_service
45755                                  <1> IRQ_service7:
45756 0000F783 36C605[4A610100]07      <1>             mov    byte [ss:IRQnum], 7
45757 0000F78B EB30                    <1>             jmp    short IRQ_service
45758                                  <1> IRQ_service9:
45759 0000F78D 36C605[4A610100]09      <1>             mov    byte [ss:IRQnum], 9
45760 0000F795 EB26                    <1>             jmp    short IRQ_service
45761                                  <1> IRQ_service10:
45762 0000F797 36C605[4A610100]0A      <1>             mov    byte [ss:IRQnum], 10
45763 0000F79F EB1C                    <1>             jmp    short IRQ_service
45764                                  <1> IRQ_service11:
45765 0000F7A1 36C605[4A610100]0B      <1>             mov    byte [ss:IRQnum], 11
45766 0000F7A9 EB12                    <1>             jmp    short IRQ_service
45767                                  <1> IRQ_service12:
45768 0000F7AB 36C605[4A610100]0C      <1>             mov    byte [ss:IRQnum], 12
45769 0000F7B3 EB08                    <1>             jmp    short IRQ_service
45770                                  <1> IRQ_service13:
45771 0000F7B5 36C605[4A610100]0D      <1>             mov    byte [ss:IRQnum], 13
45772                                  <1>             ;jmp   short IRQ_service
45773                                  <1> IRQ_service:
45774                                  <1>             ; 13/06/2017
45775                                  <1>             ; 11/06/2017
45776                                  <1>             ; 10/06/2017
45777                                  <1>             ; 01/03/2017, 04/03/2017
45778                                  <1>             ; 27/02/2017, 28/02/2017
45779 0000F7BD 1E                      <1>             push   ds
45780 0000F7BE 06                      <1>             push   es
45781 0000F7BF 0FA0                    <1>             push   fs
45782 0000F7C1 0FA8                    <1>             push   gs
45783                                  <1>
45784 0000F7C3 60                      <1>             pushad ; eax,ecx,edx,ebx,esp,ebp,esi,edi
45785 0000F7C4 66B91000                <1>             mov     cx, KDATA
45786 0000F7C8 8ED9                    <1>             mov     ds, cx
45787 0000F7CA 8EC1                    <1>             mov     es, cx
45788 0000F7CC 8EE1                    <1>             mov     fs, cx
45789 0000F7CE 8EE9                    <1>             mov     gs, cx
```

```
45790                                  <1>
45791 0000F7D0 0F20D8                  <1>             mov    eax, cr3
45792 0000F7D3 A3[46610100]            <1>             mov    [IRQ_cr3], eax
45793                                  <1>
45794 0000F7D8 A1[C84D0100]            <1>             mov    eax, [k_page_dir]
45795 0000F7DD 0F22D8                  <1>             mov    cr3, eax
45796                                  <1>
45797 0000F7E0 A0[4A610100]            <1>             mov    al, [IRQnum]
45798                                  <1>
45799                                  <1>             ;mov   cl, [sysflg]
45800                                  <1>             ;mov   [u.r_mode], cl  ; system (0) or user mode (FFh)
45801                                  <1> IRQsrv_0:
45802 0000F7E5 0FB6D8                  <1>             movzx  ebx, al
45803 0000F7E8 8A9B[CC0B0100]          <1>             mov    bl, [ebx+IRQenum] ; IRQ (available) index number + 1
45804                                  <1>             ; 01/03/2017
45805 0000F7EE FECB                    <1>             dec    bl  ; IRQ index number, 0 to 8
45806 0000F7F0 0F8807010000            <1>             js     IRQsrv_5 ; not available to use here!?
45807                                  <1>             ;
45808 0000F7F6 80BB[10610100]80        <1>             cmp    byte [ebx+IRQ.method], 80h ; using by a dev or kernel?
45809 0000F7FD 7205                    <1>             jb     short IRQsrv_1 ; no
45810                                  <1>
45811                                  <1>             ; If the IRQ service is already owned by TRDOS 386 kernel
45812                                  <1>             ;      or a Device driver
45813                                  <1>             ; we need to call 'dev_IRQ_service'
45814                                  <1>
45815                                  <1>             ; IRQ number in AL
45816 0000F7FF E868020000              <1>             call   dev_IRQ_service      ; IRQ service for device drivers
45817                                  <1>             ; IRQ number in AL
45818                                  <1> IRQsrv_1:
45819                                  <1>             ; check user callback service status
45820                                  <1>             ; AL = IRQ number
45821                                  <1>             ; EBX = IRQ (Available) Index number
45822                                  <1>
45823 0000F804 A2[D7030600]            <1>             mov    [u.irqwait], al ; set waiting IRQ flag
45824                                  <1>
45825 0000F809 8A83[FE600100]          <1>             mov    al, [ebx+IRQ.owner]
45826 0000F80F 20C0                    <1>             and    al, al
45827 0000F811 0F84E6000000            <1>             jz     IRQsrv_5 ; it is not owned by a user/proc
45828                                  <1>
45829                                  <1>             ; 03/03/2017
45830 0000F817 89DA                    <1>             mov    edx, ebx
45831 0000F819 C0E202                  <1>             shl    dl, 2
45832 0000F81C 8B92[22610100]          <1>             mov    edx, [edx+IRQ.addr] ; S.R.B. or Callback service addr
45833                                  <1>
45834 0000F822 8AA3[10610100]          <1>             mov    ah, [ebx+IRQ.method]
45835 0000F828 F6C401                  <1>             test   ah, 1
45836 0000F82B 7534                    <1>             jnz    short IRQsrv_4 ; Callback service method
45837                                  <1>
45838                                  <1>             ; Signal Response Byte method
45839                                  <1>             ;mov   edx, [edx+IRQ.addr] ; Signal Response Byte address
45840                                  <1>             ;                          ; (Physical address, non-swappable)
45841 0000F82D 80E402                  <1>             and    ah, 2 ; bit 1, (S.R.B.) counter (auto increment) method
45842 0000F830 8AA3[19610100]          <1>             mov    ah, [ebx+IRQ.srb] ; Signal Response Byte value
45843 0000F836 7408                    <1>             jz     short IRQsrv_2 ; fixed S.R.B. value
45844                                  <1>             ; counter method (auto increment)
45845 0000F838 FEC4                    <1>             inc    ah
45846 0000F83A 88A3[19610100]          <1>             mov    [ebx+IRQ.srb], ah ; next (count) number
45847                                  <1> IRQsrv_2:
45848 0000F840 8822                    <1>             mov    [edx], ah ; put S.R.B. val to the user's S.R.B. addr
45849 0000F842 C605[D7030600]00        <1>             mov    byte [u.irqwait], 0 ; clear waiting IRQ flag
45850                                  <1>
45851 0000F849 3A05[B3030600]          <1>             cmp    al, [u.uno]
45852 0000F84F 0F84A8000000            <1>             je     IRQsrv_5 ; the owner is current user/process
45853                                  <1> IRQsrv_3:
45854                                  <1>             ; the owner is not current user/process
45855                                  <1>             ; AL = process number
45856 0000F855 B202                    <1>             mov    dl, 2 ; priority, 2 = event (high)
45857 0000F857 E855FAFFFF              <1>             call   set_run_sequence
45858                                  <1>
45859                                  <1>             ; [u.irqwait] = waiting IRQ number for callback service
45860                                  <1>
45861 0000F85C E99C000000              <1>             jmp    IRQsrv_5
45862                                  <1> IRQsrv_4:
45863 0000F861 3A05[B3030600]          <1>             cmp    al, [u.uno]  ; is the owner is current user/process?
45864 0000F867 75EC                    <1>             jne    short IRQsrv_3 ; no !
45865                                  <1>
45866                                  <1>             ; Check if an IRQ callback service already in progress
45867 0000F869 803D[D8030600]00        <1>             cmp    byte [u.r_lock], 0
45868 0000F870 0F8787000000            <1>             ja     IRQsrv_5 ; nothing to do !
45869                                  <1>                             ; (we need to complete prev callback)
45870 0000F876 803D[D4030600]00        <1>             cmp    byte [u.t_lock], 0
45871 0000F87D 777E                    <1>             ja     short IRQsrv_5 ; nothing to do !
45872                                  <1>                             ; (we need to complete timer callback)
45873                                  <1>
45874                                  <1>             ; 04/03/2017
45875 0000F87F C605[D7030600]00        <1>             mov    byte [u.irqwait], 0 ; reset/clear waiting IRQ flag
45876                                  <1>
45877 0000F886 FE05[D8030600]          <1>             inc    byte [u.r_lock] ; 'IRQ callback service in progress' flag
45878                                  <1>
45879 0000F88C 8A0D[5B030600]          <1>             mov    cl, [sysflg]   ; (system call) mode flag (kernel/user)
45880 0000F892 880D[D9030600]          <1>             mov    [u.r_mode], cl ; system mode (0) or user mode (FFh)
45881                                  <1>
45882                                  <1>             ;
45883 0000F898 8B2D[644D0100]          <1>             mov    ebp, [tss.esp0] ; kernel stack address (for ring 0)
45884 0000F89E 83ED14                  <1>             sub    ebp, 20             ; eip, cs, eflags, esp, ss
45885 0000F8A1 892D[5C030600]          <1>             mov    [u.sp], ebp
45886 0000F8A7 8925[60030600]          <1>             mov    [u.usp], esp
45887                                  <1>
45888                                  <1>             ;or    word [ebp+8], 200h ; 22/01/2017, force enabling interrupts
45889                                  <1>
45890 0000F8AD 8B44241C                <1>             mov    eax, [esp+28] ; pushed eax
45891 0000F8B1 A3[64030600]            <1>             mov    [u.r0], eax
```

```
45892                                  <1>
45893 0000F8B6 E86CEEFFFF              <1>             call  wswap ; save user's registers & status
45894                                  <1>
45895                                  <1>             ; software int is in ring 0 but IRQ handler must return to ring 3
45896                                  <1>             ; so, ring 3 return address and stack registers
45897                                  <1>             ; (eip, cs, eflags, esp, ss)
45898                                  <1>             ; must be copied to IRQ handler return
45899                                  <1>             ; eip will be replaced by callback service routine address
45900                                  <1>
45901 0000F8BB C605[5B030600]FF        <1>             mov   byte [sysflg], 0FFh ; user mode
45902                                  <1>
45903                                  <1>             ; system mode (system call)
45904                                  <1>             ;mov   ebp, [u.sp] ; EIP (u), CS (UCODE), EFLAGS (u),
45905                                  <1>                                ; ESP (u), SS (UDATA)
45906                                  <1>
45907 0000F8C2 8B4510                  <1>             mov   eax, [ebp+16]; SS (UDATA)
45908 0000F8C5 89E6                    <1>             mov   esi, esp
45909 0000F8C7 50                      <1>             push  eax
45910 0000F8C8 50                      <1>             push  eax
45911 0000F8C9 89E7                    <1>             mov   edi, esp
45912 0000F8CB 893D[60030600]          <1>             mov   [u.usp], edi
45913 0000F8D1 B908000000              <1>             mov   ecx, ((ESPACE/4) - 4) ; except DS, ES, FS, GS
45914 0000F8D6 F3A5                    <1>             rep   movsd
45915 0000F8D8 B104                    <1>             mov   cl, 4
45916 0000F8DA F3AB                    <1>             rep   stosd
45917 0000F8DC 893D[5C030600]          <1>             mov   [u.sp], edi
45918 0000F8E2 89EE                    <1>             mov   esi, ebp
45919 0000F8E4 B105                    <1>             mov   cl, 5 ; EIP (u), CS (UCODE), EFLAGS (u), ESP (u), SS (UDATA)
45920 0000F8E6 F3A5                    <1>             rep   movsd
45921                                  <1>             ;
45922                                  <1>
45923 0000F8E8 8B0D[B8030600]          <1>             mov   ecx, [u.pgdir]
45924 0000F8EE 890D[46610100]          <1>             mov   [IRQ_cr3], ecx
45925                                  <1>
45926                                  <1> set_IRQ_callback_addr:
45927                                  <1>             ;
45928                                  <1>             ; This routine sets return address
45929                                  <1>             ; to start of user's interrupt
45930                                  <1>             ; service (callback) address
45931                                  <1>             ;
45932                                  <1>             ; INPUT:
45933                                  <1>             ;     EDX = callback routine/service address
45934                                  <1>             ;           (virtual, not physical address!)
45935                                  <1>             ;     [u.sp] = kernel stack, points to
45936                                  <1>             ;              user's EIP,CS,EFLAGS,ESP,SS
45937                                  <1>             ;              registers.
45938                                  <1>             ; OUTPUT:
45939                                  <1>             ;     EIP (user) = callback (service) address
45940                                  <1>             ;     CS (user) = UCODE
45941                                  <1>             ;     EFLAGS (user) = flags before callback
45942                                  <1>             ;      ESP (user) = ESP-4 (user, before callback)
45943                                  <1>             ;     [ESP](user) = EIP (user) before callback
45944                                  <1>             ;
45945                                  <1>             ; Note: If CPU was in user mode while entering
45946                                  <1>             ;     the timer interrupt service routine,
45947                                  <1>             ;     'IRET' will get return to callback routine
45948                                  <1>             ;     immediately. If CPU was in system/kernel mode
45949                                  <1>             ;     'iret' will get return to system call and
45950                                  <1>             ;     then, callback routine will be return address
45951                                  <1>             ;     from system call. (User's callback/service code
45952                                  <1>             ;     will be able to return to normal return address
45953                                  <1>             ;     via a 'sysrele' system call at the end.)
45954                                  <1>             ;
45955                                  <1>             ; Note: User's IRQ callback service code must be ended
45956                                  <1>             ;     with a 'sysrele' system call !
45957                                  <1>             ;
45958                                  <1>             ;     For example:
45959                                  <1>             ;
45960                                  <1>             ;     audio_IRQ_callback:
45961                                  <1>             ;         ...
45962                                  <1>             ;         <load DMA buffer with audio data>
45963                                  <1>             ;         ...
45964                                  <1>             ;         mov eax, 39 ; 'sysrele'
45965                                  <1>             ;         int 40h ; TRDOS 386 system call (interrupt)
45966                                  <1>             ;
45967                                  <1>
45968                                  <1>             ;mov   edx, [edx+IRQ.addr] ; Callback service address
45969                                  <1>             ;                          ; (Virtual address)
45970                                  <1>
45971 0000F8F4 8B2D[5C030600]          <1>             mov   ebp, [u.sp]; kernel's stack, points to EIP (user)
45972 0000F8FA 895500                  <1>             mov   [ebp], edx
45973                                  <1> IRQsrv_5:
45974                                  <1>             ; EOI & return
45975                                  <1>             ; 11/06/2017
45976                                  <1>             ; 10/06/2017
45977 0000F8FD A0[4A610100]            <1>             mov   al, [IRQnum]
45978 0000F902 FA                      <1>             cli
45979 0000F903 3C07                    <1>             cmp   al, 7
45980 0000F905 7604                    <1>             jna   short IRQsrv_6
45981                                  <1>             ;
45982                                  <1>             ;mov   al, EOI      ; end of interrupt
45983 0000F907 B020                    <1>             mov   al, 20h
45984                                  <1>             ;cli         ; disable interrupts till stack cleared
45985                                  <1>             ;out   INTB00, al ; For controll2 #2
45986 0000F909 E6A0                    <1>             out   0A0h, al
45987                                  <1> IRQsrv_6:
45988                                  <1>             ;mov   byte [IRQnum], 0 ; reset
45989                                  <1>             ;mov   al, EOI      ; end of interrupt
45990 0000F90B B020                    <1>             mov   al, 20h
45991                                  <1>             ;cli         ; disable interrupts till stack cleared
45992                                  <1>             ;out   INTA00, al ; end of interrupt to 8259 - 1
45993 0000F90D E620                    <1>             out   20h, al
```

```
45994                                  <1> IRQsrv_7:
45995                                  <1>             ;; 13/06/2017
45996                                  <1>             ;or    word [ebp+8], 200h ; force enabling interrupts
45997                                  <1>             ;
45998 0000F90F 8B0D[46610100]          <1>             mov    ecx, [IRQ_cr3]      ; previous content of cr3 register
45999 0000F915 0F22D9                  <1>             mov    cr3, ecx     ; restore cr3 register content
46000                                  <1>             ;
46001 0000F918 61                      <1>             popad ; edi,esi,ebp,(icrement esp by 4), ebx,edx,ecx,eax
46002                                  <1>             ;
46003 0000F919 0FA9                    <1>             pop    gs
46004 0000F91B 0FA1                    <1>             pop    fs
46005 0000F91D 07                      <1>             pop    es
46006 0000F91E 1F                      <1>             pop    ds
46007                                  <1>             ;
46008 0000F91F CF                      <1>             iretd  ; return from interrupt
46009                                  <1>
46010                                  <1> get_device_number:
46011                                  <1>             ; 08/10/2016
46012                                  <1>             ; 07/10/2016 - TRDOS 386 (TRDOS v2.0)
46013                                  <1>             ;
46014                                  <1>             ; This procedure compares name of requested
46015                                  <1>             ; device with kernel device names and
46016                                  <1>             ; installable device names. If names match,
46017                                  <1>             ; the relevant device index (entry) number
46018                                  <1>             ; will be returned the caller (sysopen)
46019                                  <1>             ; for the requested device.
46020                                  <1>             ;
46021                                  <1>             ; NOTE: Installable device drivers must
46022                                  <1>             ; be loaded before using 'sysopen'
46023                                  <1>             ; (opendev) system call.
46024                                  <1>             ;
46025                                  <1>             ; INPUT:
46026                                  <1>             ;    ESI = device name address (ASCIIZ)
46027                                  <1>             ;         (in kernel's memory space)
46028                                  <1>             ;    max name length = 8 without '/dev/')
46029                                  <1>             ;    Device name will be capitalized
46030                                  <1>             ;    and if there is, '/dev/' will be
46031                                  <1>             ;    removed from name before comparising)
46032                                  <1>             ;
46033                                  <1>             ; OUTPUT:
46034                                  <1>             ;    cf = 0 ->
46035                                  <1>             ;      EAX (AL) = device entry/index number
46036                                  <1>             ;    cf = 1 -> device not found (installed)
46037                                  <1>             ;             or invalid device name
46038                                  <1>             ;             (AL=0)
46039                                  <1>             ;    device_name = device name address (asciiz)
46040                                  <1>             ;
46041                                  <1>             ; Modified registers: EAX, EBX, ESI, EDI
46042                                  <1>
46043 0000F920 BF[895B0100]            <1>             mov    edi, device_name
46044 0000F925 E805010000              <1>             call   lodsb_capitalize
46045 0000F92A 88C4                    <1>             mov    ah, al
46046 0000F92C 3C2F                    <1>             cmp    al, '/'
46047 0000F92E 750E                    <1>             jne    short gdn_1
46048 0000F930 BF[895B0100]            <1>             mov    edi, device_name
46049 0000F935 E8F5000000              <1>             call   lodsb_capitalize
46050                                  <1> gdn_0:
46051 0000F93A 20C0                    <1>             and    al, al ; 0 ?
46052 0000F93C 7420                    <1>             jz     short gdn_err ; null name after '/'
46053                                  <1> gdn_1:
46054 0000F93E 3C44                    <1>             cmp    al, 'D'
46055 0000F940 7517                    <1>             jne    short gdn_2
46056 0000F942 E8E8000000              <1>             call   lodsb_capitalize
46057 0000F947 3C45                    <1>             cmp    al, 'E'
46058 0000F949 750E                    <1>             jne    short gdn_2
46059 0000F94B E8DF000000              <1>             call   lodsb_capitalize
46060 0000F950 3C56                    <1>             cmp    al, 'V'
46061 0000F952 7505                    <1>             jne    short gdn_2
46062 0000F954 AC                      <1>             lodsb
46063 0000F955 3C2F                    <1>             cmp    al, '/'
46064 0000F957 740D                    <1>             je     short gdn_4
46065                                  <1> gdn_2:
46066 0000F959 80FC2F                  <1>             cmp    ah, '/'
46067 0000F95C 750F                    <1>             jne    short gdn_5
46068                                  <1> gdn_err:
46069                                  <1>             ; invalid device name or device not found
46070 0000F95E 31C0                    <1>             xor    eax, eax ; 0
46071 0000F960 F9                      <1>             stc
46072 0000F961 C3                      <1>             retn
46073                                  <1> gdn_3:
46074 0000F962 3C2F                    <1>             cmp    al, '/'
46075 0000F964 7507                    <1>             jne    short gdn_5
46076                                  <1> gdn_4:
46077 0000F966 BF[895B0100]            <1>             mov    edi, device_name
46078 0000F96B EB04                    <1>             jmp    short gdn_6
46079                                  <1> gdn_5:
46080 0000F96D 3C00                    <1>             cmp    al, 0
46081 0000F96F 7419                    <1>             je     short gdn_7
46082                                  <1> gdn_6:
46083 0000F971 E8B9000000              <1>             call lodsb_capitalize
46084 0000F976 81FF[915B0100]          <1>             cmp    edi, device_name + 8
46085 0000F97C 72E4                    <1>             jb     short gdn_3
46086 0000F97E 3C00                    <1>             cmp    al, 0
46087 0000F980 75DC                    <1>             jne    short gdn_err
46088 0000F982 81FF[8A5B0100]          <1>             cmp    edi, device_name + 1
46089 0000F988 76D4                    <1>             jna    short gdn_err ; null name after '/'
46090                                  <1> gdn_7:
46091 0000F98A AA                      <1>             stosb
46092                                  <1>             ; zero padding ("NAME",0,0,0,0)
46093 0000F98B 81FF[915B0100]          <1>             cmp    edi, device_name + 8
46094 0000F991 72F7                    <1>             jb     short gdn_7
46095                                  <1> gdn_8:
```

```
46096                                  <1>             ; search for kernel device names
46097 0000F993 BE[895B0100]            <1>             mov    esi, device_name
46098 0000F998 BF[B2090100]            <1>             mov    edi, KDEV_NAME
46099 0000F99D 31C0                    <1>             xor    eax, eax
46100                                  <1> gdn_9:
46101 0000F99F A7                      <1>             cmpsd
46102 0000F9A0 7505                    <1>             jne    short gdn_10
46103 0000F9A2 A7                      <1>             cmpsd
46104 0000F9A3 7503                    <1>             jne    short gdn_11
46105 0000F9A5 EB2B                    <1>             jmp    short gdn_17 ; match
46106                                  <1> gdn_10:
46107 0000F9A7 A7                      <1>             cmpsd  ; add esi, 4 & add edi, 4
46108                                  <1> gdn_11:
46109 0000F9A8 BE[895B0100]            <1>             mov    esi, device_name
46110 0000F9AD FEC0                    <1>             inc    al
46111 0000F9AF 3C16                    <1>             cmp    al, NumOfKernelDevNames
46112 0000F9B1 72EC                    <1>             jb     short gdn_9
46113                                  <1> gdn_12:
46114                                  <1>             ; search for installable device names
46115                                  <1>             ; esi = offset device_name
46116 0000F9B3 BF[B45B0100]            <1>             mov    edi, IDEV_NAME
46117 0000F9B8 28C0                    <1>             sub    al, al ; 0
46118                                  <1> gdn_13:
46119 0000F9BA A7                      <1>             cmpsd
46120 0000F9BB 7505                    <1>             jne    short gdn_14
46121 0000F9BD A7                      <1>             cmpsd
46122 0000F9BE 7503                    <1>             jne    short gdn_15
46123 0000F9C0 EB3F                    <1>             jmp    short gdn_19 ; match
46124                                  <1> gdn_14:
46125 0000F9C2 A7                      <1>             cmpsd  ; add esi, 4 & add edi, 4
46126                                  <1> gdn_15:
46127 0000F9C3 BE[895B0100]            <1>             mov    esi, device_name
46128 0000F9C8 FEC0                    <1>             inc    al
46129 0000F9CA 3C08                    <1>             cmp    al, NumOfInstallableDevices
46130 0000F9CC 72EC                    <1>             jb     short gdn_13
46131                                  <1>
46132                                  <1> gdn_16:     ; error: invalid device name (not found) !
46133 0000F9CE 30C0                    <1>             xor    al, al
46134 0000F9D0 F9                      <1>             stc
46135 0000F9D1 C3                      <1>             retn
46136                                  <1>
46137                                  <1> gdn_17:           ; name match (with one of kernel device names)
46138                                  <1>             ;
46139                                  <1>             ; convert KDEV_NAME index to
46140                                  <1>             ; KDEV_NUMBER index
46141                                  <1>             ; (different names are used for same devices)
46142                                  <1>             ; (example: "COM1" & "TTY8" = device number 18)
46143 0000F9D2 89C3                    <1>             mov    ebx, eax ; < 256
46144 0000F9D4 8A83[620A0100]          <1>             mov    al, [KDEV_NUMBER+ebx]
46145                                  <1>
46146                                  <1>             ; check if empty dev entry in the list
46147 0000F9DA 80B8[385D0100]00        <1>             cmp    byte [DEV_OPENMODE+eax], 0
46148 0000F9E1 771B                    <1>             ja     short gdn_18 ; it must be already set
46149                                  <1>
46150                                  <1>             ; (re)set device name and access flags
46151                                  <1>             ; (remain open work will be easy after that)
46152                                  <1>             ; (NOTE: here, data will be copied to bss section)
46153 0000F9E3 88C3                    <1>             mov    bl, al
46154 0000F9E5 83EF08                  <1>             sub    edi, 8 ; kernel device name address (data)
46155 0000F9E8 66C1E302                <1>             shl    bx, 2
46156 0000F9EC 89BB[565D0100]          <1>             mov    [DEV_NAME_PTR+ebx], edi ; (all) device names
46157 0000F9F2 8A98[B80B0100]          <1>             mov    bl, [KDEV_ACCESS+eax] ; kernel dev list (data)
46158 0000F9F8 8898[845C0100]          <1>             mov    [DEV_ACCESS+eax], bl ; (all) device list (bss)
46159                                  <1> gdn_18:
46160 0000F9FE FEC0                    <1>             inc    al ; 1 to NumOfKernelDevNames (<=7Fh)
46161                                  <1>             ; eax = device index/entry number
46162 0000FA00 C3                      <1>             retn
46163                                  <1>
46164                                  <1> gdn_19:           ; name match (with one of installable device names)
46165                                  <1>             ;
46166                                  <1>             ; al = 0 to NumOfInstallableDevices - 1 (<=7Fh)
46167                                  <1>
46168 0000FA01 89C3                    <1>             mov    ebx, eax
46169 0000FA03 80C316                  <1>             add    bl, NumOfKernelDevices    ; < NUMOFDEVICES
46170                                  <1>
46171                                  <1>             ; check if empty dev entry in the list
46172 0000FA06 80BB[385D0100]00        <1>             cmp    byte [DEV_OPENMODE+ebx], 0
46173 0000FA0D 771D                    <1>             ja     short gdn_20 ; it must be already set
46174                                  <1>
46175                                  <1>             ; (re)set device name and access flags
46176                                  <1>             ; (remain open work will be easy after that)
46177 0000FA0F 83EF08                  <1>             sub    edi, 8 ; installable device name address
46178 0000FA12 66C1E302                <1>             shl    bx, 2 ;*4
46179 0000FA16 89BB[565D0100]          <1>             mov    [DEV_NAME_PTR+ebx], edi ; (all) device names
46180 0000FA1C 66C1EB02                <1>             shr    bx, 2
46181 0000FA20 8A80[FC5B0100]          <1>             mov    al, [IDEV_FLAGS+eax] ; installable dev list
46182 0000FA26 8883[845C0100]          <1>             mov    [DEV_ACCESS+ebx], al ; (all) device list
46183                                  <1> gdn_20:
46184 0000FA2C 88D8                    <1>             mov    al, bl
46185                                  <1>             ; eax = device index/entry number ; < NUMOFDEVICES
46186 0000FA2E C3                      <1>             retn
46187                                  <1>
46188                                  <1> lodsb_capitalize:
46189                                  <1>       ; 07/10/2016 - TRDOS 386 (TRDOS v2.0)
46190                                  <1>       ; INPUT -> [esi] = character
46191                                  <1>       ;          edi = destination
46192                                  <1>       ; OUTPUT -> AL contains capitalized character
46193                                  <1>       ;        esi = esi+1
46194                                  <1>       ;        edi = edi+1
46195                                  <1>       ;
46196 0000FA2F AC                      <1>       lodsb
46197 0000FA30 3C61                    <1>       cmp al, 61h
```

```
46198 0000FA32 7206              <1>       jb  short lodsb_cap_retn
46199 0000FA34 3C7A              <1>       cmp al, 7Ah
46200 0000FA36 7702              <1>       ja  short lodsb_cap_retn
46201 0000FA38 24DF              <1>       and al, 0DFh
46202                            <1> lodsb_cap_retn:
46203 0000FA3A AA                <1>       stosb
46204 0000FA3B C3                <1>       retn
46205                            <1>
46206                            <1> device_open:
46207                            <1>       ; 08/10/2016 - TRDOS 386 (TRDOS v2.0)
46208                            <1>       ; Complete device opening work for sysopen (device)
46209                            <1>       ;
46210                            <1>       ; INPUT ->
46211                            <1>       ;     EAX = Device Number (AL)
46212                            <1>       ;        CL = Open mode (1 = read, 2 = write)
46213                            <1>       ;       CH = Device access byte (bit 0 = 0)
46214                            <1>       ; OUTPUT ->
46215                            <1>       ;     EAX = Device Number
46216                            <1>       ;     CF = 0 -> device has been opened
46217                            <1>       ;     CF = 1 -> device could not be opened
46218                            <1>       ;
46219                            <1>       ;  Modified registers: ebx, (edx, ecx, esi, edi, ebp)
46220                            <1>       ;
46221                            <1>
46222 0000FA3C 89C3              <1>       mov    ebx, eax
46223 0000FA3E 66C1E302          <1>       shl    bx, 2 ; *4
46224                            <1>
46225 0000FA42 F6C580            <1>       test   ch, 80h ; bit 7, installable device driver flag
46226 0000FA45 7406              <1>       jz     short d_open_2 ; Kernel device
46227                            <1>       ; installable device
46228                            <1> d_open_1:
46229 0000FA47 FFA3[005C0100]    <1>         jmp  dword [ebx+IDEV_OADDR-4]
46230                            <1> d_open_2:
46231 0000FA4D FFA3[740A0100]    <1>       jmp    dword [ebx+KDEV_OADDR-4]
46232                            <1>
46233                            <1> device_close:
46234                            <1>       ; 08/10/2016 - TRDOS 386 (TRDOS v2.0)
46235                            <1>       ; Complete device closing work for sysclose (device)
46236                            <1>       ;
46237                            <1>       ; INPUT ->
46238                            <1>       ;     EAX = Device Number (AL)
46239                            <1>       ;        CL = Open mode (1 = read, 2 = write)
46240                            <1>       ;       CH = Device access byte (bit 0 = 0)
46241                            <1>       ; OUTPUT ->
46242                            <1>       ;     EAX = Device Number
46243                            <1>       ;     CF = 0 -> device has been closed
46244                            <1>       ;     CF = 1 -> device could not be closed
46245                            <1>       ;
46246                            <1>       ; Modified registers: ebx, (edx, ecx, esi, edi, ebp)
46247                            <1>       ;
46248                            <1>
46249 0000FA53 89C3              <1>       mov    ebx, eax
46250 0000FA55 66C1E302          <1>       shl    bx, 2 ; *4
46251                            <1>
46252 0000FA59 F6C580            <1>       test   ch, 80h ; bit 7, installable device driver flag
46253 0000FA5C 7406              <1>       jz     short d_close_2 ; Kernel device
46254                            <1>       ; installable device
46255                            <1> d_close_1:
46256 0000FA5E FFA3[205C0100]    <1>         jmp  dword [ebx+IDEV_CADDR-4]
46257                            <1> d_close_2:
46258 0000FA64 FFA3[C40A0100]    <1>       jmp    dword [ebx+KDEV_CADDR-4]
46259                            <1>
46260                            <1> rnull:
46261                            <1>       ; 07/10/2016 - TRDOS 386 (TRDOS v2.0)
46262                            <1>       ; read null (read from null device)
46263 0000FA6A C3                <1>       retn
46264                            <1>
46265                            <1> wnull:
46266                            <1>       ; 07/10/2016 - TRDOS 386 (TRDOS v2.0)
46267                            <1>       ; write null (write to null device)
46268 0000FA6B C3                <1>       retn
46269                            <1>
46270                            <1> dev_IRQ_service:
46271                            <1>       ; 12/05/2017
46272                            <1>       ; 13/04/2017
46273                            <1>       ; 27/02/2017 - TRDOS 386 (TRDOS v2.0)
46274                            <1>       ; INPUT ->
46275                            <1>       ;     AL = IRQ Number (0 to 15)
46276                            <1>       ;
46277 0000FA6C 53                <1>       push   ebx
46278 0000FA6D 0FB6D8            <1>       movzx  ebx, al
46279 0000FA70 C0E302            <1>       shl    bl, 2 ; * 4
46280 0000FA73 8B9B[BE600100]    <1>       mov    ebx, [ebx+DEV_INT_HNDLR]
46281 0000FA79 21DB              <1>       and    ebx, ebx
46282 0000FA7B 7404              <1>         jz   short dIRQ_s_retn
46283 0000FA7D 50                <1>       push   eax
46284                            <1>
46285 0000FA7E FFD3              <1>       call   ebx
46286                            <1>
46287                            <1> ;mov ah, 04Eh
46288                            <1> ;mov al, [0B8000h]
46289                            <1> ;inc al
46290                            <1> ;and al, 7
46291                            <1> ;add al, '0'
46292                            <1> ;mov [0B8000h], ax
46293                            <1>
46294 0000FA80 58                <1>       pop    eax
46295                            <1> dIRQ_s_retn:
46296 0000FA81 5B                <1>       pop    ebx
46297 0000FA82 C3                <1>       retn
46298                            <1>
46299                            <1>
```

```
46300                                  <1> set_dev_IRQ_service:
46301                                  <1>       ; 13/04/2017 - TRDOS 386 (TRDOS v2.0)
46302                                  <1>       ;
46303                                  <1>       ; Set Device Interrupt Service
46304                                  <1>       ;
46305                                  <1>       ; INPUT ->
46306                                  <1>       ;     AL = IRQ Number
46307                                  <1>       ;     EBX = Hardware Interrupt Service Address
46308                                  <1>       ;
46309                                  <1>       ; Note: There is not a validation check here
46310                                  <1>       ;     because this procedure is called by
46311                                  <1>       ;     TRDOS 386 kernel !
46312                                  <1>       ;      (Even if a device driver does not exist
46313                                  <1>       ;     this setting may be used by sysaudio
46314                                  <1>       ;     and other system calls for hardware
46315                                  <1>       ;     components which use IRQ method for I/O.)
46316                                  <1>       ;
46317                                  <1>       ;push  esi
46318 0000FA83 0FB6F0                  <1>       movzx  esi, al
46319 0000FA86 66C1E602                <1>       shl    si, 2 ; * 4
46320 0000FA8A 899E[BE600100]          <1>       mov    [esi+DEV_INT_HNDLR], ebx
46321                                  <1>       ;pop   esi
46322 0000FA90 C3                      <1>       retn
46323                                  <1>
46324                                  <1>
46325                                  <1> sysaudio: ; AUDIO FUNCTIONS
46326                                  <1>       ; 22/06/2017
46327                                  <1>       ; 10/06/2017
46328                                  <1>       ; 05/06/2017
46329                                  <1>       ; 04/06/2017
46330                                  <1>       ; 28/05/2017
46331                                  <1>       ; 01/05/2017, 12/05/2017, 15/05/2017, 20/05/2017
46332                                  <1>       ; 21/04/2017, 22/04/2017, 23/04/2017, 24/04/2017
46333                                  <1>       ; 10/04/2017, 13/04/2017, 14/04/2017, 16/04/2017
46334                                  <1>       ; 03/04/2017 (VIA VT8237R)
46335                                  <1>       ; 01/04/2016 (trdosk6.s -> tdosk8.s)
46336                                  <1>       ; 16/05/2016 - TRDOS 386 (TRDOS v2.0)
46337                                  <1>       ;
46338                                  <1>       ; Inputs:
46339                                  <1>       ;
46340                                  <1>       ;     BH = 0 -> Beep (PC Speaker)
46341                                  <1>       ;          BL = Duration Counter (1 for 1/64 second)
46342                                  <1>       ;          CX = Frequency Divisor (1193180/Frequency)
46343                                  <1>       ;             (1331 for 886 Hz)
46344                                  <1>       ;
46345                                  <1>       ;     01/04/2017
46346                                  <1>       ;
46347                                  <1>       ;     BH = 1 -> DETECT (& ENABLE) AUDIO DEVICE
46348                                  <1>       ;          BL = 0 : PC SPEAKER
46349                                  <1>       ;               1 : SOUND BLASTER 16
46350                                  <1>       ;               2 : INTEL AC'97
46351                                  <1>       ;               3 : VIA VT8237R (VT8233)
46352                                  <1>       ;               4 : INTEL HDA
46353                                  <1>       ;            5-FEh : unknown/invalid
46354                                  <1>       ;              ; 04/06/2017
46355                                  <1>       ;             FFh : Get current audio device id
46356                                  <1>       ;
46357                                  <1>       ;     BH = 2 -> ALLOCATE AUDIO BUFFER (for user)
46358                                  <1>       ;            ECX = Audio Buffer Size (must be equal to
46359                                  <1>       ;                  the half of DMA buffer size)
46360                                  <1>       ;            EDX = Virtual Address of the buffer
46361                                  <1>       ;                  (This is not DMA buffer!)
46362                                  <1>       ;
46363                                  <1>       ;     BH = 3 -> INITIALIZE AUDIO DEVICE
46364                                  <1>       ;          BL = 0,2 -> for Signal Response Byte
46365                                  <1>       ;            CL = Signal Response Byte Value (fixed)
46366                                  <1>       ;                        if BL = 0
46367                                  <1>       ;                 auto increment of S.R.B. value
46368                                  <1>       ;                        if BL = 2
46369                                  <1>       ;              EDX = Signal Response (Return) Byte Address
46370                                  <1>       ;
46371                                  <1>       ;           BL = 1 for CallBack Method
46372                                  <1>       ;             EDX = CallBack Service Address (Virtual)
46373                                  <1>       ;
46374                                  <1>       ;           BL > 2 -> invalid function
46375                                  <1>       ;
46376                                  <1>       ;          (Audio buffer must be allocated before
46377                                  <1>       ;           initialization.)
46378                                  <1>       ;
46379                                  <1>       ;      BH = 4 -> START TO PLAY
46380                                  <1>       ;           BL = Mode
46381                                  <1>       ;               Bit 0 = mono/stereo (1 = stereo)
46382                                  <1>       ;               Bit 1 = 8 bit / 16 bit (1 = 16 bit)
46383                                  <1>       ;           CX = Sampling Rate (Hz)
46384                                  <1>       ;
46385                                  <1>       ;      BH = 5 -> PAUSE
46386                                  <1>       ;           BL = Any
46387                                  <1>       ;
46388                                  <1>       ;      BH = 6 -> CONTINUE TO PLAY
46389                                  <1>       ;           BL = Any
46390                                  <1>       ;
46391                                  <1>       ;      BH = 7 -> STOP
46392                                  <1>       ;           BL = Any
46393                                  <1>       ;
46394                                  <1>       ;      BH = 8 -> RESET
46395                                  <1>       ;           BL = Any
46396                                  <1>       ;
46397                                  <1>       ;      BH = 9 -> CANCEL (CALLBACK or S.R.B. SERVICE)
46398                                  <1>       ;           BL = Any
46399                                  <1>       ;
46400                                  <1>       ;      BH = 10 -> DEALLOCATE AUDIO BUFFER (for user)
46401                                  <1>       ;           BL = Any
```

```
46402                                 <1>       ;
46403                                 <1>       ;      BH = 11 -> SET VOLUME LEVEL
46404                                 <1>       ;           BL: (Bit 0 to 6)
46405                                 <1>       ;              0 = Master (Playback, Lineout) volume
46406                                 <1>       ;           CL = Left Channel Volume
46407                                 <1>       ;           CH = Right Channel Volume
46408                                 <1>       ;
46409                                 <1>       ;           Note: If BL >= 80h (Bit 7 of BL is set),
46410                                 <1>       ;           volume level will be set for next playing
46411                                 <1>       ;           (actual volume level will not be changed
46412                                 <1>       ;           immediately)
46413                                 <1>       ;
46414                                 <1>       ;      BH = 12 -> DISABLE AUDIO DEVICE
46415                                 <1>       ;           (reset audio device and unlink dma buffer)
46416                                 <1>       ;           BL = Any
46417                                 <1>       ;
46418                                 <1>       ;      12/05/2017
46419                                 <1>       ;      BH = 13 -> MAP DMA BUFFER TO USER
46420                                 <1>       ;          (for direct access to system's dma buffer)
46421                                 <1>       ;
46422                                 <1>       ;           ECX = map size in bytes
46423                                 <1>       ;              (will be rounded up to page borders)
46424                                 <1>       ;           EDX = Virtual Address of the buffer
46425                                 <1>       ;              (Will be rounded up to page borders)
46426                                 <1>       ;
46427                                 <1>       ;      05/06/2017
46428                                 <1>       ;      04/06/2017
46429                                 <1>       ;      BH = 14 -> GET AUDIO DEVICE INFO
46430                                 <1>       ;           BL: 0 = Audio Controller Info
46431                                 <1>       ;             > 0 = Invalid for now!
46432                                 <1>       ;
46433                                 <1>       ;      22/06/2017
46434                                 <1>       ;      BH = 15 -> GET CURRENT SOUND DATA (for graphics)
46435                                 <1>       ;           BL: 0 -> PCM OUT data
46436                                 <1>       ;             > 0 -> Invalid for now!
46437                                 <1>       ;           ECX = 0 -> Get DMA Buffer Pointer
46438                                 <1>       ;              EDX = Not Used
46439                                 <1>       ;           ECX > 0 -> Byte count for buffer (EDX)
46440                                 <1>       ;               EDX = Buffer Address (Virtual)
46441                                 <1>       ;
46442                                 <1>       ;
46443                                 <1>       ; Outputs:
46444                                 <1>       ;
46445                                 <1>       ;      For BH = 0 -> Beep
46446                                 <1>       ;          None
46447                                 <1>       ;
46448                                 <1>       ;      01/04/2017
46449                                 <1>       ;
46450                                 <1>       ;      For BH = 1 -> DETECT (& ENABLE) AUDIO DEVICE
46451                                 <1>       ;          AH = 0 : PC SPEAKER
46452                                 <1>       ;               1 : SOUND BLASTER 16
46453                                 <1>       ;               2 : INTEL AC'97
46454                                 <1>       ;               3 : VIA VT8237R (VT8233)
46455                                 <1>       ;               4 : INTEL HDA
46456                                 <1>       ;             5-FFh : unknown/invalid
46457                                 <1>       ;          AL = mode status
46458                                 <1>       ;               bit 0 = mono /stereo (1 = stereo)
46459                                 <1>       ;               bit 1 = 8 bit / 16 bit ( 1 = 16 bit)
46460                                 <1>       ;          04/06/2017
46461                                 <1>       ;          EBX = PCI DEVICE/VENDOR ID (if >0)
46462                                 <1>       ;              (BX = VENDOR ID)
46463                                 <1>       ;          (if CF = 1 -> Error code in EAX)
46464                                 <1>       ;
46465                                 <1>       ;      For BH = 2 -> ALLOCATE AUDIO BUFFER (for user)
46466                                 <1>       ;          EAX = Physical Address of the buffer
46467                                 <1>       ;          (if CF = 1 -> Error code in EAX)
46468                                 <1>       ;
46469                                 <1>       ;      For BH = 3 -> INITIALIZE AUDIO DEVICE
46470                                 <1>       ;          (if CF = 1 -> Error code in EAX)
46471                                 <1>       ;
46472                                 <1>       ;      For BH = 4 -> START TO PLAY
46473                                 <1>       ;          none (if CF = 1 -> Error code in EAX)
46474                                 <1>       ;
46475                                 <1>       ;      For BH = 5 -> PAUSE
46476                                 <1>       ;          none (if CF = 1 -> Error code in EAX)
46477                                 <1>       ;
46478                                 <1>       ;      For BH = 6 -> CONTINUE TO PLAY
46479                                 <1>       ;          none (if CF = 1 -> Error code in EAX)
46480                                 <1>       ;
46481                                 <1>       ;      For BH = 7 -> STOP
46482                                 <1>       ;          none (if CF = 1 -> Error code in EAX)
46483                                 <1>       ;
46484                                 <1>       ;      For BH = 8 -> RESET
46485                                 <1>       ;          none (if CF = 1 -> Error code in EAX)
46486                                 <1>       ;
46487                                 <1>       ;      For BH = 9 -> CANCEL (CALLBACK or S.R.B. SERVICE)
46488                                 <1>       ;          none (if CF = 1 -> Error code in EAX)
46489                                 <1>       ;
46490                                 <1>       ;      For BH = 10 -> DEALLOCATE AUDIO BUFFER (for user)
46491                                 <1>       ;          none (if CF = 1 -> Error code in EAX)
46492                                 <1>       ;
46493                                 <1>       ;      For BH = 11 -> SET VOLUME LEVEL
46494                                 <1>       ;          none (if CF = 1 -> Error code in EAX)
46495                                 <1>       ;
46496                                 <1>       ;      For BH = 12 -> DISABLE AUDIO DEVICE
46497                                 <1>       ;          none (if CF = 1 -> Error code in EAX)
46498                                 <1>       ;
46499                                 <1>       ;      12/05/2017
46500                                 <1>       ;      For BH = 13 -> MAP DMA BUFFER TO USER
46501                                 <1>       ;          EAX = Physical Address of the buffer
46502                                 <1>       ;          (if CF = 1 -> Error code in EAX)
46503                                 <1>       ;
```

```
                                <1>       ;     04/06/2017
                                <1>       ;     For BH = 14 -> GET AUDIO DEVICE INFO
                                <1>       ;     (for BL = 0) ; 05/06/2017
                                <1>       ;         EAX = IRQ Number in AL
                                <1>       ;              Audio Device Number in AH
                                <1>       ;         EBX = DEV/VENDOR ID
                                <1>       ;             (DDDDDDDDDDDDDDDDVVVVVVVVVVVVVVVV)
                                <1>       ;         ECX = BUS/DEV/FN
                                <1>       ;             (00000000BBBBBBBBDDDDDFFF00000000)
                                <1>       ;         EDX = NABMBAR/NAMBAR (for AC97)
                                <1>       ;              (Low word, DX = NAMBAR address)
                                <1>       ;         EDX = Base IO Addr (DX) for SB16 & VT8233
                                <1>       ;         (if CF = 1 -> Error code in EAX)
                                <1>       ;                       (ERR_DEV_NOT_RDY = 15)
                                <1>       ;
                                <1>       ;     22/06/2017
                                <1>       ;     For BH = 15 -> GET CURRENT SOUND DATA
                                <1>       ;                    (for graphics)
                                <1>       ;     (for BL = 0)
                                <1>       ;      If ECX input is 0
                                <1>       ;         EAX = DMA Buffer Current Position (Offset)
                                <1>       ;      If ECX input >  0
                                <1>       ;         EAX = Actual transfer count
                                <1>       ;         (Sound samples will be copied from
                                <1>       ;          Current DMA Buffer Position to EDX
                                <1>       ;          virtual address as EAX bytes.)
                                <1>       ;      ((If CF = 1 ->  Error code in EAX))
                                <1>       ;
                                <1>
0000FA91 80FF10                 <1>       cmp   bh, AUDIO1L/4
0000FA94 0F8315CAFFFF           <1>       jnb   sysret
                                <1>
0000FA9A C0E702                 <1>       shl   bh, 2 ; *4
0000FA9D 0FB6F7                 <1>       movzx esi, bh
                                <1>
                                <1>       ; 22/04/2017
0000FAA0 31C0                   <1>       xor   eax, eax
0000FAA2 A3[64030600]           <1>       mov   [u.r0], eax  ; 0
                                <1>
0000FAA7 FF96[B2FA0000]         <1>       call  dword [esi+AUDIO1]
                                <1>       ;jc   error
0000FAAD E9FDC9FFFF             <1>       jmp   sysret
                                <1>
0000FAB2 [A11D0000]             <1> AUDIO1:     dd    beep ; FUNCTION = 0 (bl = Duration Counter
                                <1>                   ;                   cx = Frequency Divisor
0000FAB6 [F2FA0000]             <1>       dd    soundc_detect
0000FABA [8EFB0000]             <1>       dd    sound_alloc
0000FABE [45FC0000]             <1>       dd    soundc_init
0000FAC2 [FDFD0000]             <1>       dd    sound_play
0000FAC6 [93FE0000]             <1>       dd    sound_pause
0000FACA [BDFE0000]             <1>       dd    sound_continue
0000FACE [E7FE0000]             <1>       dd    sound_stop
0000FAD2 [10FF0000]             <1>       dd    soundc_reset
0000FAD6 [41FF0000]             <1>       dd    soundc_cancel
0000FADA [67FF0000]             <1>       dd    sound_dalloc
0000FADE [92FF0000]             <1>       dd    sound_volume
0000FAE2 [E4FF0000]             <1>       dd    soundc_disable
0000FAE6 [56000100]             <1>       dd    sound_dma_map
0000FAEA [C5000100]             <1>       dd    soundc_info
0000FAEE [24010100]             <1>       dd    sound_data
                                <1>
                                <1> AUDIO1L     EQU   $ - AUDIO1
                                <1>
                                <1> soundc_detect:
                                <1>       ; FUNCTION = 1
                                <1>       ; bl = Audio device type number
                                <1>       ; (0= pc speaker, 1 = sound blaster 16, 2 = intel ac97
                                <1>       ;  3= via vt823x, 4 = intel HDA, 0FFh= any)
                                <1>
                                <1>       ; 04/06/2017
0000FAF2 8A25[4D610100]         <1>       mov   ah, [audio_device]
0000FAF8 80FBFF                 <1>       cmp   bl, 0FFh ; get current audio device id
0000FAFB 7408                   <1>       je    short sysaudio0
                                <1>
0000FAFD 20E4                   <1>       and   ah, ah
0000FAFF 741E                   <1>       jz    short soundc_get_dev
                                <1>
0000FB01 38DC                   <1>       cmp   ah, bl
0000FB03 7567                   <1>       jne   short soundc_dev_err
                                <1>
                                <1> sysaudio0:
0000FB05 A0[4E610100]           <1>       mov   al, [audio_mode]
                                <1> sysaudio1:
0000FB0A A3[64030600]           <1>       mov   [u.r0], eax
0000FB0F 8B1D[58610100]         <1>       mov   ebx, [audio_vendor] ; (DEVICE/VENDOR ID)
0000FB15 8B2D[60030600]         <1>       mov   ebp, [u.usp]
0000FB1B 895D10                 <1>       mov   [ebp+16], ebx  ; ebx
0000FB1E C3                     <1>       retn
                                <1>
                                <1> soundc_get_dev:
                                <1>       ; 28/05/2017
                                <1>       ; 03/04/2017, 24/04/2017
0000FB1F C605[4C610100]00       <1>       mov   byte [audio_pci], 0
0000FB26 80FB03                 <1>       cmp   bl, 3 ; VIA VT8233 (VT8237R) Audio Controller & AC97 Codec
                                <1>       ;jne  short soundc_get_dev_sb
                                <1>       ; 28/05/2017
0000FB29 7220                   <1>       jb    short soundc_get_dev_sb
0000FB2B 773F                   <1>       ja    short soundc_dev_err  ; temporary (28/05/2017)
                                <1>       ;
0000FB2D E82A140000             <1>       call  DetectVT8233
0000FB32 7238                   <1>       jc    short soundc_dev_err
                                <1>       ; eax = 0
```

```
46606                                  <1>
46607                                  <1>       ;mov   ebx, [audio_vendor]
46608                                  <1>       ; ebx = DEVICE/VENDOR ID
46609                                  <1>       ;      DDDDDDDDDDDDDDDDVVVVVVVVVVVVVVVV
46610                                  <1>
46611 0000FB34 B003                    <1>       mov    al, 3  ; VIA VT8237R (VT3233) Audio Controller
46612 0000FB36 88C4                    <1>       mov    ah, al
46613                                  <1>
46614                                  <1> soundc_get_pci_dev_ok: ; 28/05/2017
46615 0000FB38 FE05[4C610100]          <1>       inc    byte [audio_pci] ; = 1
46616                                  <1> soundc_get_dev_ok:
46617                                  <1>
46618                                  <1> soundc_get_dev_sb16_ok:
46619 0000FB3E A2[4D610100]            <1>       mov    [audio_device], al
46620 0000FB43 8825[4E610100]          <1>       mov    [audio_mode], ah ; stereo (bit0), 16 bit (bit1) capability
46621 0000FB49 EBBF                    <1>       jmp    short sysaudio1
46622                                  <1>
46623                                  <1> soundc_get_dev_sb:
46624                                  <1>       ; 24/04/2017
46625 0000FB4B 80FB01                  <1>       cmp    bl, 1 ; Sound Blaster 16
46626 0000FB4E 750E                    <1>       jne    short soundc_get_dev_ich ; 28/05/2017
46627                                  <1>       ;
46628 0000FB50 E858190000              <1>       call   DetectSB
46629 0000FB55 7215                    <1>       jc     short soundc_dev_err
46630 0000FB57 B801030000              <1>       mov    eax, 0301h ; Sound Blaster 16
46631 0000FB5C EBE0                    <1>       jmp    short soundc_get_dev_sb16_ok
46632                                  <1>
46633                                  <1> soundc_get_dev_ich:
46634                                  <1>       ; 28/05/2017
46635                                  <1>       ;cmp   bl, 2 ; Intel AC'97 Audio Controller (ICH)
46636                                  <1>       ;jne   short soundc_dev_err ; Temporary  (28/05/2017)
46637                                  <1>       ;                       ; (Here will be modified just after
46638                                  <1>       ;                       ; new sound card code will be ready!)
46639 0000FB5E E8EC130000              <1>       call   DetectICH
46640 0000FB63 7207                    <1>       jc     short soundc_dev_err
46641                                  <1>       ;
46642 0000FB65 B802030000              <1>       mov    eax, 0302h ; AC'97 (ICH)
46643 0000FB6A EBCC                    <1>       jmp    short soundc_get_pci_dev_ok
46644                                  <1>
46645                                  <1> soundc_dev_err:
46646 0000FB6C B80F000000              <1>       mov    eax, ERR_DEV_NOT_RDY ; Device not ready !
46647 0000FB71 EB0C                    <1>       jmp    short sysaudio_err
46648                                  <1>
46649                                  <1> sound_buff_error:
46650 0000FB73 B82E000000              <1>       mov    eax, ERR_BUFFER ; Buffer error !
46651 0000FB78 EB05                    <1>       jmp    short sysaudio_err
46652                                  <1>
46653                                  <1> soundc_respond_err:
46654                                  <1>       ; ERR_TIME_OUT ; 'time out !' error
46655 0000FB7A B819000000              <1>       mov    eax, ERR_DEV_NOT_RESP ; 'device not responding !' error
46656                                  <1> sysaudio_err:
46657 0000FB7F A3[64030600]            <1>       mov    [u.r0], eax
46658 0000FB84 A3[C8030600]            <1>       mov    [u.error], eax
46659 0000FB89 E901C9FFFF              <1>       jmp    error
46660                                  <1>
46661                                  <1> sound_alloc:
46662                                  <1>       ; FUNCTION =  2
46663                                  <1>       ; ecx = audio buffer size (in bytes)
46664                                  <1>       ; edx = audio buffer address (virtual)
46665                                  <1>       ; 28/05/2017
46666                                  <1>       ; 01/05/2017, 15/05/2017
46667                                  <1>       ; 21/04/2017, 24/04/2017
46668 0000FB8E 803D[4C610100]00        <1>       cmp    byte [audio_pci], 0
46669 0000FB95 7708                    <1>       ja     short snd_alloc_0
46670                                  <1>       ; Max. 64KB DMA buffer !!!
46671 0000FB97 81F900800000            <1>       cmp    ecx, 32768
46672 0000FB9D 77D4                    <1>       ja     short sound_buff_error
46673                                  <1> snd_alloc_0:
46674                                  <1>       ; 15/05/2017
46675 0000FB9F 81F900100000            <1>       cmp    ecx, 4096 ; PAGE_SIZE
46676 0000FBA5 72CC                    <1>       jb     short sound_buff_error
46677                                  <1>       ;
46678 0000FBA7 A1[60610100]            <1>       mov    eax, [audio_buffer] ; audio buffer address (current)
46679 0000FBAC 09C0                    <1>       or     eax, eax
46680 0000FBAE 7445                    <1>       jz     short snd_alloc_2
46681                                  <1>       ; audio buffer exists !
46682 0000FBB0 8A1D[B3030600]          <1>       mov    bl, [u.uno]
46683 0000FBB6 3A1D[75610100]          <1>       cmp    bl, [audio_user]
46684 0000FBBC 0F85F5000000            <1>       jne    sndc_owner_error ; not owner !
46685 0000FBC2 39D0                    <1>       cmp    eax, edx ; same virtual buffer address ?
46686 0000FBC4 7508                    <1>       jne    short snd_alloc_1
46687 0000FBC6 3B0D[68610100]          <1>       cmp    ecx, [audio_buff_size]
46688 0000FBCC 746C                    <1>       je     short snd_alloc_3 ; Nothing to do !
46689                                  <1>                              ; Buffer has been set already!
46690                                  <1> snd_alloc_1:
46691 0000FBCE 51                      <1>       push   ecx
46692 0000FBCF 52                      <1>       push   edx
46693 0000FBD0 89C3                    <1>       mov    ebx, eax ; audio buffer address (current)
46694 0000FBD2 8B0D[68610100]          <1>       mov    ecx, [audio_buff_size]
46695 0000FBD8 E8A15BFFFF              <1>       call   deallocate_user_pages
46696 0000FBDD 5A                      <1>       pop    edx
46697 0000FBDE 59                      <1>       pop    ecx
46698 0000FBDF 31C0                    <1>       xor    eax, eax ; 0
46699 0000FBE1 A3[60610100]            <1>       mov    [audio_buffer], eax  ; 0
46700 0000FBE6 A3[64610100]            <1>       mov    [audio_p_buffer], eax  ; 0
46701 0000FBEB A3[68610100]            <1>       mov    [audio_buff_size], eax
46702 0000FBF0 A2[75610100]            <1>       mov    [audio_user], al ; 0
46703                                  <1> snd_alloc_2:
46704 0000FBF5 89D3                    <1>       mov    ebx, edx
46705                                  <1>       ; 01/05/2017
46706 0000FBF7 BA00F0FFFF              <1>       mov    edx, ~PAGE_OFF ; truncating page offsets
46707                                  <1>                           ; for aligning to page borders
```

```
46708                                  <1>       ;and   eax, edx
46709 0000FBFC 21D3                    <1>       and    ebx, edx
46710 0000FBFE 21D1                    <1>       and    ecx, edx
46711                                  <1>       ; 15/05/2017
46712                                  <1>       ; EAX = Beginning address (physical)
46713                                  <1>       ; EAX = 0 -> Allocate mem block from the 1st proper aperture
46714                                  <1>       ; ECX = Number of bytes to be allocated
46715 0000FC00 E81E58FFFF              <1>       call   allocate_memory_block
46716 0000FC05 0F8268FFFFFF            <1>       jc     sound_buff_error
46717                                  <1>       ; EAX = Physical address of the allocated memory block
46718                                  <1>       ; ECX = Allocated bytes (as truncated to page border)
46719                                  <1>       ; EBX = Virtual address (as truncated to page border)
46720 0000FC0B 50                      <1>       push   eax
46721 0000FC0C 53                      <1>       push   ebx
46722 0000FC0D 51                      <1>       push   ecx
46723 0000FC0E E8605CFFFF              <1>       call   allocate_user_pages
46724 0000FC13 59                      <1>       pop    ecx
46725 0000FC14 5B                      <1>       pop    ebx
46726 0000FC15 58                      <1>       pop    eax
46727 0000FC16 7223                    <1>       jc     short snd_alloc_4  ; insufficient memory, buff error
46728                                  <1>       ; eax = physical address of the user's audio buffer
46729                                  <1>       ; ebx = virtual address of the user's audio buffer
46730                                  <1>       ; ecx = buffer size (in bytes)
46731 0000FC18 A3[64610100]            <1>       mov    [audio_p_buffer], eax
46732 0000FC1D 891D[60610100]          <1>       mov    [audio_buffer], ebx
46733 0000FC23 890D[68610100]          <1>       mov    [audio_buff_size], ecx
46734 0000FC29 8A15[B3030600]          <1>       mov    dl, [u.uno]
46735 0000FC2F 8815[75610100]          <1>       mov    [audio_user], dl
46736 0000FC35 A3[64030600]            <1>       mov    [u.r0], eax
46737                                  <1> snd_alloc_3:
46738 0000FC3A C3                      <1>       retn
46739                                  <1> snd_alloc_4:
46740                                  <1>       ; 15/05/2017
46741                                  <1>       ; EAX = Beginning address (physical)
46742                                  <1>       ; ECX = Number of bytes to be deallocated
46743 0000FC3B E8F059FFFF              <1>       call   deallocate_memory_block
46744 0000FC40 E92EFFFFFF              <1>       jmp    sound_buff_error  ; insufficient memory, buff error
46745                                  <1>
46746                                  <1> soundc_init:
46747                                  <1>       ; FUNCTION = 3
46748                                  <1>       ; bl = method (0= s.r.b., 1= callback, 2= auto incr s.r.b.)
46749                                  <1>       ; cl = signal response byte (initial or fixed) value
46750                                  <1>       ; edx = signal response byte or callback address
46751                                  <1>       ; 28/05/2017
46752                                  <1>       ; 12/05/2017, 20/05/2017
46753                                  <1>       ; 22/04/2017, 23/04/2017, 24/04/2017
46754                                  <1>       ; 13/04/2017, 14/04/2017, 16/04/2017, 21/04/2017
46755                                  <1>       ; 03/04/2017, 10/04/2017
46756                                  <1>
46757 0000FC45 A0[4D610100]            <1>       mov    al, [audio_device]
46758 0000FC4A 20C0                    <1>       and    al, al
46759 0000FC4C 7549                    <1>       jnz    short sndc_init_6
46760                                  <1>       ;
46761 0000FC4E C605[4C610100]00        <1>       mov    byte [audio_pci], 0
46762 0000FC55 52                      <1>       push   edx
46763 0000FC56 53                      <1>       push   ebx
46764 0000FC57 51                      <1>       push   ecx
46765 0000FC58 E850180000              <1>       call   DetectSB
46766 0000FC5D 7213                    <1>       jc     short sndc_init_8
46767 0000FC5F 66B80103                <1>       mov    ax, 0301h ; Sound Blaster 16
46768 0000FC63 EB1E                    <1>       jmp    short sndc_init_7
46769                                  <1>
46770                                  <1> sndc_init_11:
46771                                  <1>       ; 28/05/2017
46772 0000FC65 E8E5120000              <1>       call   DetectICH ; Detect AC'97 (ICH) Audio Controller
46773 0000FC6A 7217                    <1>       jc     short sndc_init_7
46774 0000FC6C 66B80203                <1>       mov    ax, 0302h ; Intel AC'97 Audio Device
46775 0000FC70 EB0B                    <1>       jmp    short sndc_init_12 ; (PCI device)
46776                                  <1>
46777                                  <1> sndc_init_8:
46778 0000FC72 E8E5120000              <1>       call   DetectVT8233
46779                                  <1>       ;jc    short sndc_init_7
46780 0000FC77 72EC                    <1>       jc     sndc_init_11 ; 28/05/2017
46781                                  <1>       ; eax = 0
46782 0000FC79 B003                    <1>       mov    al, 3  ; VIA VT8237R (VT3233) Audio Controller
46783 0000FC7B 88C4                    <1>       mov    ah, al
46784                                  <1>
46785                                  <1> sndc_init_12:
46786 0000FC7D FE05[4C610100]          <1>       inc    byte [audio_pci] ; = 1
46787                                  <1> sndc_init_7:
46788 0000FC83 59                      <1>       pop    ecx
46789 0000FC84 5B                      <1>       pop    ebx
46790 0000FC85 5A                      <1>       pop    edx
46791 0000FC86 0F82E0FEFFFF            <1>       jc     soundc_dev_err
46792                                  <1>       ;
46793 0000FC8C A2[4D610100]            <1>       mov    [audio_device], al
46794 0000FC91 8825[4E610100]          <1>       mov    [audio_mode], ah ; stereo (bit0), 16 bit (bit1) capability
46795                                  <1>
46796                                  <1> sndc_init_6:
46797 0000FC97 833D[60610100]00        <1>       cmp    dword [audio_buffer], 0
46798 0000FC9E 0F86CFFEFFFF            <1>       jna    sound_buff_error
46799                                  <1>
46800 0000FCA4 A0[B3030600]            <1>       mov    al, [u.uno]
46801 0000FCA9 8A25[75610100]          <1>       mov    ah, [audio_user]
46802 0000FCAF 08E4                    <1>       or     ah, ah
46803 0000FCB1 7418                    <1>       jz     short sndc_init0
46804 0000FCB3 38E0                    <1>       cmp    al, ah
46805 0000FCB5 7419                    <1>       je     short sndc_init1
46806                                  <1>
46807                                  <1> sndc_owner_error:
46808 0000FCB7 B80B000000              <1>       mov    eax, ERR_NOT_OWNER ; 'permission denied !' error
46809                                  <1> sndc_perm_error:
```

```
46810 0000FCBC A3[64030600]        <1>       mov    [u.r0], eax
46811 0000FCC1 A3[C8030600]        <1>       mov    [u.error], eax
46812 0000FCC6 E9C4C7FFFF          <1>       jmp    error
46813                              <1> sndc_init0:
46814 0000FCCB A2[75610100]        <1>       mov    [audio_user], al
46815                              <1> sndc_init1:
46816 0000FCD0 8915[78610100]      <1>       mov    [audio_cb_addr], edx
46817 0000FCD6 881D[76610100]      <1>       mov    [audio_cb_mode], bl
46818 0000FCDC 880D[77610100]      <1>       mov    [audio_srb], cl
46819                              <1>
46820                              <1>       ; 24/04/2017
46821 0000FCE2 803D[4D610100]03    <1>       cmp    byte [audio_device], 3 ; VT8233 (VT8237R)
46822 0000FCE9 7438                <1>       je     short sndc_init_9
46823                              <1>       ;ja    short soundc_respond_err ; temporary (28/05/2017)
46824 0000FCEB 803D[4D610100]01    <1>       cmp    byte [audio_device], 1 ; SB 16
46825 0000FCF2 7510                <1>       jne    short sndc_init_13
46826 0000FCF4 BB[BF160100]        <1>       mov    ebx, sb16_int_handler
46827                              <1>       ; Note: 'SbInit' is at 'Start to Play' stage
46828                              <1>       ; 20/05/2017
46829 0000FCF9 66C705[82610100]08- <1>       mov    word [audio_master_volume], 0808h ; 2/8
46830 0000FD01 08                  <1>
46831 0000FD02 EB3F                <1>       jmp    short sndc_init_10
46832                              <1> sndc_init_13:
46833                              <1>       ; 28/05/2017
46834 0000FD04 803D[4D610100]02    <1>       cmp    byte [audio_device], 2 ; AC 97 (ICH)
46835 0000FD0B 0F8569FEFFFF        <1>       jne    soundc_respond_err ; temporary (28/05/2017)
46836                              <1>
46837 0000FD11 E8141B0000          <1>       call   ac97_codec_config
46838 0000FD16 0F825EFEFFFF        <1>       jc     soundc_respond_err ; codec error !
46839                              <1>
46840 0000FD1C BB[111A0100]        <1>       mov    ebx, ac97_int_handler
46841 0000FD21 EB20                <1>       jmp    short sndc_init_10
46842                              <1>
46843                              <1> sndc_init_9:
46844                              <1>       ;call  reset_codec
46845                              <1>       ;; eax = 1
46846                              <1>       ;call  codec_io_w16 ; w32
46847 0000FD23 E8AB130000          <1>       call   init_codec ; 28/05/2017
46848 0000FD28 0F824CFEFFFF        <1>       jc     soundc_respond_err ; codec error !
46849                              <1>
46850 0000FD2E E8FF150000          <1>       call   channel_reset
46851                              <1>
46852                              <1>       ; setup the Codec (actually mixer registers)
46853 0000FD33 E8E6140000          <1>         call    codec_config  ; unmute codec, set rates.
46854 0000FD38 0F823CFEFFFF        <1>       jc     soundc_respond_err ; codec error !
46855                              <1>
46856 0000FD3E BB[8B120100]        <1>       mov    ebx, vt8233_int_handler
46857                              <1> sndc_init_10:
46858                              <1>       ; 13/04/2017
46859 0000FD43 A0[4F610100]        <1>       mov    al, [audio_intr] ; IRQ number
46860 0000FD48 E836FDFFFF          <1>       call   set_dev_IRQ_service
46861                              <1>
46862                              <1>       ; SETUP (audio) INTERRUPT CALLBACK SERVICE
46863 0000FD4D 8A1D[4F610100]      <1>       mov    bl, [audio_intr] ; IRQ number
46864 0000FD53 8A3D[76610100]      <1>       mov    bh, [audio_cb_mode]
46865 0000FD59 FEC7                <1>       inc    bh  ; 1 = Signal Response Byte method (fixed value)
46866                              <1>                   ; 2 = Callback service method
46867                              <1>                   ; 3 = Auto Increment S.R.B. method
46868 0000FD5B 8A0D[77610100]      <1>       mov    cl, [audio_srb]
46869 0000FD61 8B15[78610100]      <1>       mov    edx, [audio_cb_addr]
46870 0000FD67 A0[75610100]        <1>       mov    al, [audio_user]
46871                              <1>       ; 14/04/2017
46872 0000FD6C E860040000          <1>       call   set_irq_callback_service
46873                              <1>       ; 16/04/2017
46874 0000FD71 A3[64030600]        <1>       mov    [u.r0], eax
46875                              <1>       ;jnc   sysret
46876 0000FD76 7316                <1>       jnc    short sndc_init2 ; 21/04/2017
46877                              <1>       ;
46878 0000FD78 A3[C8030600]        <1>       mov    dword [u.error], eax
46879                              <1>
46880 0000FD7D A0[4F610100]        <1>       mov    al, [audio_intr] ; IRQ number
46881 0000FD82 31DB                <1>       xor    ebx, ebx ; reset IRQ handler address
46882 0000FD84 E8FAFCFFFF          <1>       call   set_dev_IRQ_service
46883                              <1>
46884 0000FD89 E901C7FFFF          <1>       jmp    error
46885                              <1>
46886                              <1> sndc_init2:
46887                              <1>       ; 21/04/2017
46888 0000FD8E 8B0D[68610100]      <1>       mov    ecx, [audio_buff_size] ; audio buffer size
46889 0000FD94 D1E1                <1>       shl    ecx, 1 ; *2
46890 0000FD96 A1[6C610100]        <1>       mov    eax, [audio_dma_buff]
46891 0000FD9B 21C0                <1>       and    eax, eax
46892 0000FD9D 7415                <1>       jz     short sndc_init3
46893                              <1>
46894 0000FD9F 8B15[70610100]      <1>       mov    edx, [audio_dmabuff_size] ; dma buffer size
46895 0000FDA5 39D1                <1>       cmp    ecx, edx
46896 0000FDA7 744D                <1>       je     short sndc_init5
46897                              <1>
46898 0000FDA9 87CA                <1>       xchg   ecx, edx
46899 0000FDAB E88058FFFF          <1>       call   deallocate_memory_block
46900 0000FDB0 87D1                <1>       xchg   edx, ecx
46901 0000FDB2 31C0                <1>       xor    eax, eax
46902                              <1> sndc_init3:
46903                              <1>       ; 12/05/2017
46904 0000FDB4 803D[4D610100]01    <1>       cmp    byte [audio_device], 1 ; SB 16
46905 0000FDBB 7515                <1>       jne    short sndc_init4
46906 0000FDBD C705[6C610100]-     <1>       mov    dword [audio_dma_buff], sb16_dma_buffer
46907 0000FDC3 [00000200]          <1>
46908 0000FDC7 C705[70610100]0000- <1>       mov    dword [audio_dmabuff_size], 65536
46909 0000FDCF 0100                <1>
46910                              <1>       ;xor   eax, eax
46911                              <1>       ;mov   [u.r0], eax ; 0 = no error, successful
```

```
46912 0000FDD1 C3                <1>       retn
46913                            <1>
46914                            <1> sndc_init4:
46915                            <1>       ; EAX = Beginning address (physical)
46916                            <1>       ; EAX = 0 -> Allocate mem block from the 1st proper aperture
46917                            <1>       ; ECX = Number of bytes to be allocated (>0)
46918 0000FDD2 E84C56FFFF        <1>       call   allocate_memory_block
46919 0000FDD7 0F8296FDFFFF      <1>       jc     sound_buff_error
46920                            <1>
46921                            <1>       ; set dma buffer address and size parameters
46922 0000FDDD A3[6C610100]      <1>       mov    [audio_dma_buff], eax ; dma buffer address
46923 0000FDE2 890D[70610100]    <1>       mov    [audio_dmabuff_size], ecx ; dma buffer size
46924                            <1> ;     ; EAX = Beginning (physical) addr of the allocated mem block
46925                            <1> ;     ; ECX = Num of allocated bytes (rounded up to page borders)
46926                            <1> ;     cmp    byte [audio_pci], 0 ; AC97 audio controller ?
46927                            <1> ;     ja     short sndc_init4
46928                            <1> ;
46929                            <1> ;     ; Sound Blaster 16 uses classic DMA
46930                            <1> ;     mov    edx, eax
46931                            <1> ;     add    edx, ecx
46932                            <1> ;     cmp    edx, 1000000h ; 1st 16 MB
46933                            <1> ;     jna    short sndc_init4
46934                            <1> ;
46935                            <1> ;     ; error !
46936                            <1> ;     ; restore Memory Allocation Table Content
46937                            <1> ;     ; EAX = Beginning address (physical)
46938                            <1> ;     ; ECX = Number of bytes to be deallocated
46939                            <1> ;     call   deallocate_memory_block
46940                            <1> ;     ; reset dma buffer address and size parameters
46941                            <1> ;     xor    eax, eax ; 0
46942                            <1> ;     mov    [audio_dma_buff], eax ; 0
46943                            <1> ;     mov    [audio_dmabuff_size], ecx ; 0
46944                            <1> ;     jmp    sound_buff_error
46945                            <1> ;
46946                            <1> ;sndc_init4:
46947 0000FDE8 803D[4D610100]03  <1>       cmp    byte [audio_device], 3
46948                            <1>       ;jne   short sndc_init5
46949 0000FDEF 7506              <1>       jne    short sndc_init14 ; 28/05/2017
46950 0000FDF1 E87D150000        <1>       call   set_vt8233_bdl
46951                            <1> sndc_init5:
46952                            <1>       ;sub   eax, eax ; 0
46953                            <1>       ;mov   [u.r0], eax ; 0 = no error, successful
46954 0000FDF6 C3                <1>       retn
46955                            <1> sndc_init14:
46956 0000FDF7 E8471B0000        <1>       call   set_ac97_bdl
46957                            <1>       ;jmp   short sndc_init5
46958 0000FDFC C3                <1>       retn
46959                            <1>
46960                            <1> sound_play:
46961                            <1>       ; FUNCTION = 4
46962                            <1>       ; bl = Mode
46963                            <1>       ;      bit 0 = mono/stereo (1 = stereo)
46964                            <1>       ;      bit 1 = 8 bit / 16 bit (1 = 16 bit)
46965                            <1>       ; cx = Sampling Rate (Hz)
46966                            <1>
46967                            <1>       ; 13/06/2017
46968                            <1>       ; Note: Even if Mode bits are not 11b,
46969                            <1>       ;     AC'97 Audio Controller (&Codec)
46970                            <1>       ;     will play audio samples as 16 bit, stereo
46971                            <1>       ;     samples.
46972                            <1>       ;     (Program must fill the audio buffer
46973                            <1>       ;     as required; 8 bit samples must be converted
46974                            <1>       ;     to 16 bit samples and mono samples must be
46975                            <1>       ;     converted to stereo samples...)
46976                            <1>       ;
46977                            <1>       ; 28/05/2017
46978                            <1>       ; 15/05/2017, 20/05/2017
46979                            <1>       ; 21/04/2017, 24/04/2017
46980                            <1>       ; ... device check at first
46981 0000FDFD A0[4D610100]      <1>       mov    al, [audio_device]
46982 0000FE02 08C0              <1>       or     al, al ; 0 ; pc speaker or invalid
46983 0000FE04 0F84911FFFFF      <1>       jz     beeper_gfx ; 'video.s' ; temporary !
46984                            <1> ;     cmp    al, 3 ; VIA VT 8237R (vt8233)
46985                            <1> ;     je     short snd_play_1
46986                            <1> ;     cmp    al, 1 ; SB 16
46987                            <1> ;     jne    soundc_dev_err ; temporary !
46988                            <1> ;snd_play_0:
46989                            <1>       ; ... buffer & (buffer) owner check at second
46990 0000FE0A 833D[60610100]00  <1>       cmp    dword [audio_buffer], 0
46991 0000FE11 0F865CFDFFFF      <1>       jna    sound_buff_error
46992 0000FE17 A0[B3030600]      <1>       mov    al, [u.uno]
46993 0000FE1C 3A05[75610100]    <1>       cmp    al, [audio_user]
46994 0000FE22 0F858FFEFFFF      <1>       jne    sndc_owner_error
46995                            <1>
46996 0000FE28 66890D[7E610100]  <1>       mov    [audio_freq], cx ; sample frequency (Hertz)
46997 0000FE2F 88D8              <1>       mov    al, bl
46998 0000FE31 2401              <1>       and    al, 1 ; mono/stereo (1= stereo)
46999 0000FE33 FEC0              <1>       inc    al ; channels
47000 0000FE35 A2[7D610100]      <1>       mov    [audio_stmo], al ; sound channels (1 or 2)
47001 0000FE3A B008              <1>       mov    al, 8
47002 0000FE3C F6C302            <1>       test   bl, 2 ; bits per sample (1= 16 bit)
47003 0000FE3F 7402              <1>       jz     short snd_play_bps
47004 0000FE41 D0E0              <1>       shl    al, 1
47005                            <1> snd_play_bps:
47006 0000FE43 A2[7C610100]      <1>       mov    [audio_bps], al
47007                            <1>       ; Transfer ring 3 (user's) audio buffer content to dma buffer
47008 0000FE48 8B3D[6C610100]    <1>       mov    edi, [audio_dma_buff] ; dma buffer (ring 0)
47009 0000FE4E 09FF              <1>       or     edi, edi
47010 0000FE50 0F841DFDFFFF      <1>       jz     sound_buff_error
47011 0000FE56 8B35[64610100]    <1>       mov    esi, [audio_p_buffer] ; physical address (ring 3)
47012 0000FE5C 8B0D[68610100]    <1>       mov    ecx, [audio_buff_size] ; 15/05/2017
47013                            <1>       ;rep   movsb
```

```
47014 0000FE62 C1E902              <1>       shr    ecx, 2
47015 0000FE65 F3A5                <1>       rep    movsd
47016                              <1>       ; 20/05/2017
47017 0000FE67 C605[74610100]01    <1>       mov    byte [audio_flag], 1 ; next half (on next time)
47018                              <1>
47019                              <1>       ; 24/04/2017
47020 0000FE6E A0[4D610100]        <1>       mov    al, [audio_device]
47021 0000FE73 3C03                <1>       cmp    al, 3 ; VT8233 (VT8237R)
47022 0000FE75 7410                <1>       je     short snd_play_1
47023 0000FE77 3C01                <1>       cmp    al, 1 ; Sound Blaster 16
47024 0000FE79 7512                <1>       jne    short snd_play_2  ; 28/05/2017
47025 0000FE7B E8FB160000          <1>       call   SbInit_play
47026 0000FE80 0F82F4FCFFFF        <1>       jc     soundc_respond_err
47027 0000FE86 C3                  <1>       retn
47028                              <1>
47029                              <1> snd_play_1:
47030 0000FE87 E817150000          <1>       call   vt8233_start_play
47031 0000FE8C C3                  <1>       retn
47032                              <1>
47033                              <1> snd_play_2:
47034                              <1>       ; 28/05/2017
47035                              <1>       ;cmp   al, 2 ; AC'97
47036                              <1>       ;jne   short snd_play_3
47037                              <1>
47038 0000FE8D E8E51A0000          <1>       call   ac97_start_play
47039 0000FE92 C3                  <1>       retn
47040                              <1>
47041                              <1> ;snd_play_3:
47042                              <1> ;     ;call  hda_start_play
47043                              <1> ;     retn
47044                              <1>
47045                              <1> sound_pause:
47046                              <1>       ; FUNCTION = 5
47047                              <1>       ; Pause
47048                              <1>       ; 28/05/2017
47049                              <1>       ; 24/04/2017
47050                              <1>       ; 22/04/2017
47051 0000FE93 E814030000          <1>       call   snd_dev_check
47052 0000FE98 7275                <1>       jc     short snd_nothing ; temporary.
47053 0000FE9A E81A030000          <1>       call   snd_buf_check
47054 0000FE9F 726E                <1>       jc     short snd_nothing ; temporary.
47055 0000FEA1 A0[4D610100]        <1>       mov    al, [audio_device]
47056 0000FEA6 3C03                <1>       cmp    al, 3 ; VIA VT 8237R (vt8233)
47057 0000FEA8 7409                <1>       je     short snd_pause_1
47058 0000FEAA 3C01                <1>       cmp    al, 1 ; Sound Blaster 16
47059 0000FEAC 750A                <1>       jne    short snd_pause_2 ; 28/05/2017
47060 0000FEAE E9A9180000          <1>       jmp    sb16_pause
47061                              <1> snd_pause_1:
47062 0000FEB3 E9A9150000          <1>       jmp    vt8233_pause
47063                              <1> snd_pause_2:
47064                              <1>       ; 28/05/2017
47065                              <1>       ;cmp   al, 2 ; AC'97
47066                              <1>       ;jne   short snd_nothing ; temporary.
47067 0000FEB8 E91C1C0000          <1>       jmp    ac97_pause
47068                              <1>
47069                              <1> sound_continue:
47070                              <1>       ; FUNCTION = 6
47071                              <1>       ; Continue to play
47072                              <1>       ; 28/05/2017
47073                              <1>       ; 22/04/2017
47074 0000FEBD E8EA020000          <1>       call   snd_dev_check
47075 0000FEC2 724B                <1>       jc     short snd_nothing ; temporary.
47076 0000FEC4 E8F0020000          <1>       call   snd_buf_check
47077 0000FEC9 7244                <1>       jc     short snd_nothing ; temporary.
47078 0000FECB A0[4D610100]        <1>       mov    al, [audio_device]
47079 0000FED0 3C03                <1>       cmp    al, 3 ; VIA VT 8237R (vt8233)
47080 0000FED2 7409                <1>       je     short snd_cont_1
47081 0000FED4 3C01                <1>       cmp    al, 1 ; Sound Blaster 16
47082 0000FED6 750A                <1>       jne    short snd_cont_2 ; 28/05/2017
47083 0000FED8 E9A2180000          <1>       jmp    sb16_continue
47084                              <1> snd_cont_1:
47085 0000FEDD E92C150000          <1>       jmp    vt8233_play
47086                              <1> snd_cont_2:
47087                              <1>       ; 28/05/2017
47088                              <1>       ;cmp   al, 2 ; AC'97
47089                              <1>       ;jne   short snd_nothing ; temporary.
47090 0000FEE2 E9E61A0000          <1>       jmp    ac97_play
47091                              <1>
47092                              <1> sound_stop:
47093                              <1>       ; FUNCTION = 7
47094                              <1>       ; Stop playing
47095                              <1>       ; 28/05/2017
47096                              <1>       ; 24/05/2017
47097                              <1>       ; 21/04/2017, 22/04/2017, 24/04/2017
47098 0000FEE7 E8C0020000          <1>       call   snd_dev_check
47099 0000FEEC 7221                <1>       jc     short snd_nothing ; temporary.
47100                              <1>       ;call  snd_buf_check
47101 0000FEEE E8CF020000          <1>       call   snd_user_check ; 24/05/2017
47102 0000FEF3 721A                <1>       jc     short snd_nothing ; temporary.
47103                              <1>
47104 0000FEF5 A0[4D610100]        <1>       mov    al, [audio_device]
47105 0000FEFA 3C03                <1>       cmp    al, 3 ; VIA VT 8237R (vt8233)
47106 0000FEFC 0F8468140000        <1>       je     vt8233_stop
47107                              <1>       ; 28/05/2017
47108                              <1>       ;ja    short snd_nothing
47109 0000FF02 3C01                <1>       cmp    al, 1 ; Sound Blaster 16
47110 0000FF04 0F8498180000        <1>       je     sb16_stop
47111                              <1>       ;cmp   al, 2
47112                              <1>       ;je    short ac97_stop
47113 0000FF0A E9CE1B0000          <1>       jmp    ac97_stop ; temporary.
47114                              <1>       ;jmp   hda_stop
47115                              <1>
```

```
47116                                   <1> snd_nothing:
47117                                   <1>       ; 21/04/2017
47118 0000FF0F C3                       <1>       retn
47119                                   <1>
47120                                   <1> soundc_reset:
47121                                   <1>       ; FUNCTION = 8
47122                                   <1>       ; Reset Audio Controller
47123                                   <1>       ; 28/05/2017
47124                                   <1>       ; 22/04/2017
47125 0000FF10 E897020000               <1>       call   snd_dev_check
47126 0000FF15 72F8                     <1>       jc     snd_nothing ; temporary.
47127 0000FF17 E89D020000               <1>       call   snd_buf_check
47128 0000FF1C 72F1                     <1>       jc     snd_nothing ; temporary.
47129                                   <1>
47130 0000FF1E A0[4D610100]             <1>       mov    al, [audio_device]
47131 0000FF23 3C03                     <1>       cmp    al, 3 ; VIA VT 8237R (vt8233)
47132 0000FF25 0F8444150000             <1>       je     vt8233_reset
47133 0000FF2B 77E2                     <1>       ja     short snd_nothing ; temporary.
47134                                   <1>       ;ja    hda_reset
47135 0000FF2D 3C01                     <1>       cmp    al, 1 ; Sound Blaster 16
47136 0000FF2F 0F85231C0000             <1>       jne    ac97_reset
47137 0000FF35 E8BA180000               <1>       call   sb16_reset
47138 0000FF3A 0F823AFCFFFF             <1>       jc     soundc_respond_err
47139 0000FF40 C3                       <1>       retn
47140                                   <1>
47141                                   <1> soundc_cancel:
47142                                   <1>       ; FUNCTION = 9
47143                                   <1>       ; Cancel audio callback service
47144                                   <1>       ; 22/04/2017
47145 0000FF41 A0[75610100]             <1>       mov    al, [audio_user]
47146 0000FF46 3A05[B3030600]           <1>       cmp    al, [u.uno]
47147 0000FF4C 75C1                     <1>       jne    short snd_nothing
47148                                   <1>       ; RESET (audio) INTERRUPT CALLBACK SERVICE
47149 0000FF4E 8A1D[4F610100]           <1>       mov    bl, [audio_intr] ; IRQ number
47150 0000FF54 A0[B3030600]             <1>       mov    al, [u.uno]
47151 0000FF59 28FF                     <1>       sub    bh, bh ; 0 ; unlink IRQ from user service
47152 0000FF5B E871020000               <1>       call   set_irq_callback_service
47153 0000FF60 0F8256FDFFFF             <1>       jc     sndc_perm_error ; 'permission denied' error
47154 0000FF66 C3                       <1>       retn
47155                                   <1>
47156                                   <1> sound_dalloc:
47157                                   <1>       ; FUNCTION = 10
47158                                   <1>       ; Deallocate (ring 3) audio buffer
47159                                   <1>       ; 22/04/2017
47160 0000FF67 A0[75610100]             <1>       mov    al, [audio_user]
47161 0000FF6C 3A05[B3030600]           <1>       cmp    al, [u.uno]
47162 0000FF72 759B                     <1>       jne    short snd_nothing
47163 0000FF74 8B1D[60610100]           <1>       mov    ebx, [audio_buffer]
47164                                   <1>       ;or    ebx, ebx
47165                                   <1>       ;jz    short snd_nothing
47166 0000FF7A 8B0D[68610100]           <1>       mov    ecx, [audio_buff_size]
47167 0000FF80 E8F957FFFF               <1>       call   deallocate_user_pages
47168 0000FF85 31C0                     <1>       xor    eax, eax
47169 0000FF87 A3[60610100]             <1>       mov    [audio_buffer], eax ; 0
47170 0000FF8C A2[75610100]             <1>       mov    [audio_user], al ; 0
47171 0000FF91 C3                       <1>       retn
47172                                   <1>
47173                                   <1> sound_volume:
47174                                   <1>       ; FUNCTION = 11
47175                                   <1>       ; Set sound volume level
47176                                   <1>       ; 28/05/2017
47177                                   <1>       ; 20/05/2017
47178                                   <1>       ; 22/04/2017, 24/04/2017
47179                                   <1>       ; bl = component (0 = master/playback/lineout volume)
47180                                   <1>       ; cl = left channel volume level (0 to 31)
47181                                   <1>       ; ch = right channel volume level (0 to 31)
47182                                   <1>
47183 0000FF92 80FB80                   <1>       cmp    bl, 80h
47184 0000FF95 720E                     <1>       jb     short snd_vol_1
47185 0000FF97 0F8772FFFFFF             <1>       ja     snd_nothing ; temporary.
47186                                   <1>       ; Set volume level for next play (BL>= 80h)
47187 0000FF9D 66890D[82610100]         <1>       mov    [audio_master_volume], cx
47188 0000FFA4 C3                       <1>       retn
47189                                   <1> snd_vol_1:
47190                                   <1>       ; set volume level immediate (BL< 80h)
47191 0000FFA5 80FB00                   <1>       cmp    bl, 0
47192 0000FFA8 0F8761FFFFFF             <1>       ja     snd_nothing ; temporary.
47193                                   <1>
47194 0000FFAE E8F9010000               <1>       call   snd_dev_check
47195 0000FFB3 0F8256FFFFFF             <1>       jc     snd_nothing ; temporary.
47196 0000FFB9 E8FB010000               <1>       call   snd_buf_check
47197 0000FFBE 0F824BFFFFFF             <1>       jc     snd_nothing ; temporary.
47198                                   <1>
47199 0000FFC4 A0[4D610100]             <1>       mov    al, [audio_device]
47200 0000FFC9 3C03                     <1>       cmp    al, 3 ; VIA VT 8237R (vt8233)
47201 0000FFCB 0F84B7140000             <1>       je     vt8233_volume
47202                                   <1>       ; 28/05/2017
47203 0000FFD1 0F8738FFFFFF             <1>       ja     snd_nothing ; temporary.
47204                                   <1>       ;ja    hda_volume
47205                                   <1>       ; Sound Blaster 16
47206 0000FFD7 3C01                     <1>       cmp    al, 1 ; SB 16
47207 0000FFD9 0F844F170000             <1>       je     sb16_volume
47208 0000FFDF E9051A0000               <1>       jmp    ac97_volume
47209                                   <1>
47210                                   <1> soundc_disable:
47211                                   <1>       ; FUNCTION = 12
47212                                   <1>       ; Disable audio device (and unlink DMA memory)
47213                                   <1>       ; 28/05/2017
47214                                   <1>       ; 24/05/2017
47215                                   <1>       ; 22/04/2017
47216 0000FFE4 E8C3010000               <1>       call   snd_dev_check
47217 0000FFE9 0F827DFBFFFF             <1>       jc     soundc_dev_err ; temporary.
```

```
47218                                  <1>       ;call  snd_buf_check
47219                                  <1>       ;jc    sndc_owner_error ; temporary.
47220                                  <1>
47221 0000FFEF A0[4D610100]            <1>       mov    al, [audio_device]
47222 0000FFF4 3C03                    <1>       cmp    al, 3 ; VIA VT 8237R (vt8233)
47223 0000FFF6 7418                    <1>       je     short snd_disable_1
47224 0000FFF8 0F8711FFFFFF            <1>       ja     snd_nothing ; temporary.
47225 0000FFFE 3C01                    <1>       cmp    al, 1 ; Sound Blaster 16
47226 00010000 7507                    <1>       jne    short snd_disable_0
47227 00010002 E89B170000              <1>       call   sb16_stop
47228 00010007 EB0C                    <1>       jmp    short snd_disable_2
47229                                  <1> snd_disable_0:
47230 00010009 E8CF1A0000              <1>       call   ac97_stop
47231 0001000E EB05                    <1>       jmp    short snd_disable_2
47232                                  <1> snd_disable_1:
47233 00010010 E855130000              <1>       call   vt8233_stop
47234                                  <1> snd_disable_2:
47235 00010015 A0[4F610100]            <1>       mov    al, [audio_intr]
47236 0001001A 29DB                    <1>       sub    ebx, ebx ; 0 = reset
47237 0001001C E862FAFFFF              <1>       call   set_dev_IRQ_service
47238                                  <1>
47239                                  <1>       ;mov   al, [audio_intr]
47240 00010021 28E4                    <1>       sub    ah, ah ; 0 = reset
47241 00010023 E8C4F6FFFF              <1>       call   set_hardware_int_vector
47242                                  <1>
47243 00010028 31C0                    <1>       xor    eax, eax
47244 0001002A A2[4D610100]            <1>       mov    byte [audio_device], al
47245 0001002F A2[4F610100]            <1>       mov    byte [audio_intr], al
47246 00010034 8705[6C610100]          <1>       xchg   eax, [audio_dma_buff]
47247                                  <1>       ; 24/05/2017
47248                                  <1>       ;or    eax, eax
47249                                  <1>       ;jz    short snd_disable_3
47250                                  <1>       ;cmp   eax, sb16_dma_buffer ; default DMA buffer
47251                                  <1>       ;je    short snd_disable_3
47252 0001003A 803D[4C610100]00        <1>       cmp    byte [audio_pci], 0 ; AC97 audio controller ?
47253 00010041 7612                    <1>       jna    short snd_disable_3
47254 00010043 C605[4C610100]00        <1>       mov    byte [audio_pci], 0
47255                                  <1>       ;sub   ecx, ecx
47256                                  <1>       ;xchg  ecx, [audio_dmabuff_size]
47257 0001004A 8B0D[70610100]          <1>       mov    ecx, [audio_dmabuff_size]
47258 00010050 E8DB55FFFF              <1>       call   deallocate_memory_block
47259                                  <1> snd_disable_3:
47260 00010055 C3                      <1>       retn
47261                                  <1>
47262                                  <1> sound_dma_map:
47263                                  <1>       ; FUNCTION = 13
47264                                  <1>       ; Map audio dma buff addr to user's buffer addr
47265                                  <1>       ; 12/05/2017
47266 00010056 21C9                    <1>       and    ecx, ecx
47267 00010058 0F8415FBFFFF            <1>       jz     sound_buff_error
47268 0001005E 803D[4D610100]01        <1>       cmp    byte [audio_device], 1
47269 00010065 7229                    <1>       jb     short snd_dma_map_1
47270                                  <1> snd_dma_map_0:
47271 00010067 A1[6C610100]            <1>       mov    eax, [audio_dma_buff]
47272 0001006C 21C0                    <1>       and    eax, eax
47273 0001006E 7420                    <1>       jz     short snd_dma_map_1
47274                                  <1>       ;
47275 00010070 8A1D[75610100]          <1>       mov    bl, [audio_user]
47276 00010076 08DB                    <1>       or     bl, bl
47277 00010078 7416                    <1>       jz     short snd_dma_map_1
47278 0001007A 3A1D[B3030600]          <1>       cmp    bl, [u.uno]
47279 00010080 0F8531FCFFFF            <1>       jne    sndc_owner_error
47280                                  <1>       ;
47281 00010086 8B1D[70610100]          <1>       mov    ebx, [audio_dmabuff_size]
47282 0001008C 21DB                    <1>       and    ebx, ebx
47283 0001008E 750A                    <1>       jnz    short snd_dma_map_2
47284                                  <1> snd_dma_map_1:
47285 00010090 B8[00000200]            <1>       mov    eax, sb16_dma_buffer
47286 00010095 BB00000100              <1>       mov    ebx, 65536
47287                                  <1> snd_dma_map_2:
47288 0001009A 81C1FF0F0000            <1>       add    ecx, PAGE_SIZE-1 ; 4095
47289 000100A0 6681E100F0              <1>       and    cx, ~PAGE_OFF ; not 4095
47290 000100A5 39D9                    <1>       cmp    ecx, ebx
47291 000100A7 0F87C6FAFFFF            <1>       ja     sound_buff_error
47292 000100AD 50                      <1>       push   eax
47293 000100AE 89D3                    <1>       mov    ebx, edx
47294 000100B0 C1E90C                  <1>       shr    ecx, 12 ; byte count to page count
47295                                  <1>       ; eax = physical address of (audio) dma buffer
47296                                  <1>       ; ebx = virtual address of (audio) dma buffer (user's pgdir)
47297                                  <1>       ; ecx = page count (>0)
47298 000100B3 E8E855FFFF              <1>       call   direct_memory_access
47299 000100B8 58                      <1>       pop    eax
47300 000100B9 0F82B4FAFFFF            <1>       jc     sound_buff_error
47301 000100BF A3[64030600]            <1>       mov    [u.r0], eax
47302 000100C4 C3                      <1>       retn
47303                                  <1>
47304                                  <1> soundc_info:
47305                                  <1>       ; FUNCTION = 14
47306                                  <1>       ; Get Audio Controller Info
47307                                  <1>       ; 10/06/2017
47308                                  <1>       ; 05/06/2017
47309 000100C5 20DB                    <1>       and    bl, bl ; 0
47310 000100C7 740A                    <1>       jz     short sndc_info_0
47311                                  <1>       ; invalid parameter !
47312 000100C9 B817000000              <1>       mov    eax, ERR_INV_PARAMETER ; 23
47313                                  <1> ;sndc_inf_error:
47314                                  <1> ;     mov    [u.r0], eax
47315                                  <1> ;     mov    [u.error], eax
47316                                  <1> ;     jmp    error
47317 000100CE E9ACFAFFFF              <1>       jmp    sysaudio_err
47318                                  <1>
47319                                  <1> sndc_info_0:
```

```
47320 000100D3 E8D4000000          <1>       call   snd_dev_check
47321 000100D8 0F828EFAFFFF        <1>       jc     soundc_dev_err
47322                              <1>
47323 000100DE 8B1D[58610100]      <1>       mov    ebx, [audio_vendor]
47324 000100E4 8B0D[54610100]      <1>       mov    ecx, [audio_dev_id]
47325                              <1>       ;mov   al,  [audio_device]
47326 000100EA 3C02                <1>       cmp    al, 2 ; AC'97 (ICH)
47327 000100EC 7513                <1>       jne    short sndc_info_1
47328                              <1>       ; Intel AC97 (ICH) Audio Controller (=2)
47329 000100EE 668B15[86610100]    <1>       mov    dx, [NABMBAR]
47330 000100F5 C1E210              <1>       shl    edx, 16
47331 000100F8 668B15[84610100]    <1>       mov    dx, [NAMBAR]
47332 000100FF EB07                <1>       jmp    short sndc_info_2
47333                              <1> sndc_info_1:
47334                              <1>       ; 05/06/2017
47335                              <1>       ; Note: Intel HDA code (here) is not ready yet!
47336                              <1>       ; !!! SB16 or VT8233 (VT8237R) !!!
47337 00010101 0FB715[52610100]    <1>       movzx  edx, word [audio_io_base]
47338                              <1> sndc_info_2:
47339 00010108 88C4                <1>       mov    ah, al ;  [audio_device]
47340 0001010A A0[4F610100]        <1>       mov    al, [audio_intr]
47341                              <1>
47342                              <1>       ; EAX = IRQ Number in AL
47343                              <1>       ;       Audio Device Number in AH
47344                              <1>       ; EBX = DEV/VENDOR ID
47345                              <1>       ;       (DDDDDDDDDDDDDDDDVVVVVVVVVVVVVVVV)
47346                              <1>       ; ECX = BUS/DEV/FN
47347                              <1>       ;       (00000000BBBBBBBBDDDDDFFF00000000)
47348                              <1>       ; EDX = NABMBAR/NAMBAR (for AC97)
47349                              <1>       ;       (Low word, DX = NAMBAR address)
47350                              <1>       ; EDX = Base IO Addr (DX) for SB16 & VT8233
47351                              <1>
47352                              <1>       ; 10/06/2017
47353 0001010F A3[64030600]        <1>       mov    [u.r0], eax
47354 00010114 8B2D[60030600]      <1>       mov    ebp, [u.usp]
47355 0001011A 895D10              <1>       mov    [ebp+16], ebx  ; ebx
47356 0001011D 895514              <1>       mov    [ebp+20], edx  ; edx
47357 00010120 894D18              <1>       mov    [ebp+24], ecx  ; ecx
47358                              <1>
47359 00010123 C3                  <1>       retn
47360                              <1>
47361                              <1> sound_data:
47362                              <1>       ; FUNCTION = 15
47363                              <1>       ; Get Current Sound data for graphics
47364                              <1>       ; 22/06/2017
47365                              <1>       ;
47366 00010124 E883000000          <1>       call   snd_dev_check
47367 00010129 0F823DFAFFFF        <1>       jc     soundc_dev_err ; Device not ready !
47368                              <1>
47369 0001012F 80FB00              <1>       cmp    bl, 0
47370 00010132 760A                <1>       jna    short sound_data_0
47371                              <1>
47372                              <1>       ; Only PCM OUT buffer data is valid for now!
47373 00010134 B817000000          <1>       mov    eax, ERR_INV_PARAMETER  ; 23
47374 00010139 E941FAFFFF          <1>       jmp    sysaudio_err
47375                              <1>
47376                              <1> sound_data_0:
47377 0001013E A1[6C610100]        <1>       mov    eax, [audio_dma_buff]
47378 00010143 09C0                <1>       or     eax, eax
47379 00010145 0F8428FAFFFF        <1>       jz     sound_buff_error
47380                              <1>
47381 0001014B 803D[4D610100]04    <1>       cmp    byte [audio_device], 4 ; Intel HDA
47382 00010152 744F                <1>       je     short sound_data_4 ; temporary ! (22/06/2017)
47383                              <1>
47384 00010154 21C9                <1>       and    ecx, ecx
47385                              <1>       ;jnz   short sound_data_1 ; sample tranfer
47386                              <1>
47387                              <1>       ; Return only DMA Buffer pointer/offset...
47388                              <1>       ; (If DMA Buffer has been mapped to user's
47389                              <1>       ;  memory space; program can get graphics
47390                              <1>       ;  data by using only this pointer value.)
47391                              <1>
47392                              <1>       ;call  get_dma_buffer_offset
47393                              <1>       ;; eax = DMA buffer offset
47394                              <1>       ;;     (!not half buffer offset!)
47395                              <1>       ;mov   [u.r0], eax
47396                              <1>       ;retn
47397                              <1>
47398 00010156 0F84441B0000        <1>       jz     get_dma_buffer_offset
47399                              <1>
47400                              <1> sound_data_1:
47401                              <1>       ;mov   eax, [audio_dmabuff_size]
47402                              <1>       ;shr   eax, 1 ; half buffer size
47403                              <1>       ;cmp   ecx, eax
47404                              <1>       ;ja    short sound_buff_error
47405                              <1>
47406 0001015C 3B0D[70610100]      <1>       cmp    ecx, [audio_dmabuff_size]
47407 00010162 0F870BFAFFFF        <1>       ja     sound_buff_error
47408                              <1>
47409 00010168 89D0                <1>       mov    eax, edx
47410 0001016A 25FF0F0000          <1>       and    eax, PAGE_OFF ; 4095 (0FFFh)
47411 0001016F 81F900100000        <1>       cmp    ecx, 4096
47412 00010175 7605                <1>       jna    short sound_data_2
47413 00010177 B900100000          <1>       mov    ecx, 4096 ; max. 1 page
47414                              <1> sound_data_2:
47415 0001017C 01C8                <1>       add    eax, ecx
47416 0001017E 3D00100000          <1>       cmp    eax, 4096
47417 00010183 7606                <1>       jna    short sound_data_3
47418 00010185 6625FF0F            <1>       and    ax, PAGE_OFF ; 4095 (0FFFh)
47419 00010189 29C1                <1>       sub    ecx, eax
47420                              <1>       ; here, ECX has been adjusted to fit
47421                              <1>       ;       in page border..  (<= 4096, >0)
```

```
47422                                  <1> sound_data_3:
47423 0001018B 51                      <1>         push   ecx
47424 0001018C 52                      <1>         push   edx
47425 0001018D 89D3                    <1>         mov    ebx, edx
47426 0001018F E8FA50FFFF              <1>         call   get_physical_addr
47427 00010194 5A                      <1>         pop    edx
47428 00010195 59                      <1>         pop    ecx
47429 00010196 0F82D7F9FFFF            <1>         jc     sound_buff_error
47430                                  <1>
47431                                  <1>         ; eax = physical address of user's buffer
47432 0001019C 89C3                    <1>         mov    ebx, eax
47433                                  <1>         ; ecx = byte (transfer) count
47434                                  <1>         ;call  get_current_sound_data
47435                                  <1>         ;retn
47436 0001019E E9721A0000              <1>         jmp    get_current_sound_data
47437                                  <1>
47438                                  <1> sound_data_4:
47439                                  <1>         ; Intel HDA code is not ready yet !
47440                                  <1>         ; 22/06/2017
47441 000101A3 31C0                    <1>         xor    eax, eax
47442 000101A5 48                      <1>         dec    eax
47443 000101A6 A3[64030600]            <1>         mov    [u.r0], eax ; 0FFFFFFFFh
47444 000101AB C3                      <1>         retn
47445                                  <1>
47446                                  <1> snd_dev_check:
47447                                  <1>         ; 10/06/2017
47448                                  <1>         ; 05/06/2017
47449                                  <1>         ; 24/05/2017
47450                                  <1>         ; 22/04/2017
47451                                  <1>         ; 21/04/2017
47452                                  <1>         ; ... device check at first
47453 000101AC A0[4D610100]            <1>         mov    al, [audio_device]
47454 000101B1 3C01                    <1>         cmp    al, 1 ; SB 16
47455 000101B3 7203                    <1>         jb     short snd_dev_chk_retn ; error !
47456                                  <1>         ;cmp   al, 4 ; Intel HDA
47457                                  <1>         ;ja    short snd_dbchk_stc ; invalid !
47458                                  <1>         ; 10/06/2017
47459 000101B5 3C05                    <1>         cmp    al, 5
47460 000101B7 F5                      <1>         cmc
47461                                  <1> snd_dev_chk_retn:
47462 000101B8 C3                      <1>         retn
47463                                  <1>
47464                                  <1> snd_buf_check:
47465                                  <1>         ; 10/06/2017
47466                                  <1>         ; 22/04/2017
47467                                  <1>         ; 21/04/2017
47468                                  <1>         ; ... buffer & (buffer) owner check at second
47469 000101B9 833D[60610100]00        <1>         cmp    dword [audio_buffer], 0
47470 000101C0 760D                    <1>         jna    short snd_dbchk_stc
47471                                  <1> snd_user_check:
47472 000101C2 A0[B3030600]            <1>         mov    al, [u.uno]
47473 000101C7 3A05[75610100]          <1>         cmp    al, [audio_user]
47474                                  <1>         ;jne   short snd_dbchk_stc
47475                                  <1>         ;retn
47476 000101CD 74E9                    <1>         je     short snd_dev_chk_retn
47477                                  <1>
47478                                  <1> snd_dbchk_stc:
47479 000101CF F9                      <1>         stc
47480 000101D0 C3                      <1>         retn
47481                                  <1>
47482                                  <1> set_irq_callback_service:
47483                                  <1>         ; 10/06/2017
47484                                  <1>         ; 12/05/2017
47485                                  <1>         ; 24/04/2017
47486                                  <1>         ; 22/04/2017
47487                                  <1>         ; caller: 'syscalbac' or 'sysaudio' or ...
47488                                  <1>         ; 13/04/2017, 14/04/2017, 17/04/2017
47489                                  <1>         ; 24/02/2017, 26/02/2017, 28/02/2017
47490                                  <1>         ; 21/02/2017 - TRDOS 386 (TRDOS v2.0)
47491                                  <1>         ;
47492                                  <1>         ; Link or unlink IRQ callback service to/from user (ring 3)
47493                                  <1>         ;
47494                                  <1>         ; INPUT ->
47495                                  <1>         ;     If AL = 0, the caller is 'syscalbac';
47496                                  <1>         ;        otherwise, the caller is 'sysaudio' or ...
47497                                  <1>         ;        (AL = user number)
47498                                  <1>         ;
47499                                  <1>         ;     BL = IRQ number (Hardware interrupt request number)
47500                                  <1>         ;          (0 t0 15 but IRQ 0,1,2,6,8,14,15 are prohibited)
47501                                  <1>         ;          IRQ numbers 3,4,5,7,9,10,11,12,13 are valid
47502                                  <1>         ;          (numbers >15 are invalid)
47503                                  <1>         ;
47504                                  <1>         ;     BH = 0 = Unlink IRQ (in BL) from user (ring 3) service
47505                                  <1>         ;          1 = Link IRQ by using Signal Response Byte method
47506                                  <1>         ;          2 = Link IRQ by using Callback service method
47507                                  <1>         ;          3 = Link IRQ by using Auto Increment S.R.B. method
47508                                  <1>         ;         >3 = invalid
47509                                  <1>         ;             (syscallback version will return to user)
47510                                  <1>         ;
47511                                  <1>         ;     CL = Signal Return/Response Byte value
47512                                  <1>         ;
47513                                  <1>         ;     If BH = 2, kernel will put a counter value
47514                                  <1>         ;               (into the S.R.B. addr)
47515                                  <1>         ;              between 0 to 255. (start value = CL+1)
47516                                  <1>         ;
47517                                  <1>         ;     NOTE: counter value, for example: even and odd numbers
47518                                  <1>         ;           may be used for -audio- DMA buffer switch
47519                                  <1>         ;           within double buffer method, etc.
47520                                  <1>         ;
47521                                  <1>         ;     EDX = Signal return (Response) byte address
47522                                  <1>         ;                           - or -
47523                                  <1>         ;           Interrupt/Callback service/routine address
```

```
47524                                  <1>       ;
47525                                  <1>       ;           (virtual address in user's memory space)
47526                                  <1>       ;
47527                                  <1>       ; OUTPUT ->
47528                                  <1>       ;     CF = 0 & EAX = 0 -> Successful setting
47529                                  <1>       ;     CF = 1 & EAX > 0 -> IRQ is prohibited or locked
47530                                  <1>       ;                         by another process
47531                                  <1>       ;                   eax = ERR_PERM_DENIED -> prohibited or locked
47532                                  <1>       ;                   eax = ERR_INV_PARAMETER ->
47533                                  <1>       ;                         invalid parameter/option or bad address
47534                                  <1>       ;
47535                                  <1>       ; TRDOS 386 - IRQ CALLBACK structures (parameters):
47536                                  <1>       ;
47537                                  <1>       ;       [u.irqlock] = 1 word, IRQ flags (0-15) that indicates
47538                                  <1>       ;                 which IRQs are locked by (that) user.
47539                                  <1>       ;                   Lock and unlock (by user) will change
47540                                  <1>       ;                 these flags or 'terminate process' (sysexit)
47541                                  <1>       ;                 will clear these flags and unlock those IRQs.
47542                                  <1>       ;
47543                                  <1>       ;                 Bit 0 is for IRQ 0 and Bit 15 is for IRQ 15
47544                                  <1>       ;
47545                                  <1>       ;       IRQ(x).owner      : 1 byte, user, [u.uno], 0 = free (unlocked)
47546                                  <1>       ;
47547                                  <1>       ;       IRQ(x).method : 1 byte for callback method & status
47548                                  <1>       ;                   0 = Signal Response Byte method
47549                                  <1>       ;                   1 = Callback service method
47550                                  <1>       ;                   >1 = invalid for current 'syscalback'.
47551                                  <1>       ;                 or(+) 80h = IRQ is in use by system (ring 0)
47552                                  <1>       ;                             function (audio etc.) or
47553                                  <1>       ;                           a device driver.
47554                                  <1>       ;                 (system function will ignore the lock/owner)
47555                                  <1>       ;
47556                                  <1>       ;       IRQ(x).srb: 1 byte, Signal Return/Response byte value
47557                                  <1>       ;                   (a fixed value by user or a counter value
47558                                  <1>       ;                  from 0 to 255, which is increased by every
47559                                  <1>       ;                  interrupt just before putting it into
47560                                  <1>       ;                  the Signal Response byte address
47561                                  <1>       ;                  (This is not used in callback serv method)
47562                                  <1>       ;
47563                                  <1>       ;       IRQ(x).addr       : 1 dword
47564                                  <1>       ;                   Signal Response Byte address (physical)
47565                                  <1>       ;                               -or-
47566                                  <1>       ;                   Callback service address (virtual)
47567                                  <1>       ;
47568                                  <1>       ;       IRQ(x).dev: 1 byte
47569                                  <1>       ;                   0 = Default device or kernel function
47570                                  <1>       ;                               -or-
47571                                  <1>       ;                   1-255 = Assigned device driver number
47572                                  <1>       ;
47573                                  <1>       ;       (x) = 3,4,5,7,9,10,11,12,13
47574                                  <1>       ;
47575                                  <1>
47576 000101D1 80FB0F                 <1>       cmp    bl, 15
47577 000101D4 7729                   <1>       ja     short scbs_2
47578                                  <1>
47579 000101D6 80FF03                 <1>       cmp    bh, 3
47580 000101D9 7724                   <1>       ja     short scbs_2  ; invalid parameter
47581                                  <1>
47582 000101DB 0FB6FB                 <1>       movzx  edi, bl ; save IRQ number
47583                                  <1>
47584                                  <1>              ;  IRQ 0,1,2,6,8,14,15 are prohibited
47585                                  <1>       ;IRQenum: ; 'trdosk9.s'
47586                                  <1>       ;     db  0,0,0,1,2,3,0,4,0,5,6,7,8,9,0,0
47587                                  <1>
47588 000101DE 0FB6B7[CC0B0100]       <1>       movzx  esi, byte [edi+IRQenum] ; IRQ availability
47589                                  <1>                                      ; enumeration/index
47590                                  <1>       ;dec   esi
47591 000101E5 664E                   <1>       dec    si
47592 000101E7 780F                   <1>       js     short scbs_1 ;  0 -> 0FFFFh
47593                                  <1>
47594                                  <1>       ; ESI = IRQ callback parameters index number (0 to 8)
47595                                  <1>
47596 000101E9 08FF                   <1>       or     bh, bh
47597 000101EB 7419                   <1>       jz     short scbs_4 ; unlink the IRQ (in BL)
47598                                  <1>
47599 000101ED FECF                   <1>       dec    bh
47600                                  <1>       ; bh = method (0 = signal response byte, 1 = callback)
47601                                  <1>       ;            (2 = auto increment of signal response byte)
47602                                  <1>
47603 000101EF 80BE[FE600100]00       <1>       cmp    byte [esi+IRQ.owner], 0 ; locked ?
47604 000101F6 7637                   <1>       jna    short scbs_6 ; no... OK...
47605                                  <1>
47606                                  <1> scbs_1:
47607                                  <1>       ; permission denied (prohibited IRQ)
47608 000101F8 B80B000000             <1>       mov    eax, ERR_PERM_DENIED
47609 000101FD F9                     <1>       stc
47610 000101FE C3                     <1>       retn
47611                                  <1> scbs_2:
47612 000101FF F9                     <1>       stc
47613                                  <1> scbs_3:
47614 00010200 B817000000             <1>       mov    eax, ERR_INV_PARAMETER
47615 00010205 C3                     <1>       retn
47616                                  <1>
47617                                  <1> scbs_4: ; unlink the requested IRQ (if it belongs to current user)
47618                                  <1>       ; 10/06/2017
47619                                  <1>       ; 22/04/2017
47620                                  <1>       ; 14/04/2017
47621                                  <1>       ; If AL = 0 -> The caller is 'syscalbac'
47622 00010206 8AA6[FE600100]         <1>       mov    ah, [esi+IRQ.owner]
47623 0001020C 3A25[B3030600]         <1>       cmp    ah, [u.uno]
47624 00010212 75E4                   <1>       jne    short scbs_1
47625                                  <1>
```

```
47626 00010214 FE0D[D6030600]      <1>       dec    byte [u.irqc] ; decrease IRQ count (in use)
47627                              <1>
47628                              <1>       ;sub   ah, ah
47629                              <1>       ;mov   [esi+IRQ.owner], ah ; 0 ; free !!!
47630                              <1>       ;and   byte [esi+IRQ.method], 80h
47631                              <1>       ;mov   [esi+IRQ.srb], ah ; 0
47632                              <1>       ;mov   [esi+IRQ.dev], ah ; 0
47633                              <1>       ;mov   dword [esi+IRQ.addr], 0
47634                              <1>       ;mov   dword [u.r0], 0
47635                              <1>
47636                              <1>       ;mov   byte [esi+IRQ.owner], 0
47637                              <1>
47638                              <1>       ; 22/04/2017
47639 0001021A 29C0               <1>       sub    eax, eax
47640 0001021C 8886[FE600100]     <1>       mov    [esi+IRQ.owner], al ; 0
47641                              <1>       ; 10/06/2017
47642 00010222 8686[10610100]     <1>       xchg   al, [esi+IRQ.method]
47643 00010228 2480               <1>       and    al, 80h
47644 0001022A 745E               <1>       jz     short scbs_12
47645                              <1>       ; Audio device must be disabled -later- ! ([IRQ.medhod] = 80h)
47646                              <1>
47647                              <1> ;     cmp    byte [esi+IRQ.method], 80h ; device drv or kernel extension ?
47648                              <1> ;     jb     short scbs_12 ; bh = 0 reset to default IRQ handler
47649                              <1> ;
47650                              <1> ;     and    al, al
47651                              <1> ;     jz     short  scbs_5  ; the caller is 'syscalbac'
47652                              <1> ;     ; The caller is 'sysaudio' or ...
47653 0001022C 30C0               <1>       xor    al, al
47654                              <1> ;     mov    [esi+IRQ.method], al ; 0 ; reset kernel extension flag
47655                              <1> ;scbs_5:
47656                              <1> ;     sub    ah, ah
47657                              <1>       ;mov   [u.r0], eax ; 0
47658 0001022E C3                 <1>       retn
47659                              <1>
47660                              <1> scbs_6:
47661                              <1>       ; 14/04/2017
47662 0001022F 20C0               <1>       and    al, al
47663 00010231 7405               <1>       jz     short scbs_7  ; the caller is 'syscalbac'
47664                              <1>       ; AL = user number ([u.uno] or [audio.user] or ...)
47665                              <1>       ; The caller is 'sysaudio' or ...
47666                              <1>       ;
47667                              <1>       ; bh = method (0 = signal response byte, 1 = callback)
47668                              <1>       ;           (2 = auto increment of signal response byte)
47669                              <1>
47670 00010233 80CF80             <1>       or     bh, 80h             ; Kernel extension flag !
47671 00010236 EB0A               <1>       jmp    short scbs_8
47672                              <1> scbs_7:
47673 00010238 8A86[10610100]     <1>       mov    al, [esi+IRQ.method] ; >= 80h = kernel is using this IRQ
47674 0001023E 2480               <1>       and    al, 80h ; use only bit 7 (kernel function flag)
47675 00010240 08C7               <1>       or     bh, al         ; method
47676                              <1>                             ; 0 = signal response byte, 1 = callback
47677                              <1>                             ; 2 = auto increment of s.r.b.
47678                              <1> scbs_8:
47679 00010242 A0[B3030600]       <1>       mov    al, [u.uno] ; user (process) number (1 to 16)
47680 00010247 8886[FE600100]     <1>       mov    [esi+IRQ.owner], al ; lock the IRQ for user
47681 0001024D 88BE[10610100]     <1>       mov    [esi+IRQ.method], bh
47682                              <1>
47683                              <1> ;     test   bh, 1
47684                              <1> ;     jnz    short scbs_9  ; Callback method, CX will not be used
47685                              <1> ;
47686                              <1> ;     test   bh, 2         ; use auto increment (counter) method
47687                              <1> ;     jz     short scbs_10 ; (count can be used for buffer switch)
47688                              <1> ;scbs_9:
47689                              <1> ;     xor    ecx, ecx ; 0
47690                              <1> scbs_10:
47691                              <1>       ;mov   [esi+IRQ.method], bh
47692 00010253 888E[19610100]     <1>       mov    [esi+IRQ.srb], cl
47693 00010259 C686[07610100]00   <1>       mov    byte [esi+IRQ.dev], 0 ; device number is always 0
47694                              <1>                             ; for this system call
47695                              <1>       ;test  bh, 1
47696 00010260 80E701             <1>       and    bh, 1 ; 17/04/2017
47697 00010263 7513               <1>       jnz    short scbs_11 ; callback method, use virtual address
47698                              <1>
47699 00010265 53                 <1>       push   ebx ; IRQ number (in BL)
47700 00010266 89D3               <1>       mov    ebx, edx
47701                              <1>       ; ebx = virtual address
47702                              <1>       ; [u.pgdir] = page directory's physical address
47703 00010268 FE05[9E600100]     <1>       inc     byte [no_page_swap] ; 1
47704                              <1>                   ; Do not add this page to swap queue
47705                              <1>                   ; and remove it from swap queue if it is
47706                              <1>                   ; on the queue.
47707 0001026E E81B50FFFF         <1>       call   get_physical_addr
47708 00010273 5B                 <1>       pop    ebx
47709 00010274 728A               <1>       jc     scbs_3 ; invalid address !
47710                              <1>       ; eax = physical address of the virtual address in user's space
47711 00010276 89C2               <1>       mov    edx, eax
47712                              <1> scbs_11:
47713 00010278 66C1E602           <1>       shl    si, 2        ; byte (index) to dword (offset)
47714 0001027C 8996[22610100]     <1>       mov    [esi+IRQ.addr], edx
47715                              <1>
47716 00010282 FE05[D6030600]     <1>       inc    byte [u.irqc]; increase IRQ (in use) count
47717                              <1>
47718 00010288 FEC7               <1>       inc     bh ; 17/04/2017
47719                              <1>       ; bh > 0 -> set to requested IRQ handler (IRQ_u_list)
47720                              <1> scbs_12:
47721 0001028A 88D8               <1>       mov    al, bl ; IRQ number
47722 0001028C 88FC               <1>       mov    ah, bh ; 0 = reset, >0 = set
47723 0001028E E859F4FFFF         <1>       call   set_hardware_int_vector
47724                              <1>
47725 00010293 31C0               <1>       xor    eax, eax
47726                              <1>       ;mov   [u.r0], eax ; 0
47727                              <1>
```

```
47728 00010295 C3                <1>       retn   ; return with success (cf=0, eax=0)
47729                            <1>
47730                            <1> otty:
47731                            <1> sret:
47732                            <1> ocvt:
47733                            <1> ctty:
47734                            <1> cret:
47735                            <1> ccvt:
47736                            <1> rtty:
47737                            <1> wtty:
47738                            <1> rmem:
47739                            <1> wmem:
47740                            <1> rfd:
47741                            <1> rhd:
47742                            <1> wfd:
47743                            <1> whd:
47744                            <1> rlpt:
47745                            <1> wlpt:
47746                            <1> rcvt:
47747                            <1> xmtt:
47748 00010296 C3                <1>       retn
47749                                %include 'trdosk9.s' ; 04/01/2016
47750                            <1> ; ****************************************************************************
47751                            <1> ; TRDOS386.ASM (TRDOS 386 Kernel - v2.0.0) - INITIALIZED DATA : trdosk9.s
47752                            <1> ; ----------------------------------------------------------------------------
47753                            <1> ; Last Update: 22/07/2017
47754                            <1> ; ----------------------------------------------------------------------------
47755                            <1> ; Beginning: 04/01/2016
47756                            <1> ; ----------------------------------------------------------------------------
47757                            <1> ; ----------------------------------------------------------------------------
47758                            <1> ; Assembler: NASM version 2.11 (trdos386.s)
47759                            <1> ; ----------------------------------------------------------------------------
47760                            <1> ; Derived from TRDOS Operating System v1.0 (8086) source code by Erdogan Tan
47761                            <1> ; TRDOS2.ASM (09/11/2011)
47762                            <1> ; ****************************************************************************
47763                            <1> ; DRV_INIT.ASM [26/09/2009] Last Update: 07/08/2011
47764                            <1> ; MAINPROG.ASM [17/01/2004] Last Update: 09/11/2011
47765                            <1> ; CMD_INTR.ASM [29/01/2005] Last Update: 09/11/2011
47766                            <1> ; FILE.ASM [29/10/2009] Last Update: 09/10/2011
47767                            <1>
47768                            <1> ; 12/02/2016
47769                            <1> Last_DOS_DiskNo:
47770 00010297 01                <1>             db 1 ; A: = 0 & B: = 1
47771                            <1>
47772                            <1> Restore_CDIR:
47773 00010298 FF                <1>             db 0FFh ; Initial value -> any number except 0
47774                            <1>
47775                            <1> msg_CRLF_temp:
47776 00010299 070D0A00          <1>             db 07h, 0Dh, 0Ah, 0
47777                            <1>
47778                            <1> Magic_Bytes:
47779 0001029D 04                <1>             db 4
47780 0001029E 01                <1>             db 1
47781                            <1> mainprog_Version:
47782 0001029F 07                <1>             db 7
47783 000102A0 5B5452444F535D204D- <1>             db "[TRDOS] Main Program v2.0.220717"
47784 000102A9 61696E2050726F6772- <1>
47785 000102B2 616D2076322E302E32- <1>
47786 000102BB 3230373137          <1>
47787 000102C0 0D0A              <1>             db 0Dh, 0Ah
47788 000102C2 286329204572646F67- <1>             db "(c) Erdogan Tan 2005-2017"
47789 000102CB 616E2054616E203230- <1>
47790 000102D4 30352D32303137      <1>
47791 000102DB 0D0A00            <1>             db 0Dh, 0Ah, 0
47792                            <1>
47793                            <1> MainProgCfgFile: ; 14/04/2016
47794 000102DE 4D41494E50524F472E- <1>             db "MAINPROG.CFG", 0
47795 000102E7 43464700          <1>
47796                            <1>
47797                            <1> TRDOSPromptLabel:
47798 000102EB 5452444F53        <1>             db "TRDOS"
47799 000102F0 00                <1>             db 0
47800 000102F1 00<rept>          <1>                times 5 db 0
47801 000102F6 00                <1>             db 0
47802                            <1>
47803                            <1> ; INTERNAL COMMANDS
47804                            <1> Command_List:
47805 000102F7 44495200          <1> Cmd_Dir:    db "DIR", 0
47806 000102FB 434400            <1> Cmd_Cd:        db "CD", 0
47807 000102FE 433A00            <1> Cmd_Drive:  db "C:", 0
47808 00010301 56455200          <1> Cmd_Ver:    db "VER", 0
47809 00010305 4558495400        <1> Cmd_Exit:   db "EXIT", 0
47810 0001030A 50524F4D505400    <1> Cmd_Prompt: db "PROMPT", 0
47811 00010311 564F4C554D4500    <1> Cmd_Volume: db "VOLUME", 0
47812 00010318 4C4F4E474E414D4500 <1> Cmd_LongName:     db "LONGNAME", 0
47813 00010321 4441544500        <1> Cmd_Date:   db "DATE", 0
47814 00010326 54494D4500        <1> Cmd_Time:   db "TIME", 0
47815 0001032B 52554E00          <1> Cmd_Run:    db "RUN", 0
47816 0001032F 53455400          <1> Cmd_Set:    db "SET", 0
47817 00010333 434C5300          <1> Cmd_Cls:    db "CLS", 0
47818 00010337 53484F5700        <1> Cmd_Show:   db "SHOW", 0
47819 0001033C 44454C00          <1> Cmd_Del:    db "DEL", 0
47820 00010340 41545452494200    <1> Cmd_Attrib: db "ATTRIB", 0
47821 00010347 52454E414D4500    <1> Cmd_Rename: db "RENAME", 0
47822 0001034E 524D44495200      <1> Cmd_Rmdir:  db "RMDIR", 0
47823 00010354 4D4B44495200      <1> Cmd_Mkdir:  db "MKDIR", 0
47824 0001035A 434F505900        <1> Cmd_Copy:   db "COPY", 0
47825 0001035F 4D4F564500        <1> Cmd_Move:   db "MOVE", 0
47826 00010364 5041544800        <1> Cmd_Path:   db "PATH", 0
47827 00010369 4D454D00          <1> Cmd_Mem:    db "MEM", 0
47828 0001036D 00                <1>             db 0
47829 0001036E 46494E4400        <1> Cmd_Find:   db "FIND", 0
```

```
47830 00010373 4543484F00          <1> Cmd_Echo:    db "ECHO", 0
47831 00010378 2A00                <1> Cmd_Remark:  db "*", 0
47832 0001037A 3F00                <1> Cmd_Help:    db "?", 0
47833 0001037C 44455649434500      <1> Cmd_Device:  db "DEVICE", 0
47834 00010383 4445564C49535400    <1> Cmd_DevList: db "DEVLIST", 0
47835 0001038B 434844495200        <1> Cmd_Chdir:   db "CHDIR", 0
47836 00010391 4245455000          <1> Cmd_Beep:    db "BEEP", 0
47837                              <1>
47838 00010396 00                  <1>              db 0
47839                              <1>
47840                              <1> ; 15/02/2016 (FILE.ASM, 09/10/2011)
47841                              <1> invalid_fname_chars:
47842 00010397 222728292A2B2C2F    <1>              db 22h, 27h, 28h, 29h, 2Ah, 2Bh, 2Ch, 2Fh
47843 0001039F 3A3B3C3D3E3F40      <1>              db 3Ah, 3Bh, 3Ch, 3Dh, 3Eh, 3Fh, 40h
47844 000103A6 5B5C5D5E60          <1>              db 5Bh, 5Ch, 5Dh, 5Eh, 60h
47845                              <1> sizeInvFnChars  equ ($ - invalid_fname_chars)
47846                              <1> ;
47847                              <1>
47848                              <1> Msg_Enter_Date:
47849 000103AB 456E746572206E6577- <1>                  db 'Enter new date (dd-mm-yy): '
47850 000103B4 206461746520286464- <1>
47851 000103BD 2D6D6D2D7979293A20  <1>
47852 000103C6 00                  <1>                  db 0
47853                              <1> Msg_Show_Date:
47854 000103C7 43757272656E742064- <1>                  db   'Current date is '
47855 000103D0 61746520697320      <1>
47856 000103D7 30                  <1> Day:             db   '0'
47857 000103D8 30                  <1>                  db   '0'
47858 000103D9 2F                  <1>                  db   '/'
47859 000103DA 30                  <1> Month:           db   '0'
47860 000103DB 30                  <1>                  db   '0'
47861 000103DC 2F                  <1>                  db   '/'
47862 000103DD 30                  <1> Century:         db   '0'
47863 000103DE 30                  <1>                  db   '0'
47864 000103DF 30                  <1> Year:            db   '0'
47865 000103E0 30                  <1>                  db   '0'
47866 000103E1 0D0A00              <1>                  db   0Dh, 0Ah, 0
47867                              <1>
47868                              <1> Msg_Enter_Time:
47869 000103E4 456E746572206E6577- <1>                  db 'Enter new time: '
47870 000103ED 2074696D653A20      <1>
47871 000103F4 00                  <1>                  db 0
47872                              <1> Msg_Show_Time:
47873 000103F5 43757272656E742074- <1>                  db   'Current time is '
47874 000103FE 696D6520697320      <1>
47875 00010405 30                  <1> Hour:            db   '0'
47876 00010406 30                  <1>                  db   '0'
47877 00010407 3A                  <1>                  db   ':'
47878 00010408 30                  <1> Minute:          db   '0'
47879 00010409 30                  <1>                  db   '0'
47880 0001040A 3A                  <1>                  db   ':'
47881 0001040B 30                  <1> Second:          db   '0'
47882 0001040C 30                  <1>                  db   '0'
47883 0001040D 0D0A00              <1>                  db   0Dh, 0Ah, 0
47884                              <1>
47885                              <1> ;VolSize_Unit1:   dd 0
47886                              <1> ;VolSize_Unit2:   dd 0
47887                              <1>
47888                              <1> VolSize_KiloBytes:
47889 00010410 206B696C6F62797465- <1>                  db " kilobytes", 0Dh, 0Ah, 0
47890 00010419 730D0A00            <1>
47891                              <1> VolSize_Bytes:
47892 0001041D 2062797465730D0A00  <1>                  db " bytes", 0Dh, 0Ah, 0
47893                              <1> Volume_in_drive:
47894 00010426 0D0A                <1>                  db 0Dh, 0Ah
47895                              <1> Vol_FS_Name:
47896 00010428 54522046533120      <1>                  db "TR FS1 "
47897 0001042F 566F6C756D6520696E- <1>                  db "Volume in drive "
47898 00010438 20647269766520      <1>
47899 0001043F 30                  <1> Vol_Drv_Name:    db 30h
47900 00010440 3A                  <1>                  db ":"
47901 00010441 20697320            <1>                  db " is "
47902 00010445 0D0A00              <1>                  db 0Dh, 0Ah, 0
47903                              <1> Dir_Drive_Str:
47904 00010448 54522D444F53204472- <1>                   db "TR-DOS Drive "
47905 00010451 69766520            <1>
47906                              <1> Dir_Drive_Name:
47907 00010455 303A                <1>                   db "0:"
47908 00010457 0D0A                <1>                   db  0Dh, 0Ah
47909                              <1> Vol_Str_Header:
47910 00010459 566F6C756D65204E61- <1>                   db "Volume Name: "
47911 00010462 6D653A20            <1>
47912                              <1> Vol_Name:
47913 00010466 00<rept>            <1>                  times 64 db 0
47914 000104A6 00                  <1>                  db 0
47915                              <1> Vol_Serial_Header:
47916 000104A7 0D0A                <1>                  db 0Dh, 0Ah
47917 000104A9 566F6C756D65205365- <1>                  db "Volume Serial No: "
47918 000104B2 7269616C204E6F3A20  <1>
47919                              <1> Vol_Serial1:
47920 000104BB 30303030            <1>                  db "0000"
47921 000104BF 2D                  <1>                  db "-"
47922                              <1> Vol_Serial2:
47923 000104C0 30303030            <1>                  db "0000"
47924 000104C4 0D0A00              <1>                  db 0Dh, 0Ah, 0
47925                              <1>
47926                              <1> ;Vol_Tot_Sec_Str_Start:
47927                              <1> ;                dd 0
47928                              <1> Vol_Total_Sector_Header:
47929 000104C7 0D0A                <1>                  db 0Dh, 0Ah
47930 000104C9 566F6C756D65205369- <1>                  db "Volume Size : ", 0
47931 000104D2 7A65203A2000        <1>
```

```
47932                                   <1> ;Vol_Tot_Sec_Str:
47933                                   <1> ;           db "0000000000"
47934                                   <1> ;Vol_Tot_Sec_Str_End:
47935                                   <1> ;           db 0
47936                                   <1> ;Vol_Free_Sectors_Str_Start:
47937                                   <1> ;           dd 0
47938                                   <1> Vol_Free_Sectors_Header:
47939 000104D8 467265652053706163-      <1>             db "Free Space  : ", 0
47940 000104E1 6520203A2000             <1>
47941                                   <1> ;Vol_Free_Sectors_Str:
47942                                   <1> ;           db "0000000000"
47943                                   <1> ;Vol_Free_Sectors_Str_End:
47944                                   <1> ;           db 0
47945                                   <1>
47946                                   <1> Dir_Str_Header:
47947 000104E7 4469726563746F7279-      <1>                 db "Directory: "
47948 000104F0 3A20                     <1>
47949 000104F2 2F                       <1> Dir_Str_Root:   db "/"
47950 000104F3 00<rept>                 <1> Dir_Str:        times 64 db 0
47951 00010533 00000000                 <1>                 dd 0
47952 00010537 00                       <1>                 db 0
47953                                   <1>
47954                                   <1> Msg_Bad_Command:
47955 00010538 42616420636F6D6D61-      <1>                 db "Bad command or file name!"
47956 00010541 6E64206F722066696C-      <1>
47957 0001054A 65206E616D6521           <1>
47958 00010551 0D0A00                   <1>                 db 0Dh, 0Ah, 0
47959                                   <1>
47960                                   <1> msgl_drv_not_ready:
47961 00010554 070D0A                   <1>             db 07h, 0Dh, 0Ah
47962                                   <1>
47963                                   <1> ; CMD_INTR.ASM - 09/11/2011 - Messages
47964                                   <1>
47965                                   <1> Msg_Not_Ready_Read_Err:
47966 00010557 4472697665206E6F74-      <1>                 db "Drive not ready or read error!"
47967 00010560 207265616479206F72-      <1>
47968 00010569 207265616420657272-      <1>
47969 00010572 6F7221                   <1>
47970 00010575 0D0A00                   <1>                 db 0Dh, 0Ah, 0
47971                                   <1>
47972                                   <1> Msg_Not_Ready_Write_Err:
47973 00010578 4472697665206E6F74-      <1>                 db "Drive not ready or write error!"
47974 00010581 207265616479206F72-      <1>
47975 0001058A 207772697465206572-      <1>
47976 00010593 726F7221                 <1>
47977 00010597 0D0A00                   <1>                 db 0Dh, 0Ah, 0
47978                                   <1>
47979                                   <1> Msg_Dir_Not_Found:
47980 0001059A 4469726563746F7279-      <1>                 db "Directory not found!"
47981 000105A3 206E6F7420666F756E-      <1>
47982 000105AC 6421                     <1>
47983 000105AE 0D0A00                   <1>                 db 0Dh, 0Ah, 0
47984                                   <1>
47985                                   <1> Msg_File_Not_Found:
47986 000105B1 46696C65206E6F7420-      <1>                 db "File not found!"
47987 000105BA 666F756E6421             <1>
47988 000105C0 0D0A00                   <1>                 db 0Dh, 0Ah, 0
47989                                   <1>
47990                                   <1> Msg_File_Directory_Not_Found:
47991 000105C3 46696C65206F722064-      <1>                 db "File or directory not found!"
47992 000105CC 69726563746F727920-      <1>
47993 000105D5 6E6F7420666F756E64-      <1>
47994 000105DE 21                       <1>
47995 000105DF 0D0A00                   <1>                 db 0Dh, 0Ah, 0
47996                                   <1>
47997                                   <1> Msg_LongName_Not_Found:
47998 000105E2 4C6F6E67206E616D65-      <1>                 db "Long name not found!"
47999 000105EB 206E6F7420666F756E-      <1>
48000 000105F4 6421                     <1>
48001 000105F6 0D0A00                   <1>                 db 0Dh, 0Ah, 0
48002                                   <1>
48003                                   <1> beep_Insufficient_Memory: ; 20/02/2017
48004 000105F9 0D0A                     <1>             db 0Dh, 0Ah
48005 000105FB 07                       <1>             db 07h
48006                                   <1> Msg_Insufficient_Memory:
48007 000105FC 496E73756666696369-      <1>                 db "Insufficient memory!"
48008 00010605 656E74206D656D6F72-      <1>
48009 0001060E 7921                     <1>
48010 00010610 0D0A00                   <1>                 db 0Dh, 0Ah, 0
48011                                   <1>
48012                                   <1> Msg_Error_Code:
48013 00010613 436F6D6D616E642066-      <1>                 db 'Command failed! Error code : '
48014 0001061C 61696C656421204572-      <1>
48015 00010625 726F7220636F646520-      <1>
48016 0001062E 3A20                     <1>
48017 00010630 303068                   <1> error_code_hex: db '00h'
48018 00010633 0A0A00                   <1>                 db 0Ah, 0Ah, 0
48019                                   <1>
48020                                   <1> align 2
48021                                   <1>
48022                                   <1> ; 10/02/2016
48023                                   <1> ; DIR.ASM - 09/10/2011
48024                                   <1>
48025 00010636 3C4449523E20202020-      <1> Type_Dir:       db '<DIR>     ' ; 10 bytes
48026 0001063F 20                       <1>
48027                                   <1>
48028                                   <1> File_Name:
48029 00010640 20<rept>                 <1>                 times 12 db 20h
48030 0001064C 20                       <1>             db 20h
48031                                   <1> Dir_Or_FileSize:
48032 0001064D 20<rept>                 <1>                 times 10 db 20h
48033 00010657 20                       <1>             db 20h
```

```
48034                                  <1> File_Attribute:
48035 00010658 20202020                <1>             dd 20202020h
48036 0001065C 20                      <1>             db 20h
48037                                  <1> File_Day:
48038 0001065D 3030                    <1>                 db '0','0'
48039 0001065F 2F                      <1>             db '/'
48040                                  <1> File_Month:
48041 00010660 3030                    <1>                 db '0','0'
48042 00010662 2F                      <1>             db '/'
48043                                  <1> File_Year:
48044 00010663 30303030                <1>                 db '0','0','0','0'
48045 00010667 20                      <1>             db 20h
48046                                  <1> File_Hour:
48047 00010668 3030                    <1>                 db '0','0'
48048 0001066A 3A                      <1>             db ':'
48049                                  <1> File_Minute:
48050 0001066B 3030                    <1>                 db '0','0'
48051 0001066D 00                      <1>             db 0
48052                                  <1>
48053                                  <1> Decimal_File_Count_Header:
48054 0001066E 0D0A                    <1>             db 0Dh, 0Ah
48055                                  <1> Decimal_File_Count:
48056 00010670 00<rept>                <1>             times 6 db 0
48057                                  <1>
48058 00010676 2066696C6528732920-     <1> str_files:   db " file(s) & "
48059 0001067F 2620                    <1>
48060                                  <1> Decimal_Dir_Count:
48061 00010681 00<rept>                <1>             times 6 db 0
48062                                  <1> str_dirs:
48063 00010687 206469726563746F72-     <1>             db " directory(s) "
48064 00010690 7928732920              <1>
48065 00010695 0D0A00                  <1>             db 0Dh, 0Ah, 0
48066                                  <1>
48067 00010698 206279746528732920-     <1> str_bytes:   db " byte(s) in file(s)"
48068 000106A1 696E2066696C652873-     <1>
48069 000106AA 29                      <1>
48070 000106AB 0D0A00                  <1>             db 0Dh, 0Ah, 0
48071                                  <1>
48072                                  <1> ; CMD_INTR.ASM - 09/11/2011
48073                                  <1> ; 07/10/2010
48074                                  <1> Msg_invalid_name_chars:
48075 000106AE 496E76616C69642066-     <1>                 db "Invalid file or directory name characters!"
48076 000106B7 696C65206F72206469-     <1>
48077 000106C0 726563746F7279206E-     <1>
48078 000106C9 616D65206368617261-     <1>
48079 000106D2 637465727321            <1>
48080 000106D8 0D0A00                  <1>             db 0Dh, 0Ah, 0
48081                                  <1> ; 21/02/2016
48082 000106DB 46696C65206F722064-     <1> Msg_Name_Exists: db "File or directory name exists!"
48083 000106E4 69726563746F727920-     <1>
48084 000106ED 6E616D652065786973-     <1>
48085 000106F6 747321                  <1>
48086 000106F9 0D0A00                  <1>                 db 0Dh, 0Ah, 0
48087                                  <1> Msg_DoYouWantMkdir:
48088 000106FC 446F20796F75207761-     <1>                 db "Do you want to make directory ", 0
48089 00010705 6E7420746F206D616B-     <1>
48090 0001070E 65206469726563746F-     <1>
48091 00010717 72792000                <1>
48092 0001071B 2028592F4E29203F20-     <1> Msg_YesNo:      db " (Y/N) ? ", 0
48093 00010724 00                      <1>
48094 00010725 000D0A00                <1> Y_N_nextline:      db 0, 0Dh, 0Ah, 0
48095 00010729 4F4B2E0D0A00            <1> Msg_OK:            db "OK.", 0Dh, 0Ah, 0
48096                                  <1>
48097                                  <1> ; 27/02/2016
48098                                  <1> Msg_DoYouWantRmDir:
48099 0001072F 446F20796F75207761-     <1>                 db "Do you want to delete directory ", 0
48100 00010738 6E7420746F2064656C-     <1>
48101 00010741 657465206469726563-     <1>
48102 0001074A 746F72792000            <1>
48103                                  <1> Msg_Dir_Not_Empty:
48104 00010750 4469726563746F7279-     <1>                 db "Directory not empty!"
48105 00010759 206E6F7420656D7074-     <1>
48106 00010762 7921                    <1>
48107 00010764 0D0A00                  <1>                 db 0Dh, 0Ah, 0
48108                                  <1>
48109                                  <1> Msg_DoYouWantDelete:
48110 00010767 446F20796F75207761-     <1>                 db "Do you want to delete file ",0
48111 00010770 6E7420746F2064656C-     <1>
48112 00010779 6574652066696C6520-     <1>
48113 00010782 00                      <1>
48114                                  <1>
48115 00010783 44656C657465642E2E-     <1> Msg_Deleted:    db "Deleted...", 0Dh, 0Ah, 0
48116 0001078C 2E0D0A00                <1>
48117                                  <1>
48118                                  <1> Msg_Permission_Denied:
48119 00010790 07                      <1>                 db 7
48120 00010791 5065726D697373696F-     <1>                 db "Permission denied!", 0Dh, 0Ah, 0
48121 0001079A 6E2064656E69656421-     <1>
48122 000107A3 0D0A00                  <1>
48123                                  <1>
48124                                  <1> ; 04/03/2016
48125 000107A6 4E657720                <1> Msg_New:        db "New "
48126 000107AA 00                      <1>                 db 0
48127                                  <1> Str_Attributes:
48128 000107AB 417474726962757465-     <1>                 db "Attributes : "
48129 000107B4 73203A20                <1>
48130 000107B8 4E4F524D414C            <1> Attr_Chars:     db "NORMAL"
48131 000107BE 00                      <1>                 db 0
48132                                  <1>
48133                                  <1> ; 06/03/2016
48134                                  <1> ; CMD_INTR.ASM - 16/11/2010
48135                                  <1> Msg_DoYouWantRename:
```

```
48136 000107BF 446F20796F75207761- <1>                db "Do you want to rename ", 0
48137 000107C8 6E7420746F2072656E- <1>
48138 000107D1 616D652000          <1>
48139 000107D6 66696C652000        <1> Rename_File:    db "file ", 0
48140 000107DC 6469726563746F7279- <1> Rename_Directory: db "directory ", 0
48141 000107E5 2000                <1>
48142 000107E7 00<rept>            <1> Rename_OldName: times 13 db 0
48143 000107F4 20617320            <1> Msg_File_rename_as: db " as "
48144 000107F8 00<rept>            <1> Rename_NewName: times 13 db 0
48145                              <1>
48146                              <1> ; 08/03/2016
48147                              <1> ; CMD_INTR.ASM - 01/08/2010 - 23/04/2011
48148                              <1> msg_not_same_drv:
48149 00010805 4E6F742073616D6520- <1>                db "Not same drive!"
48150 0001080E 647269766521        <1>
48151 00010814 0D0A00              <1>                db 0Dh, 0Ah, 0
48152                              <1>
48153                              <1> Msg_DoYouWantMoveFile:
48154 00010817 446F20796F75207761- <1>                db "Do you want to move file", 0
48155 00010820 6E7420746F206D6F76- <1>
48156 00010829 652066696C6500      <1>
48157                              <1>
48158                              <1> msg_insufficient_disk_space:
48159 00010830 496E73756666696369- <1>                db "Insufficient disk space!"
48160 00010839 656E74206469736B20- <1>
48161 00010842 737061636521        <1>
48162 00010848 0D0A00              <1>                db 0Dh, 0Ah, 0
48163                              <1>
48164                              <1> ; 01/08/2010
48165                              <1> msg_source_file:
48166 0001084B 0D0A536F7572636520- <1>                db 0Dh, 0Ah, "Source file name      :   "
48167 00010854 66696C65206E616D65- <1>
48168 0001085D 2020202020203A2020- <1>
48169 00010866 20                  <1>
48170                              <1> msg_source_file_drv:
48171 00010867 203A00              <1>                db " :", 0
48172                              <1> msg_destination_file:
48173 0001086A 0D0A44657374696E61- <1>                db 0Dh, 0Ah, "Destination file name :   "
48174 00010873 74696F6E2066696C65- <1>
48175 0001087C 206E616D65203A2020- <1>
48176 00010885 20                  <1>
48177                              <1> msg_destination_file_drv:
48178 00010886 203A00              <1>                db " :", 0
48179                              <1> msg_copy_nextline:
48180 00010889 0D0A00              <1>                db 0Dh, 0Ah, 0
48181                              <1>
48182                              <1> ; 15/03/2016
48183                              <1> ; CMD_INTR.ASM
48184                              <1>
48185                              <1> Msg_DoYouWantOverWriteFile:
48186 0001088C 446F20796F75207761- <1>                    db "Do you want to overwrite file ",0
48187 00010895 6E7420746F206F7665- <1>
48188 0001089E 727772697465206669- <1>
48189 000108A7 6C652000            <1>
48190                              <1>
48191                              <1> Msg_DoYouWantCopyFile:
48192 000108AB 446F20796F75207761- <1>                    db "Do you want to copy file",0
48193 000108B4 6E7420746F20636F70- <1>
48194 000108BD 792066696C6500      <1>
48195                              <1>
48196                              <1> Msg_read_file_error_before_EOF:
48197 000108C4 46696C652072656164- <1>                db "File reading error! (before EOF)"
48198 000108CD 696E67206572726F72- <1>
48199 000108D6 2120286265666F7265- <1>
48200 000108DF 20454F4629          <1>
48201 000108E4 0A0A00              <1>                db 0Ah, 0Ah, 0
48202                              <1>
48203                              <1> ; 18/03/2016
48204                              <1> ; TRDOS 386 (v2.0) mainprog copy procedure
48205                              <1> msg_reading:
48206 000108E7 52656164696E672E2E- <1>                db "Reading... ", 0
48207 000108F0 2E2000              <1>
48208                              <1> msg_writing:
48209 000108F3 57726974696E672E2E- <1>                db "Writing... ", 0
48210 000108FC 2E2000              <1>
48211                              <1> percentagestr:
48212 000108FF 2020202500          <1>                db "   %", 0  ; "  0%" .. "100%"
48213                              <1> ; 11/04/2016
48214                              <1> Msg_No_Set_Space:
48215 00010904 496E73756666696369- <1>                    db "Insufficient environment space!"
48216 0001090D 656E7420656E766972- <1>
48217 00010916 6F6E6D656E74207370- <1>
48218 0001091F 61636521            <1>
48219 00010923 0D0A00              <1>                    db 0Dh, 0Ah, 0
48220                              <1> ; 18/04/2016
48221                              <1> isc_msg:
48222 00010926 0D0A                <1>                db 0Dh, 0Ah
48223 00010928 494E56414C49442053- <1>                db "INVALID SYSTEM CALL", 0
48224 00010931 595354454D2043414C- <1>
48225 0001093A 4C00                <1>
48226                              <1> usi_msg:
48227 0001093C 0D0A                <1>                db 0Dh, 0Ah
48228 0001093E 554E444546494E4544- <1>                db "UNDEFINED SOFTWARE INTERRUPT", 0
48229 00010947 20534F465457415245- <1>
48230 00010950 20494E544552525550- <1>
48231 00010959 5400                <1>
48232                              <1> ifc_msg:
48233 0001095B 0D0A                <1>                db 0Dh, 0Ah
48234 0001095D 494E56414C49442046- <1>                db "INVALID FUNCTION CALL"
48235 00010966 554E4354494F4E2043- <1>
48236 0001096F 414C4C              <1>
48237                              <1> inv_msg_for_trdos_v2:
```

```
48238 00010972 20                  <1>             db 20h
48239 00010973 666F72205452444F53- <1>             db "for TRDOS v2!"
48240 0001097C 20763221            <1>
48241 00010980 07                  <1>             db 07h
48242 00010981 0D0A                <1>             db 0Dh, 0Ah
48243 00010983 0D0A                <1>             db 0Dh, 0Ah
48244 00010985 494E5420            <1>             db "INT "
48245 00010989 303068              <1> int_num_str: db "00h"
48246 0001098C 0D0A                <1>             db 0Dh, 0Ah
48247 0001098E 454158203A20        <1>             db "EAX : "
48248 00010994 303030303030303068- <1> eax_str:     db "00000000h", 0Dh, 0Ah
48249 0001099D 0D0A                <1>
48250 0001099F 454950203A20        <1>             db "EIP : "
48251 000109A5 303030303030303068- <1> eip_str:     db "00000000h", 0Dh, 0Ah, 0
48252 000109AE 0D0A00              <1>
48253                              <1>
48254                              <1> ; 07/10/2016
48255                              <1> ; Device names & parameters (for kernel devices)
48256                              <1>
48257 000109B1 90                  <1> align 2
48258                              <1> KDEV_NAME:
48259 000109B2 5454590000000000    <1>             db 'TTY',0,0,0,0,0 ; 1
48260 000109BA 4D454D0000000000    <1>             db 'MEM',0,0,0,0,0 ; 2
48261 000109C2 4644300000000000    <1>             db 'FD0',0,0,0,0,0 ; 3
48262 000109CA 4644310000000000    <1>             db 'FD1',0,0,0,0,0 ; 4
48263 000109D2 4844300000000000    <1>             db 'HD0',0,0,0,0,0 ; 5
48264 000109DA 4844310000000000    <1>             db 'HD1',0,0,0,0,0 ; 6
48265 000109E2 4844320000000000    <1>             db 'HD2',0,0,0,0,0 ; 7
48266 000109EA 4844330000000000    <1>             db 'HD3',0,0,0,0,0 ; 8
48267 000109F2 4C505400000000000   <1>             db 'LPT',0,0,0,0,0 ; 9
48268 000109FA 5454593000000000    <1>             db 'TTY0',0,0,0,0 ; 10
48269 00010A02 5454593100000000    <1>             db 'TTY1',0,0,0,0 ; 11
48270 00010A0A 5454593200000000    <1>             db 'TTY2',0,0,0,0 ; 12
48271 00010A12 5454593300000000    <1>             db 'TTY3',0,0,0,0 ; 13
48272 00010A1A 5454593400000000    <1>             db 'TTY4',0,0,0,0 ; 14
48273 00010A22 5454593500000000    <1>             db 'TTY5',0,0,0,0 ; 15
48274 00010A2A 5454593600000000    <1>             db 'TTY6',0,0,0,0 ; 16
48275 00010A32 5454593700000000    <1>             db 'TTY7',0,0,0,0 ; 17
48276 00010A3A 5454593800000000    <1>             db 'TTY8',0,0,0,0 ; 18
48277 00010A42 5454593900000000    <1>             db 'TTY9',0,0,0,0 ; 19
48278 00010A4A 434F4D3100000000    <1>             db 'COM1',0,0,0,0 ; 18
48279 00010A52 434F4D3200000000    <1>             db 'COM2',0,0,0,0 ; 19
48280                              <1>             ;db 'CONSOLE',0       ; 1
48281                              <1>             ;db 'PRINTER',0   ; 9
48282                              <1>             ;db 'CDROM'    ; 20
48283                              <1>             ;db 'CDROM0'   ; 20
48284                              <1>             ;db 'CDROM1'   ; 21
48285                              <1>             ;db 'DVD'      ; 22
48286                              <1>             ;db 'DVD0'     ; 22
48287                              <1>             ;db 'DVD1'     ; 23
48288                              <1>             ;db 'USB'      ; 24
48289                              <1>             ;db 'USB0'     ; 24
48290                              <1>             ;db 'USB1'     ; 25
48291                              <1>             ;db 'USB2'     ; 26
48292                              <1>             ;db 'USB3'        ; 27
48293                              <1>             ;db 'KEYBOARD'        ; 1
48294                              <1>             ;db 'MOUSE'    ; 28
48295                              <1>             ;db 'SOUND'    ; 29
48296                              <1>             ;db 'VGA',0,0,0,0 ; 30
48297                              <1>             ;db 'CGA',0,0,0,0 ; 31
48298                              <1>             ;db 'AUDIO',0,0,0 ; 29
48299                              <1>             ;db 'VIDEO',0,0,0 ; 32
48300                              <1>             ;db 'MUSIC',0,0,0 ; 33
48301                              <1>             ;db 'ETHERNET'        ; 34
48302                              <1>             ;db 'SD0',0,0,0,0,0 ; 35
48303                              <1>             ;db 'SD1',0,0,0,0,0 ; 36
48304                              <1>             ;db 'SD2',0,0,0,0,0 ; 37
48305                              <1>             ;db 'SD3',0,0,0,0,0 ; 38
48306                              <1>             ;db 'SATA0'    ; 35
48307                              <1>             ;db 'SATA1'    ; 36
48308                              <1>             ;db 'SATA2'       ; 37
48309                              <1>             ;db 'SATA3'       ; 38
48310                              <1>             ;db 'PATA0',0,0,0  ; 5
48311                              <1>             ;db 'PATA1',0,0,0  ; 6
48312                              <1>             ;db 'PATA2',0,0,0  ; 7
48313                              <1>             ;db 'PATA3',0,0,0  ; 8
48314                              <1>             ;db 'WIRELESS'        ; 39
48315                              <1>             ;db 'HDMI',0,0,0,0 ; 40
48316 00010A5A 4E554C4C00000000    <1>             db 'NULL',0,0,0,0 ; 0
48317                              <1>
48318                              <1> NumOfKernelDevNames equ ($-KDEV_NAME) / 8 ; 20 (07/10/2016)
48319                              <1>
48320                              <1> KDEV_NUMBER:
48321 00010A62 0102030405060708 09 <1>             db 1,2,3,4,5,6,7,8,9
48322 00010A6B 0A0B0C0D0E0F101112- <1>             db 10,11,12,13,14,15,16,17,18,19
48323 00010A74 13                  <1>
48324 00010A75 121300              <1>             db 18,19,0
48325                              <1>
48326                              <1> NumOfKernelDevices equ $ - KDEV_NUMBER
48327                              <1>
48328                              <1> KDEV_OADDR:
48329 00010A78 [96020100]          <1>             dd otty ;tty  ; 1
48330 00010A7C [96020100]          <1>             dd sret ;mem  ; 2
48331 00010A80 [96020100]          <1>             dd sret ;fd0  ; 3
48332 00010A84 [96020100]          <1>             dd sret ;fd1  ; 4
48333 00010A88 [96020100]          <1>             dd sret ;hd0  ; 5
48334 00010A8C [96020100]          <1>             dd sret ;hd1  ; 6
48335 00010A90 [96020100]          <1>             dd sret ;hd2  ; 7
48336 00010A94 [96020100]          <1>             dd sret ;hd3  ; 8
48337 00010A98 [96020100]          <1>             dd sret ;lpt  ; 9
48338 00010A9C [96020100]          <1>             dd ocvt ;tty0 ; 10
48339 00010AA0 [96020100]          <1>             dd ocvt ;tty1 ; 11
```

```
48340 00010AA4 [96020100]          <1>             dd ocvt ;tty2 ; 12
48341 00010AA8 [96020100]          <1>             dd ocvt ;tty3 ; 13
48342 00010AAC [96020100]          <1>             dd ocvt ;tty4 ; 14
48343 00010AB0 [96020100]          <1>             dd ocvt ;tty5 ; 15
48344 00010AB4 [96020100]          <1>             dd ocvt ;tty6 ; 16
48345 00010AB8 [96020100]          <1>             dd ocvt ;tty7 ; 17
48346 00010ABC [96020100]          <1>             dd ocvt ;tty8 ; 18
48347 00010AC0 [96020100]          <1>             dd ocvt ;tty9 ; 19
48348                              <1>             ;dd ocvt ;com1 ; 18
48349                              <1>             ;dd ocvt ;com2 ; 19
48350 00010AC4 [96020100]          <1>             dd sret ;null ; 20
48351                              <1> KDEV_CADDR:
48352 00010AC8 [96020100]          <1>             dd ctty ;tty  ; 1
48353 00010ACC [96020100]          <1>             dd cret ;mem  ; 2
48354 00010AD0 [96020100]          <1>             dd cret ;fd0  ; 3
48355 00010AD4 [96020100]          <1>             dd cret ;fd1  ; 4
48356 00010AD8 [96020100]          <1>             dd cret ;hd0  ; 5
48357 00010ADC [96020100]          <1>             dd cret ;hd1  ; 6
48358 00010AE0 [96020100]          <1>             dd cret ;hd2  ; 7
48359 00010AE4 [96020100]          <1>             dd cret ;hd3  ; 8
48360 00010AE8 [96020100]          <1>             dd cret ;lpt  ; 9
48361 00010AEC [96020100]          <1>             dd ocvt ;tty0 ; 10
48362 00010AF0 [96020100]          <1>             dd ccvt ;tty1 ; 11
48363 00010AF4 [96020100]          <1>             dd ccvt ;tty2 ; 12
48364 00010AF8 [96020100]          <1>             dd ccvt ;tty3 ; 13
48365 00010AFC [96020100]          <1>             dd ccvt ;tty4 ; 14
48366 00010B00 [96020100]          <1>             dd ccvt ;tty5 ; 15
48367 00010B04 [96020100]          <1>             dd ccvt ;tty6 ; 16
48368 00010B08 [96020100]          <1>             dd ccvt ;tty7 ; 17
48369 00010B0C [96020100]          <1>             dd ccvt ;tty8 ; 18
48370 00010B10 [96020100]          <1>             dd ccvt ;tty9 ; 19
48371                              <1>             ;dd ccvt ;com1 ; 18
48372                              <1>             ;dd ccvt ;com2 ; 19
48373 00010B14 [96020100]          <1>             dd cret ;null ; 20
48374                              <1>
48375                              <1> KDEV_RADDR:
48376 00010B18 [96020100]          <1>             dd rtty ;tty  ; 1
48377 00010B1C [96020100]          <1>             dd rmem ;mem  ; 2
48378 00010B20 [96020100]          <1>             dd rfd  ;fd0  ; 3
48379 00010B24 [96020100]          <1>             dd rfd  ;fd1  ; 4
48380 00010B28 [96020100]          <1>             dd rhd  ;hd0  ; 5
48381 00010B2C [96020100]          <1>             dd rhd  ;hd1  ; 6
48382 00010B30 [96020100]          <1>             dd rhd  ;hd2  ; 7
48383 00010B34 [96020100]          <1>             dd rhd  ;hd3  ; 8
48384 00010B38 [96020100]          <1>             dd rlpt ;lpt  ; 9
48385 00010B3C [96020100]          <1>             dd rcvt ;tty0 ; 10
48386 00010B40 [96020100]          <1>             dd rcvt ;tty1 ; 11
48387 00010B44 [96020100]          <1>             dd rcvt ;tty2 ; 12
48388 00010B48 [96020100]          <1>             dd rcvt ;tty3 ; 13
48389 00010B4C [96020100]          <1>             dd rcvt ;tty4 ; 14
48390 00010B50 [96020100]          <1>             dd rcvt ;tty5 ; 15
48391 00010B54 [96020100]          <1>             dd rcvt ;tty6 ; 16
48392 00010B58 [96020100]          <1>             dd rcvt ;tty7 ; 17
48393 00010B5C [96020100]          <1>             dd rcvt ;tty8 ; 18
48394 00010B60 [96020100]          <1>             dd rcvt ;tty9 ; 19
48395                              <1>             ;dd rcvt ;com1 ; 18
48396                              <1>             ;dd rcvt ;com2 ; 19
48397 00010B64 [6AFA0000]          <1>             dd rnull ;null ; 20
48398                              <1> KDEV_WADDR:
48399 00010B68 [96020100]          <1>             dd wtty ;tty  ; 1
48400 00010B6C [96020100]          <1>             dd wmem ;mem  ; 2
48401 00010B70 [96020100]          <1>             dd wfd  ;fd0  ; 3
48402 00010B74 [96020100]          <1>             dd wfd  ;fd1  ; 4
48403 00010B78 [96020100]          <1>             dd whd  ;hd0  ; 5
48404 00010B7C [96020100]          <1>             dd whd  ;hd1  ; 6
48405 00010B80 [96020100]          <1>             dd whd  ;hd2  ; 7
48406 00010B84 [96020100]          <1>             dd whd  ;hd3  ; 8
48407 00010B88 [96020100]          <1>             dd wlpt ;lpt  ; 9
48408 00010B8C [96020100]          <1>             dd xmtt ;tty0 ; 10
48409 00010B90 [96020100]          <1>             dd xmtt ;tty1 ; 11
48410 00010B94 [96020100]          <1>             dd xmtt ;tty2 ; 12
48411 00010B98 [96020100]          <1>             dd xmtt ;tty3 ; 13
48412 00010B9C [96020100]          <1>             dd xmtt ;tty4 ; 14
48413 00010BA0 [96020100]          <1>             dd xmtt ;tty5 ; 15
48414 00010BA4 [96020100]          <1>             dd xmtt ;tty6 ; 16
48415 00010BA8 [96020100]          <1>             dd xmtt ;tty7 ; 17
48416 00010BAC [96020100]          <1>             dd xmtt ;tty8 ; 18
48417 00010BB0 [96020100]          <1>             dd xmtt ;tty9 ; 19
48418                              <1>             ;dd xmtt ;com1 ; 18
48419                              <1>             ;dd xmtt ;com2 ; 19
48420 00010BB4 [6BFA0000]          <1>             dd wnull ;null ; 20
48421                              <1>
48422                              <1> ; DEV_ACCESS bits:
48423                              <1>       ; bit 0 = accessable by normal users
48424                              <1>       ; bit 1 = read access permission
48425                              <1>       ; bit 2 = write access permission
48426                              <1>       ; bit 3 = IOCTL permission to users
48427                              <1>       ; bit 4 = block device if it is set
48428                              <1>       ; bit 5 = 16 bit or 1024 byte data
48429                              <1>       ; bit 6 = 32 bit or 2048 byte data
48430                              <1>       ; bit 7 = installable device driver
48431                              <1>
48432                              <1> KDEV_ACCESS: ; 08/10/2016
48433 00010BB8 07                 <1>             db  00000111b; tty, 1
48434 00010BB9 07                 <1>             db  00000111b; mem, 2
48435 00010BBA 8F                 <1>             db  10001111b; fd0, 3
48436 00010BBB 8F                 <1>             db  10001111b; fd1, 4
48437 00010BBC 8F                 <1>             db  10001111b; hd0, 5
48438 00010BBD 8F                 <1>             db  10001111b; hd1, 6
48439 00010BBE 8F                 <1>             db  10001111b; hd2, 7
48440 00010BBF 8F                 <1>             db  10001111b; hd3, 8
48441 00010BC0 07                 <1>             db  00000111b   ; lpt, 9
```

```
48442 00010BC1 07                  <1>             db  00000111b; tty0, 10
48443 00010BC2 07                  <1>             db  00000111b; tty1, 11
48444 00010BC3 07                  <1>             db  00000111b; tty2, 12
48445 00010BC4 07                  <1>             db  00000111b; tty3, 13
48446 00010BC5 07                  <1>             db  00000111b; tty4, 14
48447 00010BC6 07                  <1>             db  00000111b; tty5, 15
48448 00010BC7 07                  <1>             db  00000111b; tty6, 16
48449 00010BC8 07                  <1>             db  00000111b; tty7, 17
48450 00010BC9 07                  <1>             db  00000111b; tty8, 18
48451 00010BCA 07                  <1>             db  00000111b; tty9, 19
48452                              <1>             ;db 00000111b; com1, 18
48453                              <1>             ;db 00000111b; com2, 19
48454 00010BCB 00                  <1>             db  00000000b   ; null, 0
48455                              <1>
48456                              <1> ; 07/10/2016
48457                              <1> NumOfInstallableDevices equ 8
48458                              <1> NUMIDEV            equ NumOfInstallableDevices ; 8
48459                              <1> NUMOFDEVICES equ NumOfKernelDevices + NumOfInstallableDevices
48460                              <1>
48461                              <1> ; 26/02/2017
48462                              <1> ; IRQ Callback (& Signal Response Byte) service availibity
48463                              <1> ; 'syscalbac'
48464                              <1> ; ************************************************
48465                              <1> ; IRQ 00 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15
48466                              <1> ; --- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
48467                              <1> ; --- 00 00 00 01 02 03 00 04 00 05 06 07 08 09 00 00
48468                              <1> ; ************************************************
48469                              <1> IRQenum:
48470 00010BCC 00000010203000400- <1>       db  0,0,0,1,2,3,0,4,0,5,6,7,8,9,0,0
48471 00010BD5 05060708090000     <1>
48472
48473                                  ; 27/08/2014
48474                                  scr_row:
48475 00010BDC E0810B00                      dd 0B8000h + 0A0h + 0A0h + 0A0h ; Row 3
48476                                  scr_col:
48477 00010BE0 00000000                      dd 0
48478
48479                                  Align 4
48480                                        ; 15/04/2016
48481                                        ; TRDOS 386 (TRDOS v2.0)
48482
48483                                        ; 21/08/2014
48484                                  ilist:
48485                                        ;times       32 dd cpu_except ; INT 0 to INT 1Fh
48486                                        ;
48487                                        ; Exception list
48488                                        ; 25/08/2014
48489 00010BE4 [17090000]                  dd    exc0   ; 0h,  Divide-by-zero Error
48490 00010BE8 [1E090000]                  dd    exc1
48491 00010BEC [25090000]                  dd    exc2
48492 00010BF0 [2C090000]                  dd    exc3
48493 00010BF4 [30090000]                  dd    exc4
48494 00010BF8 [34090000]                  dd    exc5
48495 00010BFC [38090000]                  dd    exc6   ; 06h,  Invalid Opcode
48496 00010C00 [3C090000]                  dd    exc7
48497 00010C04 [40090000]                  dd    exc8
48498 00010C08 [44090000]                  dd    exc9
48499 00010C0C [48090000]                  dd    exc10
48500 00010C10 [4C090000]                  dd    exc11
48501 00010C14 [50090000]                  dd    exc12
48502 00010C18 [54090000]                  dd    exc13  ; 0Dh, General Protection Fault
48503 00010C1C [58090000]                  dd    exc14  ; 0Eh, Page Fault
48504 00010C20 [5C090000]                  dd    exc15
48505 00010C24 [60090000]                  dd    exc16
48506 00010C28 [64090000]                  dd    exc17
48507 00010C2C [68090000]                  dd    exc18
48508 00010C30 [6C090000]                  dd    exc19
48509 00010C34 [70090000]                  dd    exc20
48510 00010C38 [74090000]                  dd    exc21
48511 00010C3C [78090000]                  dd    exc22
48512 00010C40 [7C090000]                  dd    exc23
48513 00010C44 [80090000]                  dd    exc24
48514 00010C48 [84090000]                  dd    exc25
48515 00010C4C [88090000]                  dd    exc26
48516 00010C50 [8C090000]                  dd    exc27
48517 00010C54 [90090000]                  dd    exc28
48518 00010C58 [94090000]                  dd    exc29
48519 00010C5C [98090000]                  dd    exc30
48520 00010C60 [9C090000]                  dd    exc31
48521                                  IRQ_list: ; 28/02/2017 ('syscalbac')
48522                                        ; Interrupt list
48523 00010C64 [8B060000]                  dd    timer_int    ; INT 20h
48524                                        ;dd   irq0
48525 00010C68 [FF0D0000]                  dd    kb_int       ; 24/01/2016
48526                                        ;dd   irq1
48527 00010C6C [6D080000]                  dd    irq2
48528                                        ; COM2 int
48529 00010C70 [71080000]                  dd    irq3
48530                                        ; COM1 int
48531 00010C74 [7C080000]                  dd    irq4
48532 00010C78 [87080000]                  dd    irq5
48533                                  ;DISKETTE_INT: ;06/02/2015
48534 00010C7C [B0410000]                  dd    fdc_int            ; 16/02/2015, IRQ 6 handler
48535                                        ;dd   irq6
48536                                  ; Default IRQ 7 handler against spurious IRQs (from master PIC)
48537                                  ; 25/02/2015 (source: http://wiki.osdev.org/8259_PIC)
48538 00010C80 [F60B0000]                  dd    default_irq7 ; 25/02/2015
48539                                        ;dd   irq7
48540                                  ; Real Time Clock Interrupt
48541 00010C84 [F6070000]                  dd    rtc_int            ; 23/02/2015, IRQ 8 handler
48542                                        ;dd   irq8   ; INT 28h
48543 00010C88 [97080000]                  dd    irq9
```

```
48544 00010C8C [9B080000]                 dd    irq10
48545 00010C90 [9F080000]                 dd    irq11
48546 00010C94 [A3080000]                 dd    irq12
48547 00010C98 [A7080000]                 dd    irq13
48548                                   ;HDISK_INT1:  ;06/02/2015
48549 00010C9C [2C4B0000]                 dd    hdc1_int     ; 21/02/2015, IRQ 14 handler
48550                                           ;dd    irq14
48551                                   ;HDISK_INT2:  ;06/02/2015
48552 00010CA0 [534B0000]                 dd    hdc2_int     ; 21/02/2015, IRQ 15 handler
48553                                           ;dd    irq15  ; INT 2Fh
48554                                           ; 14/08/2015
48555                                     ;dd   sysent       ; INT 30h (system calls)
48556
48557                                     ; 15/04/2016
48558                                     ; TRDOS 386(TRDOS v2.0) Software Interrupts
48559
48560 00010CA4 [010D0100]                 dd    int30h       ; Reserved for
48561                                                        ; !!! Retro UNIX (RUNIX) !!!
48562                                                        ; !!! SINGLIX !!! System Calls
48563 00010CA8 [F2140000]                 dd    int31h       ; Video BIOS (IBM PC/AT, Int 10h)
48564 00010CAC [1E0C0000]                 dd    int32h       ; Keyboard Functions (IBM PC/AT, Int 16h)
48565 00010CB0 [66420000]                 dd    int33h       ; DISK I/O (IBM PC/AT, Int 13h)
48566 00010CB4 [39F30000]                 dd    int34h       ; #IOCTL# (I/O port access support for ring 3)
48567 00010CB8 [81590000]                 dd    int35h       ; Time/Date Functions (IBM PC/AT, Int 1Ah)
48568 00010CBC [AA0A0000]                 dd    ignore_int   ; INT 36h : Timer Functions
48569 00010CC0 [AA0A0000]                 dd    ignore_int   ; INT 37h
48570 00010CC4 [AA0A0000]                 dd    ignore_int   ; INT 38h
48571 00010CC8 [AA0A0000]                 dd    ignore_int   ; INT 39h
48572 00010CCC [AA0A0000]                 dd    ignore_int   ; INT 3Ah
48573 00010CD0 [AA0A0000]                 dd    ignore_int   ; INT 3Bh
48574 00010CD4 [AA0A0000]                 dd    ignore_int   ; INT 3Ch
48575 00010CD8 [AA0A0000]                 dd    ignore_int   ; INT 3Dh
48576 00010CDC [AA0A0000]                 dd    ignore_int   ; INT 3Eh
48577 00010CE0 [AA0A0000]                 dd    ignore_int   ; INT 3Fh
48578 00010CE4 [61C30000]                 dd    sysent       ; INT 40h : !!! TRDOS 386 System Calls !!!
48579                                     ;dd   ignore_int
48580 00010CE8 00000000                   dd    0
48581
48582                                   ; 20/08/2014
48583                                     ; /* This is the default interrupt "handler" :-) */
48584                                     ; Linux v0.12 (head.s)
48585                                   int_msg:
48586 00010CEC 556E6B6E6F776E2069-        db "Unknown interrupt ! ", 0
48587 00010CF5 6E7465727275707420-
48588 00010CFE 212000
48589
48590                                   ; 15/04/2016
48591                                   ; TRDOS 386 (TRDOS v2.0)
48592
48593                                   ; 29/04/2016
48594                                   int30h:
48595                                   trdos_isc_routine:
48596                                         ; 02/05/2016
48597                                         ; 01/05/2016
48598                                         ; 29/04/2016
48599                                         ; 18/04/2016
48600                                         ; 15/04/2016 (TRDOS 386 = TRDOS v2.0)
48601                                         ; 17/04/2011 (TRDOS v1.0, 'IFC.ASM')
48602                                         ; 03/02/2011 ('trdos_ifc_routine')
48603                                         ;
48604 00010D01 8B1C24                     mov   ebx, [esp] ; EIP (next)
48605 00010D04 83EB02                     sub   ebx, 2 ; EIP (CD ##h)
48606
48607 00010D07 89C1                       mov   ecx, eax
48608 00010D09 8A4301                     mov   al, [ebx+1] ; CDh ##h
48609
48610 00010D0C 66BA1000                   mov   dx, KDATA
48611 00010D10 8EDA                       mov   ds, dx
48612 00010D12 8EC2                       mov   es, dx
48613
48614 00010D14 FC                         cld
48615 00010D15 8B15[C84D0100]             mov   edx, [k_page_dir]
48616 00010D1B 0F22DA                     mov   cr3, edx
48617
48618 00010D1E E8A625FFFF                 call  bytetohex
48619 00010D23 66A3[89090100]             mov   [int_num_str], ax
48620
48621 00010D29 89D8                       mov   eax, ebx ; EIP
48622 00010D2B E8D925FFFF                 call  dwordtohex
48623 00010D30 8915[A5090100]             mov   [eip_str], edx
48624 00010D36 A3[A9090100]               mov   [eip_str+4], eax
48625
48626 00010D3B 89C8                       mov   eax, ecx
48627 00010D3D E8C725FFFF                 call  dwordtohex
48628 00010D42 8915[94090100]             mov   [eax_str], edx
48629 00010D48 A3[98090100]               mov   [eax_str+4], eax
48630
48631 00010D4D 43                         inc   ebx
48632 00010D4E 8A03                       mov   al, [ebx] ; Interrupt number
48633
48634                                   trdos_isc_handler:
48635 00010D50 80FE30                     cmp   dh, 30h ; Retro UNIX, SINGLIX System calls
48636 00010D53 7507                       jne   short trdos_usi_handler
48637 00010D55 BE[26090100]               mov   esi, isc_msg
48638 00010D5A EB05                       jmp   short loc_write_inv_system_call_msg
48639
48640                                   trdos_usi_handler:
48641 00010D5C BE[3C090100]               mov   esi, usi_msg
48642
48643                                   loc_write_inv_system_call_msg:
48644 00010D61 E8F755FFFF                 call  print_msg
48645                                     ; 29/04/2016
```

```
48646 00010D66 BE[72090100]               mov   esi, inv_msg_for_trdos_v2
48647 00010D6B E8ED55FFFF                 call  print_msg
48648
48649                                  loc_ifc_terminate_process:
48650                                     ; u.uno = process number
48651                                     ; 29/04/2016
48652
48653                                     ; 02/05/2016
48654 00010D70 FE05[5B030600]             inc   byte [sysflg] ; 0FFh -> 0
48655
48656 00010D76 B801000000                 mov   eax, 1
48657 00010D7B E9B6B8FFFF                 jmp   sysexit
48658
48659                                  ; 07/03/2015
48660                                  ; Temporary Code
48661                                  display_disks:
48662 00010D80 803D[F65C0000]00            cmp   byte [fd0_type], 0
48663 00010D87 7605                       jna   short ddsks1
48664 00010D89 E87D000000                 call  pdskm
48665                                  ddsks1:
48666 00010D8E 803D[F75C0000]00            cmp   byte [fd1_type], 0
48667 00010D95 760C                       jna   short ddsks2
48668 00010D97 C605[DB0E0100]31           mov   byte [dskx], '1'
48669 00010D9E E868000000                 call  pdskm
48670                                  ddsks2:
48671 00010DA3 803D[F85C0000]00            cmp   byte [hd0_type], 0
48672 00010DAA 7654                       jna   short ddsk6
48673 00010DAC 66C705[D90E0100]68-        mov   word [dsktype], 'hd'
48674 00010DB4 64
48675 00010DB5 C605[DB0E0100]30           mov   byte [dskx], '0'
48676 00010DBC E84A000000                 call  pdskm
48677                                  ddsks3:
48678 00010DC1 803D[F95C0000]00            cmp   byte [hd1_type], 0
48679 00010DC8 7636                       jna   short ddsk6
48680 00010DCA C605[DB0E0100]31           mov   byte [dskx], '1'
48681 00010DD1 E835000000                 call  pdskm
48682                                  ddsks4:
48683 00010DD6 803D[FA5C0000]00            cmp   byte [hd2_type], 0
48684 00010DDD 7621                       jna   short ddsk6
48685 00010DDF C605[DB0E0100]32           mov   byte [dskx], '2'
48686 00010DE6 E820000000                 call  pdskm
48687                                  ddsks5:
48688 00010DEB 803D[FB5C0000]00            cmp   byte [hd3_type], 0
48689 00010DF2 760C                       jna   short ddsk6
48690 00010DF4 C605[DB0E0100]33           mov   byte [dskx], '3'
48691 00010DFB E80B000000                 call  pdskm
48692                                  ddsk6:
48693 00010E00 BE[030F0100]               mov   esi, nextline
48694 00010E05 E806000000                 call  pdskml
48695                                  pdskm_ok:
48696 00010E0A C3                         retn
48697                                  pdskm:
48698 00010E0B BE[D70E0100]               mov   esi, dsk_ready_msg
48699                                  pdskml:
48700 00010E10 AC                         lodsb
48701 00010E11 08C0                       or    al, al
48702 00010E13 74F5                       jz    short pdskm_ok
48703 00010E15 56                         push  esi
48704                                     ; 13/05/2016
48705 00010E16 BB07000000                   mov     ebx, 7  ; Black background,
48706                                                       ; light gray forecolor
48707                                                       ; Video page 0 (bh=0)
48708 00010E1B E8920EFFFF                 call  _write_tty
48709 00010E20 5E                         pop   esi
48710 00010E21 EBED                       jmp   short pdskml
48711
48712 00010E23 90                      Align 2
48713                                     ; 21/08/2014
48714                                  exc_msg:
48715 00010E24 435055206578636570-        db "CPU exception ! "
48716 00010E2D 74696F6E202120
48717                                  excnstr:          ; 25/08/2014
48718 00010E34 3F3F68202045495020-        db "??h", "  EIP : "
48719 00010E3D 3A20
48720                                  EIPstr: ; 29/08/2014
48721 00010E3F 00<rept>                   times 12 db 0
48722
48723                                     ; 23/02/2015
48724                                     ; 25/08/2014
48725                                  ;scounter:
48726                                  ;     db 5
48727                                  ;     db 19
48728
48729                                  ; 06/11/2014
48730                                  ; Memory Information message
48731                                  ; 14/08/2015
48732                                  msg_memory_info:
48733 00010E4B 07                         db    07h
48734 00010E4C 0D0A                       db    0Dh, 0Ah
48735                                     ;db    "MEMORY ALLOCATION INFO", 0Dh, 0Ah, 0Dh, 0Ah
48736 00010E4E 546F74616C206D656D-        db    "Total memory : "
48737 00010E57 6F7279203A20
48738                                  mem_total_b_str: ; 10 digits
48739 00010E5D 303030303030303030-        db    "0000000000 bytes", 0Dh, 0Ah
48740 00010E66 302062797465730D0A
48741 00010E6F 202020202020202020-        db    "               ", 20h, 20h, 20h
48742 00010E78 202020202020202020
48743                                  mem_total_p_str: ; 7 digits
48744 00010E81 303030303030302070-        db    "0000000 pages", 0Dh, 0Ah
48745 00010E8A 616765730D0A
48746 00010E90 0D0A                       db    0Dh, 0Ah
48747 00010E92 46726565206D656D6F-        db    "Free memory  : "
```

```
48748 00010E9B 727920203A20
48749                                  free_mem_b_str:  ; 10 digits
48750 00010EA1 3F3F3F3F3F3F3F3F3F-             db     "?????????? bytes", 0Dh, 0Ah
48751 00010EAA 3F2062797465730D0A
48752 00010EB3 2020202020202020-               db     "               ", 20h, 20h, 20h
48753 00010EBC 2020202020202020
48754                                  free_mem_p_str:  ; 7 digits
48755 00010EC5 3F3F3F3F3F3F3F2070-             db     "??????? pages", 0Dh, 0Ah
48756 00010ECE 616765730D0A
48757 00010ED4 0D0A00                          db     0Dh, 0Ah, 0
48758
48759                                  dsk_ready_msg:
48760 00010ED7 0D0A                            db     0Dh, 0Ah
48761                                  dsktype:
48762 00010ED9 6664                            db     'fd'
48763                                  dskx:
48764 00010EDB 30                              db     '0'
48765 00010EDC 20                              db     20h
48766 00010EDD 697320524541445920-             db     'is READY ...'
48767 00010EE6 2E2E2E
48768 00010EE9 00                              db     0
48769
48770                                  setup_error_msg:
48771 00010EEA 0D0A                            db 0Dh, 0Ah
48772 00010EEC 4469736B2053657475-             db 'Disk Setup Error !'
48773 00010EF5 70204572726F722021
48774 00010EFE 0D0A00                          db 0Dh, 0Ah,0
48775
48776                                  next2line: ; 08/02/2016
48777 00010F01 0D0A                            db     0Dh, 0Ah
48778                                  nextline:
48779 00010F03 0D0A00                          db     0Dh, 0Ah, 0
48780
48781                                  ; KERNEL - SYSINIT Messages
48782                                  ; 24/08/2015
48783                                  ; 13/04/2015 - (Retro UNIX 386 v1 Beginning)
48784                                  ; 14/07/2013
48785                                  ;kernel_init_err_msg:
48786                                  ;     db 0Dh, 0Ah
48787                                  ;     db 07h
48788                                  ;     db 'Kernel initialization ERROR !'
48789                                  ;     db 0Dh, 0Ah, 0
48790
48791                                  ;welcome_msg:
48792                                  ;     db 0Dh, 0Ah
48793                                  ;     db 07h
48794                                  ;     db 'Welcome to TRDOS 386 Operating System !'
48795                                  ;     db 0Dh, 0Ah
48796                                  ;     db 'by Erdogan Tan - 22/07/2017 (v2.0.0)'
48797                                  ;     db 0Dh, 0Ah, 0
48798
48799                                  panic_msg:
48800 00010F06 0D0A07                          db 0Dh, 0Ah, 07h
48801 00010F09 4552524F523A204B65-             db 'ERROR: Kernel Panic !'
48802 00010F12 726E656C2050616E69-
48803 00010F1B 632021
48804 00010F1E 0D0A00                          db 0Dh, 0Ah, 0
48805
48806                                  ;msgl_drv_not_ready:
48807                                  ;     db 07h, 0Dh, 0Ah
48808                                  ;       db 'Drive not ready or read error !'
48809                                  ;       db 0Dh, 0Ah, 0
48810
48811                                  starting_msg:
48812 00010F21 5475726B6973682052-             db "Turkish Rational DOS v2.0 [22/07/2017] ...", 0
48813 00010F2A 6174696F6E616C2044-
48814 00010F33 4F532076322E30205B-
48815 00010F3C 32322F30372F323031-
48816 00010F45 375D202E2E2E00
48817                                  NextLine:
48818 00010F4C 0D0A00                          db 0Dh, 0Ah, 0
48819
48820                                      %include 'audio.s' ; 03/04/2017
48821                                <1> ; ****************************************************************************
48822                                <1> ; TRDOS386.ASM (TRDOS 386 Kernel) - v2.0.0 - audio.s
48823                                <1> ; ----------------------------------------------------------------------------
48824                                <1> ; Last Update: 13/07/2017
48825                                <1> ; ----------------------------------------------------------------------------
48826                                <1> ; Beginning: 03/04/2017
48827                                <1> ; ----------------------------------------------------------------------------
48828                                <1> ; Assembler: NASM version 2.11 (trdos386.s)
48829                                <1> ; ****************************************************************************
48830                                <1>
48831                                <1> ; AUDIO CONTROLLER & CODEC DEFINITIONS & CODE FOR TRDOS 386
48832                                <1>
48833                                <1> ;============================================================================
48834                                <1> ;               EQUATES
48835                                <1> ;============================================================================
48836                                <1>
48837                                <1> ; PCI EQUATES
48838                                <1>
48839                                <1> BIT0  EQU 1
48840                                <1> BIT1  EQU 2
48841                                <1> BIT2  EQU 4
48842                                <1> BIT3  EQU 8
48843                                <1> BIT4  EQU 10h
48844                                <1> BIT5  EQU 20h
48845                                <1> BIT6  EQU 40h
48846                                <1> BIT7  EQU 80h
48847                                <1> BIT8  EQU 100h
48848                                <1> BIT9  EQU 200h
48849                                <1> BIT10 EQU 400h
```

```
48850                                  <1> BIT11 EQU 800h
48851                                  <1> BIT12 EQU 1000h
48852                                  <1> BIT13 EQU 2000h
48853                                  <1> BIT14 EQU 4000h
48854                                  <1> BIT15 EQU 8000h
48855                                  <1> BIT16 EQU 10000h
48856                                  <1> BIT17 EQU 20000h
48857                                  <1> BIT18 EQU 40000h
48858                                  <1> BIT19 EQU 80000h
48859                                  <1> BIT20 EQU 100000h
48860                                  <1> BIT21 EQU 200000h
48861                                  <1> BIT22 EQU 400000h
48862                                  <1> BIT23 EQU 800000h
48863                                  <1> BIT24 EQU 1000000h
48864                                  <1> BIT25 EQU 2000000h
48865                                  <1> BIT26 EQU 4000000h
48866                                  <1> BIT27 EQU 8000000h
48867                                  <1> BIT28 EQU 10000000h
48868                                  <1> BIT29 EQU 20000000h
48869                                  <1> BIT30 EQU 40000000h
48870                                  <1> BIT31 EQU 80000000h
48871                                  <1> NOT_BIT31 EQU 7FFFFFFFh
48872                                  <1>
48873                                  <1> ; PCI equates
48874                                  <1> ; PCI function address (PFA)
48875                                  <1> ; bit 31 = 1
48876                                  <1> ; bit 23:16 = bus number     (0-255)
48877                                  <1> ; bit 15:11 = device number  (0-31)
48878                                  <1> ; bit 10:8 = function number (0-7)
48879                                  <1> ; bit 7:0 = register number  (0-255)
48880                                  <1>
48881                                  <1> IO_ADDR_MASK    EQU     0FFFEh  ; mask off bit 0 for reading BARs
48882                                  <1> PCI_INDEX_PORT  EQU     0CF8h
48883                                  <1> PCI_DATA_PORT   EQU     0CFCh
48884                                  <1> PCI32           EQU     BIT31   ; bitflag to signal 32bit access
48885                                  <1> PCI16           EQU     BIT30   ; bitflag for 16bit access
48886                                  <1> NOT_PCI32_PCI16    EQU   03FFFFFFFh ; NOT BIT31+BIT30 ; 19/03/2017
48887                                  <1>
48888                                  <1> PCI_FN0         EQU     0 << 8
48889                                  <1> PCI_FN1         EQU     1 << 8
48890                                  <1> PCI_FN2         EQU     2 << 8
48891                                  <1> PCI_FN3         EQU     3 << 8
48892                                  <1> PCI_FN4         EQU     4 << 8
48893                                  <1> PCI_FN5         EQU     5 << 8
48894                                  <1> PCI_FN6         EQU     6 << 8
48895                                  <1> PCI_FN7         EQU     7 << 8
48896                                  <1>
48897                                  <1> PCI_CMD_REG  EQU    04h    ; reg 04, command reg
48898                                  <1> IO_ENA              EQU    BIT0   ; i/o decode enable
48899                                  <1> MEM_ENA             EQU    BIT1   ; memory decode enable
48900                                  <1> BM_ENA              EQU     BIT2 ; bus master enable
48901                                  <1>
48902                                  <1> ; VIA VT8233 EQUATES
48903                                  <1>
48904                                  <1> VIA_VID             equ 1106h    ; VIA's PCI vendor ID
48905                                  <1> VT8233_DID      equ 3059h ; VT8233 (VT8235) device ID
48906                                  <1>
48907                                  <1> PCI_IO_BASE          equ 10h
48908                                  <1> AC97_INT_LINE        equ 3Ch
48909                                  <1> VIA_ACLINK_CTRL      equ 41h
48910                                  <1> VIA_ACLINK_STAT      equ 40h
48911                                  <1> VIA_ACLINK_C00_READY equ 01h ; primary codec ready
48912                                  <1>
48913                                  <1> VIA_REG_AC97     equ 80h ; dword
48914                                  <1>
48915                                  <1> VIA_ACLINK_CTRL_ENABLE    equ   80h ; 0: disable, 1: enable
48916                                  <1> VIA_ACLINK_CTRL_RESET     equ   40h ; 0: assert, 1: de-assert
48917                                  <1> VIA_ACLINK_CTRL_SYNC      equ   20h ; 0: release SYNC, 1: force SYNC hi
48918                                  <1> VIA_ACLINK_CTRL_VRA       equ   08h ; 0: disable VRA, 1: enable VRA
48919                                  <1> VIA_ACLINK_CTRL_PCM       equ   04h ; 0: disable PCM, 1: enable PCM
48920                                  <1> VIA_ACLINK_CTRL_INIT      equ  (VIA_ACLINK_CTRL_ENABLE +
VIA_ACLINK_CTRL_RESET +                                VIA_ACLINK_CTRL_PCM +                                  VIA_ACLINK_CTRL_VRA)
 48921                                 <1>
 48922                                 <1> CODEC_AUX_VOL             equ   04h
 48923                                 <1> VIA_REG_AC97_BUSY  equ   01000000h ;(1<<24)
 48924                                 <1> VIA_REG_AC97_CMD_SHIFT    equ   10h ; 16
 48925                                 <1> VIA_REG_AC97_PRIMARY_VALID equ 02000000h ;(1<<25)
 48926                                 <1> VIA_REG_AC97_READ  equ   00800000h ;(1<<23)
 48927                                 <1> VIA_REG_AC97_CODEC_ID_SHIFT   equ  1Eh ; 30
 48928                                 <1> VIA_REG_AC97_CODEC_ID_PRIMARY equ  0
 48929                                 <1> VIA_REG_AC97_DATA_SHIFT equ   0
 48930                                 <1> VIADEV_PLAYBACK         equ   0
 48931                                 <1> VIA_REG_OFFSET_STATUS   equ   0    ;; byte - channel status
 48932                                 <1> VIA_REG_OFFSET_CONTROL  equ   01h  ;; byte - channel control
 48933                                 <1> VIA_REG_CTRL_START equ   80h  ;; WO
 48934                                 <1> VIA_REG_CTRL_TERMINATE  equ   40h  ;; WO
 48935                                 <1> VIA_REG_CTRL_PAUSE      equ   08h  ;; RW
 48936                                 <1> VIA_REG_CTRL_RESET      equ   01h  ;; RW - probably reset? undocumented
 48937                                 <1> VIA_REG_OFFSET_STOP_IDX equ   08h  ;; dword - stop index, channel type, sample rate
 48938                                 <1> VIA8233_REG_TYPE_16BIT  equ   200000h ;; RW
 48939                                 <1> VIA8233_REG_TYPE_STEREO equ   100000h ;; RW
 48940                                 <1> VIA_REG_OFFSET_CURR_INDEX equ 0Fh ;; byte - channel current index (for via8233 only)
 48941                                 <1> VIA_REG_OFFSET_TABLE_PTR equ  04h  ;; dword - channel table pointer
 48942                                 <1> VIA_REG_OFFSET_CURR_PTR equ   04h  ;; dword - channel current pointer
 48943                                 <1> VIA_REG_OFS_PLAYBACK_VOLUME_L equ  02h ;; byte
 48944                                 <1> VIA_REG_OFS_PLAYBACK_VOLUME_R equ  03h ;; byte
 48945                                 <1> VIA_REG_CTRL_AUTOSTART    equ   20h
 48946                                 <1> VIA_REG_CTRL_INT_EOL      equ   02h
 48947                                 <1> VIA_REG_CTRL_INT_FLAG     equ   01h
 48948                                 <1> VIA_REG_CTRL_INT   equ  (VIA_REG_CTRL_INT_FLAG +
VIA_REG_CTRL_INT_EOL +                                 VIA_REG_CTRL_AUTOSTART)
 48949                                 <1>
```

```
48950                                  <1> VIA_REG_STAT_STOPPED      equ   04h    ;; RWC
48951                                  <1> VIA_REG_STAT_EOL   equ   02h    ;; RWC
48952                                  <1> VIA_REG_STAT_FLAG  equ   01h    ;; RWC
48953                                  <1> VIA_REG_STAT_ACTIVE       equ   80h    ;; RO
48954                                  <1> ; 28/11/2016
48955                                  <1> VIA_REG_STAT_LAST  equ   40h    ;; RO
48956                                  <1> VIA_REG_STAT_TRIGGER_QUEUED equ 08h  ;; RO
48957                                  <1> VIA_REF_CTRL_INT_STOP     equ   04h  ; Interrupt on Current Index = Stop Index
48958                                  <1>                               ; and End of Block
48959                                  <1>
48960                                  <1> VIA_REG_OFFSET_CURR_COUNT equ 0Ch ;; dword - channel current count, index
48961                                  <1>
48962                                  <1> PORTB       EQU    061h
48963                                  <1> REFRESH_STATUS     EQU    010h   ; Refresh signal status
48964                                  <1>
48965                                  <1> ; AC97 Codec registers.
48966                                  <1>
48967                                  <1> ; each codec/mixer register is 16bits
48968                                  <1>
48969                                  <1> CODEC_RESET_REG                equ     00h  ; reset codec
48970                                  <1> CODEC_MASTER_VOL_REG           equ     02h  ; master volume
48971                                  <1> CODEC_HP_VOL_REG               equ     04h  ; headphone volume
48972                                  <1> CODEC_MASTER_MONO_VOL_REG      equ     06h  ; master mono volume
48973                                  <1> CODEC_MASTER_TONE_REG          equ     08h  ; master tone (R+L)
48974                                  <1> CODEC_PCBEEP_VOL_REG           equ     0Ah  ; PC beep volume
48975                                  <1> CODEC_PHONE_VOL_REG            equ     0Bh  ; phone volume
48976                                  <1> CODEC_MIC_VOL_REG              equ     0Eh  ; MIC volume
48977                                  <1> CODEC_LINE_IN_VOL_REG          equ     10h  ; line input volume
48978                                  <1> CODEC_CD_VOL_REG               equ     12h  ; CD volume
48979                                  <1> CODEC_VID_VOL_REG              equ     14h  ; video volume
48980                                  <1> CODEC_AUX_VOL_REG              equ     16h  ; aux volume
48981                                  <1> CODEC_PCM_OUT_REG              equ     18h  ; PCM output volume
48982                                  <1> CODEC_RECORD_SELECT_REG        equ     1Ah  ; record select input
48983                                  <1> CODEC_RECORD_VOL_REG           equ     1Ch  ; record volume
48984                                  <1> CODEC_RECORD_MIC_VOL_REG       equ     1Eh  ; record mic volume
48985                                  <1> CODEC_GP_REG                   equ     20h  ; general purpose
48986                                  <1> CODEC_3D_CONTROL_REG           equ     22h  ; 3D control
48987                                  <1> ; 24h is reserved
48988                                  <1> CODEC_POWER_CTRL_REG           equ     26h  ; powerdown control
48989                                  <1> CODEC_EXT_AUDIO_REG            equ     28h  ; extended audio
48990                                  <1> CODEC_EXT_AUDIO_CTRL_REG       equ     2Ah  ; extended audio control
48991                                  <1> CODEC_PCM_FRONT_DACRATE_REG    equ     2Ch  ; PCM out sample rate
48992                                  <1> CODEC_PCM_SURND_DACRATE_REG    equ     2Eh  ; surround sound sample rate
48993                                  <1> CODEC_PCM_LFE_DACRATE_REG      equ     30h  ; LFE sample rate
48994                                  <1> CODEC_LR_ADCRATE_REG           equ     32h  ; PCM in sample rate
48995                                  <1> CODEC_MIC_ADCRATE_REG          equ     34h  ; mic in sample rate
48996                                  <1>
48997                                  <1> ; VT8233 SGD bits (21/04/2017)
48998                                  <1> FLAG  EQU BIT30
48999                                  <1> EOL   EQU BIT31
49000                                  <1>
49001                                  <1> ; INTEL ICH EQUATES
49002                                  <1> ; 28/05/2017
49003                                  <1> INTEL_VID   equ   8086h  ; Intel's PCI vendor ID
49004                                  <1> ICH_DID           equ   2415h  ; ICH (82801AA) device ID
49005                                  <1> NAMBAR_REG     equ      10h    ; native audio mixer Base Address Register
49006                                  <1> NABMBAR_REG    equ      14h    ; native audio bus mastering Base Addr Reg
49007                                  <1>
49008                                  <1> PI_CR_REG      equ    0Bh     ; PCM in Control Register
49009                                  <1> PO_CR_REG   equ    1Bh     ; PCM out Control Register
49010                                  <1> MC_CR_REG   equ    2Bh     ; MIC in Control Register
49011                                  <1>
49012                                  <1> PI_SR_REG   equ    6       ; PCM in Status register
49013                                  <1> PO_SR_REG   equ    16h     ; PCM out Status register
49014                                  <1> MC_SR_REG   equ    26h     ; MIC in Status register
49015                                  <1>
49016                                  <1> IOCE        equ    BIT4    ; interrupt on complete enable.
49017                                  <1> FEIFE       equ    BIT3    ; set if you want an interrupt to fire
49018                                  <1> LVBIE       equ    BIT2    ; last valid buffer interrupt enable.
49019                                  <1> RR          equ    BIT1    ; reset registers.  Nukes all regs
49020                                  <1>                               ; except bits 4:2 of this register.
49021                                  <1>                               ; Only set this bit if BIT 0 is 0
49022                                  <1> RPBM        equ    BIT0    ; Run/Pause
49023                                  <1>                           ; set this bit to start the codec!
49024                                  <1>
49025                                  <1> PI_BDBAR_REG equ    0       ; PCM in buffer descriptor BAR
49026                                  <1> PO_BDBAR_REG equ    10h     ; PCM out buffer descriptor BAR
49027                                  <1> MC_BDBAR_REG equ    20h     ; MIC in buffer descriptor BAR
49028                                  <1>
49029                                  <1> PI_CIV_REG  equ    4       ; PCM in current Index value (RO)
49030                                  <1> PO_CIV_REG  equ    14h     ; PCM out current Index value (RO)
49031                                  <1> MC_CIV_REG  equ    24h     ; MIC in current Index value (RO)
49032                                  <1>
49033                                  <1> PI_LVI_REG  equ    5       ; PCM in Last Valid Index
49034                                  <1> PO_LVI_REG  equ    15h     ; PCM out Last Valid Index
49035                                  <1> MC_LVI_REG  equ    25h     ; MIC in Last Valid Index
49036                                  <1>
49037                                  <1> IOC         equ    BIT31; Fire an interrupt whenever this
49038                                  <1>                           ; buffer is complete.
49039                                  <1> BUP         equ    BIT30; Buffer Underrun Policy.
49040                                  <1>
49041                                  <1> GLOB_CNT_REG equ    2Ch     ; Global Control Register
49042                                  <1> GLOB_STS_REG equ    30h     ; Global Status register (RO)
49043                                  <1>
49044                                  <1> CTRL_ST_CREADY     equ   BIT8+BIT9+BIT28 ; Primary Codec Ready
49045                                  <1>
49046                                  <1> CODEC_REG_POWERDOWN   equ 26h
49047                                  <1> CODEC_REG_ST          equ 26h
49048                                  <1>
49049                                  <1> ; 22/06/2017
49050                                  <1> PO_PICB_REG  equ 18h      ; PCM Out Position In Current Buffer Register
49051                                  <1>
```

```
49052                                  <1> ;============================================================================
49053                                  <1> ;                CODE
49054                                  <1> ;============================================================================
49055                                  <1>
49056                                  <1> ; CODE for INTEL ICH AC'97 AUDIO CONTROLLER
49057                                  <1>
49058                                  <1> DetectICH:
49059                                  <1>       ; 10/06/2017
49060                                  <1>       ; 05/06/2017
49061                                  <1>       ; 29/05/2017
49062                                  <1>       ; 28/05/2017
49063 00010F4F B886801524              <1>       mov     eax, (ICH_DID << 16) + INTEL_VID
49064 00010F54 E876000000              <1>         call    pciFindDevice
49065 00010F59 730D                    <1>         jnc     short d_ac97_1
49066                                  <1> d_ac97_0:
49067                                  <1> ; couldn't find the audio device!
49068 00010F5B C3                      <1>       retn
49069                                  <1>
49070                                  <1> ; CODE for VIA VT8233 AUDIO CONTROLLER
49071                                  <1>
49072                                  <1> DetectVT8233:
49073                                  <1>       ; 10/06/2017
49074                                  <1>       ; 05/06/2017
49075                                  <1>       ; 29/05/2017
49076                                  <1>       ; 03/04/2017
49077 00010F5C B806115930              <1>       mov     eax, (VT8233_DID << 16) + VIA_VID
49078 00010F61 E869000000              <1>         call    pciFindDevice
49079                                  <1> ;       jnc     short d_vt8233_0
49080                                  <1> ; couldn't find the audio device!
49081                                  <1> ;     retn
49082 00010F66 72F3                    <1>       jc     short d_ac97_0  ; 28/05/2017
49083                                  <1> d_vt8233_0:
49084                                  <1>       ; 24/03/2017 ('player.asm')
49085                                  <1>       ; 12/11/2016
49086                                  <1>       ; Erdogan Tan - 8/11/2016
49087                                  <1>       ; References: Kolibrios - vt823x.asm (2016)
49088                                  <1>       ;           VIA VT8235 V-Link South Bridge (VT8235-VIA.PDF)(2002)
49089                                  <1>       ;           lowlevel.eu - AC97 (2016)
49090                                  <1>       ;           .wav player for DOS by Jeff Leyda (2002) -this file-
49091                                  <1>       ;           Linux kernel - via82xx.c (2016)
49092                                  <1> d_ac97_1:
49093                                  <1>       ; eax = BUS/DEV/FN
49094                                  <1>       ;     00000000BBBBBBBBDDDDDFFF00000000
49095                                  <1>       ; edx = DEV/VENDOR
49096                                  <1>       ;     DDDDDDDDDDDDDDDDVVVVVVVVVVVVVVVV
49097                                  <1>
49098 00010F68 A3[54610100]            <1>       mov    [audio_dev_id], eax
49099 00010F6D 8915[58610100]          <1>       mov    [audio_vendor], edx
49100                                  <1>
49101                                  <1>       ; init controller
49102 00010F73 B004                    <1>       mov    al, PCI_CMD_REG ; command register (04h)
49103 00010F75 E8E2000000              <1>       call   pciRegRead32
49104                                  <1>
49105                                  <1>       ; eax = BUS/DEV/FN/REG
49106                                  <1>       ; edx = STATUS/COMMAND
49107                                  <1>       ;     SSSSSSSSSSSSSSSSCCCCCCCCCCCCCCCC
49108 00010F7A 8915[5C610100]          <1>       mov    [audio_stats_cmd], edx
49109                                  <1>
49110 00010F80 B010                    <1>       mov    al, PCI_IO_BASE ; IO base address register (10h)
49111                                  <1>       ;mov   al, NAMBAR_REG      ; Native Audio Mixer BAR (10h)
49112 00010F82 E8D5000000              <1>       call   pciRegRead32
49113                                  <1>
49114 00010F87 66813D[58610100]86-     <1>       cmp    word [audio_vendor], 8086h ; AC'97 ?
49115 00010F8F 80                      <1>
49116 00010F90 751F                    <1>       jne    short d_vt8233_1
49117                                  <1>
49118 00010F92 6683E2FE                <1>       and    dx, 0FFFEh ; Audio Codec IO_ADDR_MASK
49119 00010F96 668915[84610100]        <1>       mov    [NAMBAR], dx
49120                                  <1>
49121 00010F9D B014                    <1>       mov    al, NABMBAR_REG ; Native Audio Bus Mastering BAR (14h)
49122 00010F9F E8B8000000              <1>       call   pciRegRead32
49123                                  <1>
49124 00010FA4 6683E2C0                <1>       and    dx, 0FFC0h ; Audio Controller IO_ADDR_MASK
49125 00010FA8 668915[86610100]        <1>       mov    [NABMBAR], dx
49126                                  <1>         ;mov [audio_io_base], dx
49127                                  <1>
49128 00010FAF EB0B                    <1>       jmp    short d_ac97_2
49129                                  <1>
49130                                  <1> d_vt8233_1:
49131 00010FB1 6683E2C0                <1>       and     dx, 0FFC0h ; Audio Controller IO_ADDR_MASK
49132 00010FB5 668915[52610100]        <1>         mov     [audio_io_base], dx
49133                                  <1>
49134                                  <1> d_ac97_2:
49135                                  <1>       ; 10/06/2017
49136 00010FBC B03C                    <1>       mov    al, AC97_INT_LINE ; Interrupt Line Register (3Ch)
49137                                  <1>       ;call  pciRegRead32
49138 00010FBE E886000000              <1>       call   pciRegRead8
49139                                  <1>
49140                                  <1>       ;and   edx, 0FFh
49141 00010FC3 6681E2FF00              <1>       and    dx, 0FFh
49142                                  <1>
49143 00010FC8 8815[4F610100]          <1>       mov     [audio_intr], dl
49144                                  <1>
49145 00010FCE C3                      <1>       retn
49146                                  <1>
49147                                  <1>       ;; (Note: Interrupts are already enabled by TRDOS 386 kernel!)
49148                                  <1>       ;mov   cx, dx
49149                                  <1>
49150                                  <1>       ;in    al, 0A1h ; irq 8-15
49151                                  <1>       ;mov   ah, al
49152                                  <1>       ;in    al, 21h  ; irq 0-7
49153                                  <1>       ;btr   ax, dx  ; unmask ; 17/03/2017
```

```
49154                                  <1>       ;;bts  ax, dx   ; MASK interrupt ; 10/06/2017
49155                                  <1>       ;out   21h, al  ; irq <= 7
49156                                  <1>       ;mov   al, ah
49157                                  <1>       ;out   0A1h, al ; irq > 7
49158                                  <1>       ;
49159                                  <1>
49160                                  <1>       ; 10/06/2017
49161                                  <1>       ; === Intel ICH I/O Controller Hub Datasheet, Section 8.1.16 ===
49162                                  <1>       ; PRQ[n]_ROUT Register (61h, PRQB) Bit 7:
49163                                  <1>       ; Interrupt Routing Enable (IRQEN).
49164                                  <1>       ; 0 = The corresponding PIRQ is routed to one of the ISA-compatible
49165                                  <1>       ;     interrupts specified in bits[3:0].
49166                                  <1>       ; 1 = The PIRQ is not routed to the 8259.
49167                                  <1>       ; Note: If the PIRQ is intended to cause an interrupt to the ICH's
49168                                  <1>       ;      integrated I/O APIC, then this bit should be set to 0 and
49169                                  <1>       ;      the APIC_EN bit should be set to 1.
49170                                  <1>       ;      The IRQEN must be set to 0 and the PIRQ routed to
49171                                  <1>       ;      an 8259 interrupt via the IRQ Routing filed (bits[3:0).
49172                                  <1>       ;      The corresponding 8259 interrupt must be masked via the
49173                                  <1>       ;      appropriated bit in the 8259's OCW1 (Interrupt Mask)
49174                                  <1>       ;      register. The IOAPIC must then be enabled by setting
49175                                  <1>       ;      the APIC_EN bit in the GEN_CNTL register.
49176                                  <1>
49177                                  <1>       ;mov   eax, 0F861h  ; D31:F0
49178                                  <1>             ;AL=61h : PIRQ[B] Routing Control Reg, LPC interface
49179                                  <1>       ;;mov  dl,  [audio_intr]
49180                                  <1>       ;call  pciRegWrite8
49181                                  <1>       ;;mov  al, 0D0h ; General Control Register (GEN_CTL)
49182                                  <1>       ;;call pciRegRead32
49183                                  <1>       ;;or   edx, 100h ; Bit 8, APIC_EN (Enable I/O APIC)
49184                                  <1>       ;;;call      pciRegWrite32
49185                                  <1>       ;;and  edx, ~100h
49186                                  <1>       ;;call pciRegWrite32 ; ; Bit 8, APIC_EN (Disable I/O APIC)
49187                                  <1>       ;
49188                                  <1>
49189                                  <1>       ;mov   dx, 4D1h     ; 8259 ELCR2
49190                                  <1>       ;in    al, dx
49191                                  <1>       ;mov   ah, al
49192                                  <1>       ;;mov  dx, 4D0h     ; 8259 ELCR1
49193                                  <1>       ;dec   dl
49194                                  <1>       ;in    al, dx
49195                                  <1>       ;bts   ax, cx
49196                                  <1>       ;;mov  dx, 4D0h
49197                                  <1>       ;out   dx, al       ; set level-triggered mode
49198                                  <1>       ;mov   al, ah ; 29/05/2017
49199                                  <1>       ;;mov  dx, 4D1h
49200                                  <1>       ;inc   dl
49201                                  <1>       ;out   dx, al       ; set level-triggered mode
49202                                  <1>
49203                                  <1>       ;xor   eax, eax ; 0
49204                                  <1>
49205                                  <1>       ;retn
49206                                  <1>
49207                                  <1> ; CODE for PCI
49208                                  <1>
49209                                  <1> pciFindDevice:
49210                                  <1>       ; 03/04/2017 ('pci.asm', 20/03/2017)
49211                                  <1>       ;
49212                                  <1>       ; scan through PCI space looking for a device+vendor ID
49213                                  <1>       ;
49214                                  <1>       ; Entry: EAX=Device+Vendor ID
49215                                  <1>       ;
49216                                  <1>       ; Exit: EAX=PCI address if device found
49217                                  <1>       ;      EDX=Device+Vendor ID
49218                                  <1>       ;       CY clear if found, set if not found. EAX invalid if CY set.
49219                                  <1>       ;
49220                                  <1>       ; Destroys: ebx, esi, edi, cl
49221                                  <1>       ;
49222                                  <1>
49223                                  <1>       ;push  ecx
49224 00010FCF 50                      <1>       push   eax
49225                                  <1>       ;push  esi
49226                                  <1>       ;push  edi
49227                                  <1>
49228 00010FD0 89C6                    <1>         mov     esi, eax                ; save off vend+device ID
49229 00010FD2 BF00FFFF7F              <1>         mov     edi, (80000000h - 100h) ; start with bus 0, dev 0 func 0
49230                                  <1>
49231                                  <1> nextPCIdevice:
49232 00010FD7 81C700010000            <1>         add     edi, 100h
49233 00010FDD 81FF00F8FF80            <1>         cmp     edi, 80FFF800h         ; scanned all devices?
49234 00010FE3 F9                      <1>         stc
49235 00010FE4 740C                    <1>         je      short PCIScanExit       ; not found
49236                                  <1>
49237 00010FE6 89F8                    <1>         mov     eax, edi                ; read PCI registers
49238 00010FE8 E86F000000              <1>         call    pciRegRead32
49239 00010FED 39F2                    <1>         cmp     edx, esi                ; found device?
49240 00010FEF 75E6                    <1>         jne     short nextPCIdevice
49241 00010FF1 F8                      <1>         clc
49242                                  <1>
49243                                  <1> PCIScanExit:
49244 00010FF2 9C                      <1>       pushf
49245 00010FF3 B8FFFFFF7F              <1>       mov    eax, NOT_BIT31      ; 19/03/2017
49246 00010FF8 21F8                    <1>       and    eax, edi     ; return only bus/dev/fn #
49247 00010FFA 9D                      <1>       popf
49248                                  <1>
49249                                  <1>       ;pop   edi
49250                                  <1>       ;pop   esi
49251 00010FFB 5A                      <1>       pop    edx
49252                                  <1>       ;pop   ecx
49253 00010FFC C3                      <1>       retn
49254                                  <1>
49255                                  <1> pciRegRead:
```

```
49256                                  <1>       ; 03/04/2017 ('pci.asm', 20/03/2017)
49257                                  <1>       ;
49258                                  <1>       ; 8/16/32bit PCI reader
49259                                  <1>       ;
49260                                  <1>       ; Entry: EAX=PCI Bus/Device/fn/register number
49261                                  <1>       ;          BIT30 set if 32 bit access requested
49262                                  <1>       ;          BIT29 set if 16 bit access requested
49263                                  <1>       ;          otherwise defaults to 8 bit read
49264                                  <1>       ;
49265                                  <1>       ; Exit:  DL,DX,EDX register data depending on requested read size
49266                                  <1>       ;
49267                                  <1>       ; Note1: this routine is meant to be called via pciRegRead8,
49268                                  <1>       ;      pciRegread16 or pciRegRead32, listed below.
49269                                  <1>       ;
49270                                  <1>       ; Note2: don't attempt to read 32 bits of data from a non dword
49271                                  <1>       ;      aligned reg number.  Likewise, don't do 16 bit reads from
49272                                  <1>       ;      non word aligned reg #
49273                                  <1>
49274 00010FFD 53                      <1>       push   ebx
49275 00010FFE 51                      <1>       push   ecx
49276 00010FFF 89C3                    <1>         mov     ebx, eax        ; save eax, dh
49277 00011001 88F1                    <1>         mov     cl, dh
49278                                  <1>
49279 00011003 25FFFFFF3F              <1>         and     eax, NOT_PCI32_PCI16   ; clear out data size request
49280 00011008 0D00000080              <1>         or      eax, BIT31             ; make a PCI access request
49281 0001100D 24FC                    <1>         and     al, ~3 ; NOT 3         ; force index to be dword
49282                                  <1>
49283 0001100F 66BAF80C                <1>         mov     dx, PCI_INDEX_PORT
49284 00011013 EF                      <1>         out  dx, eax                   ; write PCI selector
49285                                  <1>
49286 00011014 66BAFC0C                <1>         mov     dx, PCI_DATA_PORT
49287 00011018 88D8                    <1>         mov     al, bl
49288 0001101A 2403                    <1>         and     al, 3                  ; figure out which port to
49289 0001101C 00C2                    <1>         add     dl, al                 ; read to
49290                                  <1>
49291 0001101E F7C3000000C0            <1>       test    ebx, PCI32+PCI16
49292 00011024 7507                    <1>         jnz     short _pregr0
49293 00011026 EC                      <1>       in     al, dx              ; return 8 bits of data
49294 00011027 88C2                    <1>         mov  dl, al
49295 00011029 88CE                    <1>       mov     dh, cl                   ; restore dh for 8 bit read
49296 0001102B EB12                    <1>       jmp    short _pregr2
49297                                  <1> _pregr0:
49298 0001102D F7C300000080            <1>       test    ebx, PCI32
49299 00011033 7507                    <1>         jnz  short _pregr1
49300 00011035 66ED                    <1>       in     ax, dx
49301 00011037 6689C2                  <1>         mov     dx, ax                 ; return 16 bits of data
49302 0001103A EB03                    <1>       jmp    short _pregr2
49303                                  <1> _pregr1:
49304 0001103C ED                      <1>       in     eax, dx                  ; return 32 bits of data
49305 0001103D 89C2                    <1>       mov    edx, eax
49306                                  <1> _pregr2:
49307 0001103F 89D8                    <1>       mov     eax, ebx          ; restore eax
49308 00011041 25FFFFFF3F              <1>         and     eax, NOT_PCI32_PCI16   ; clear out data size request
49309 00011046 59                      <1>       pop    ecx
49310 00011047 5B                      <1>       pop    ebx
49311 00011048 C3                      <1>       retn
49312                                  <1>
49313                                  <1> pciRegRead8:
49314 00011049 25FFFFFF3F              <1>         and     eax, NOT_PCI32_PCI16   ; set up 8 bit read size
49315 0001104E EBAD                    <1>         jmp     short pciRegRead; call generic PCI access
49316                                  <1>
49317                                  <1> pciRegRead16:
49318 00011050 25FFFFFF3F              <1>         and     eax, NOT_PCI32_PCI16   ; set up 16 bit read size
49319 00011055 0D00000040              <1>         or      eax, PCI16             ; call generic PCI access
49320 0001105A EBA1                    <1>         jmp     short pciRegRead
49321                                  <1>
49322                                  <1> pciRegRead32:
49323 0001105C 25FFFFFF3F              <1>         and     eax, NOT_PCI32_PCI16   ; set up 32 bit read size
49324 00011061 0D00000080              <1>         or      eax, PCI32             ; call generic PCI access
49325 00011066 EB95                    <1>         jmp     pciRegRead
49326                                  <1>
49327                                  <1> pciRegWrite:
49328                                  <1>       ; 03/04/2017 ('pci.asm', 29/11/2016)
49329                                  <1>       ;
49330                                  <1>       ; 8/16/32bit PCI writer
49331                                  <1>       ;
49332                                  <1>       ; Entry: EAX=PCI Bus/Device/fn/register number
49333                                  <1>       ;          BIT31 set if 32 bit access requested
49334                                  <1>       ;          BIT30 set if 16 bit access requested
49335                                  <1>       ;          otherwise defaults to 8bit read
49336                                  <1>       ;        DL/DX/EDX data to write depending on size
49337                                  <1>       ;
49338                                  <1>       ; Note1: this routine is meant to be called via pciRegWrite8,
49339                                  <1>       ;      pciRegWrite16 or pciRegWrite32 as detailed below.
49340                                  <1>       ;
49341                                  <1>       ; Note2: don't attempt to write 32bits of data from a non dword
49342                                  <1>       ;      aligned reg number.  Likewise, don't do 16 bit writes from
49343                                  <1>       ;      non word aligned reg #
49344                                  <1>
49345 00011068 53                      <1>       push   ebx
49346 00011069 51                      <1>       push   ecx
49347 0001106A 89C3                    <1>         mov     ebx, eax        ; save eax, edx
49348 0001106C 89D1                    <1>         mov     ecx, edx
49349 0001106E 25FFFFFF3F              <1>       and     eax, NOT_PCI32_PCI16     ; clear out data size request
49350 00011073 0D00000080              <1>         or      eax, BIT31             ; make a PCI access request
49351 00011078 24FC                    <1>         and     al, ~3 ; NOT 3         ; force index to be dword
49352                                  <1>
49353 0001107A 66BAF80C                <1>         mov     dx, PCI_INDEX_PORT
49354 0001107E EF                      <1>         out  dx, eax                   ; write PCI selector
49355                                  <1>
49356 0001107F 66BAFC0C                <1>         mov     dx, PCI_DATA_PORT
49357 00011083 88D8                    <1>         mov     al, bl
```

```
49358 00011085 2403              <1>         and     al, 3                   ; figure out which port to
49359 00011087 00C2              <1>         add     dl, al                  ; write to
49360                            <1>
49361 00011089 F7C3000000C0      <1>       test    ebx, PCI32+PCI16
49362 0001108F 7505              <1>         jnz     short _pregw0
49363 00011091 88C8              <1>       mov   al, cl                      ; put data into al
49364 00011093 EE                <1>       out   dx, al
49365 00011094 EB12              <1>       jmp   short _pregw2
49366                            <1> _pregw0:
49367 00011096 F7C300000080      <1>       test    ebx, PCI32
49368 0001109C 7507              <1>         jnz     short _pregw1
49369 0001109E 6689C8            <1>       mov   ax, cx              ; put data into ax
49370 000110A1 66EF              <1>       out   dx, ax
49371 000110A3 EB03              <1>       jmp   short _pregw2
49372                            <1> _pregw1:
49373 000110A5 89C8              <1>       mov   eax, ecx            ; put data into eax
49374 000110A7 EF                <1>       out   dx, eax
49375                            <1> _pregw2:
49376 000110A8 89D8              <1>         mov     eax, ebx        ; restore eax
49377 000110AA 25FFFFFF3F        <1>         and     eax, NOT_PCI32_PCI16   ; clear out data size request
49378 000110AF 89CA              <1>         mov     edx, ecx        ; restore dx
49379 000110B1 59                <1>       pop   ecx
49380 000110B2 5B                <1>       pop   ebx
49381 000110B3 C3                <1>       retn
49382                            <1>
49383                            <1> pciRegWrite8:
49384 000110B4 25FFFFFF3F        <1>         and     eax, NOT_PCI32_PCI16   ; set up 8 bit write size
49385 000110B9 EBAD              <1>         jmp  short pciRegWrite   ; call generic PCI access
49386                            <1>
49387                            <1> pciRegWrite16:
49388 000110BB 25FFFFFF3F        <1>         and     eax, NOT_PCI32_PCI16   ; set up 16 bit write size
49389 000110C0 0D00000040        <1>         or      eax, PCI16             ; call generic PCI access
49390 000110C5 EBA1              <1>         jmp  short pciRegWrite
49391                            <1>
49392                            <1> pciRegWrite32:
49393 000110C7 25FFFFFF3F        <1>         and     eax, NOT_PCI32_PCI16   ; set up 32 bit write size
49394 000110CC 0D00000080        <1>         or      eax, PCI32             ; call generic PCI access
49395 000110D1 EB95              <1>         jmp  pciRegWrite
49396                            <1>
49397                            <1> init_codec:
49398                            <1>       ; 05/06/2017
49399                            <1>       ; 28/05/2017 - Erdogan Tan (Ref: KolibriOS, vt823x.asm)
49400                            <1>       ;
49401 000110D3 A1[54610100]      <1>       mov   eax, [audio_dev_id]
49402 000110D8 B041              <1>       mov   al, VIA_ACLINK_CTRL
49403 000110DA E86AFFFFFF        <1>       call  pciRegRead8
49404                            <1>       ; ?
49405 000110DF B040              <1>       mov   al, VIA_ACLINK_STAT
49406 000110E1 E863FFFFFF        <1>       call  pciRegRead8
49407 000110E6 F6C201            <1>       test  dl, VIA_ACLINK_C00_READY
49408 000110E9 7508              <1>         jnz     short _codec_ready_1
49409 000110EB E80E000000        <1>       call  reset_codec
49410 000110F0 7306              <1>       jnc   short _codec_ready_2 ; eax = 1
49411 000110F2 C3                <1>       retn
49412                            <1> _codec_ready_1:
49413 000110F3 B801000000        <1>       mov   eax, 1
49414                            <1> _codec_ready_2:
49415 000110F8 E87A000000        <1>       call  codec_io_w16
49416                            <1> detect_codec:
49417 000110FD C3                <1>       retn
49418                            <1>
49419                            <1> reset_codec:
49420                            <1>       ; 16/04/2017
49421                            <1>       ; 23/03/2017
49422                            <1>       ; ('codec.asm')
49423                            <1>       ; 12/11/2016 - Erdogan Tan (Ref: KolibriOS, vt823x.asm)
49424 000110FE A1[54610100]      <1>       mov   eax, [audio_dev_id]
49425 00011103 B041              <1>       mov   al, VIA_ACLINK_CTRL
49426 00011105 B2E0              <1>             mov   dl, VIA_ACLINK_CTRL_ENABLE + VIA_ACLINK_CTRL_RESET + VIA_ACLINK_CTRL_SYNC
49427 00011107 E8A8FFFFFF        <1>       call  pciRegWrite8
49428                            <1>
49429 0001110C E83D000000        <1>       call  delay_100ms  ; wait 100 ms
49430                            <1> _rc_cold:
49431 00011111 E808000000        <1>         call    cold_reset
49432 00011116 7301              <1>         jnc     short _reset_codec_ok
49433                            <1>
49434                            <1>       ; 16/04/2017
49435                            <1>         ;xor     eax, eax         ; timeout error
49436                            <1>             ;stc
49437 00011118 C3                <1>       retn
49438                            <1>
49439                            <1> _reset_codec_ok:
49440 00011119 31C0              <1>         xor     eax, eax
49441                            <1>         ;mov al, VIA_ACLINK_C00_READY ; 1
49442 0001111B FEC0              <1>         inc  al
49443 0001111D C3                <1>       retn
49444                            <1>
49445                            <1> cold_reset:
49446                            <1>       ; 16/04/2017
49447                            <1>       ; 23/03/2017
49448                            <1>       ; ('codec.asm')
49449                            <1>       ; 12/11/2016 - Erdogan Tan (Ref: KolibriOS, vt823x.asm)
49450                            <1>       ;mov   eax, [audio_dev_id]
49451                            <1>       ;mov   al, VIA_ACLINK_CTRL
49452 0001111E 30D2              <1>       xor   dl, dl ; 0
49453 00011120 E88FFFFFFF        <1>       call  pciRegWrite8
49454                            <1>
49455 00011125 E824000000        <1>       call  delay_100ms  ; wait 100 ms
49456                            <1>
49457                            <1>       ;; ACLink on, deassert ACLink reset, VSR, SGD data out
49458                            <1>         ;; note - FM data out has trouble with non VRA codecs !!
49459                            <1>
```

```
                                <1>       ;mov   eax, [audio_dev_id]
                                <1>       ;mov   al, VIA_ACLINK_CTRL
0001112A B2CC                   <1>       mov    dl, VIA_ACLINK_CTRL_INIT
0001112C E883FFFFFF             <1>       call   pciRegWrite8
                                <1>
00011131 B910000000             <1>       mov    ecx, 16      ; total 2s
                                <1>
                                <1> _crst_wait:
                                <1>       ;mov   eax, [audio_dev_id]
00011136 B040                   <1>       mov    al, VIA_ACLINK_STAT
00011138 E80CFFFFFF             <1>       call   pciRegRead8
                                <1>
0001113D F6C201                 <1>         test    dl, VIA_ACLINK_C00_READY
00011140 750B                   <1>         jnz     short _crst_ok
                                <1>
00011142 51                     <1>       push   ecx
00011143 E806000000             <1>       call   delay_100ms
00011148 59                     <1>       pop    ecx
                                <1>
00011149 49                     <1>         dec     ecx
0001114A 75EA                   <1>         jnz     short _crst_wait
                                <1>
                                <1> _crst_fail:
0001114C F9                     <1>         stc
                                <1> _crst_ok:
0001114D C3                     <1>       retn
                                <1>
                                <1> delay_100ms:
                                <1>       ; 29/05/2017
                                <1>       ; 24/03/2017 ('codec.asm')
                                <1>       ; wait 100 ms
0001114E B990010000             <1>       mov    ecx, 400  ; 400*0.25ms
                                <1>
                                <1> _delay_x_ms:
00011153 E803000000             <1>       call   delay1_4ms
00011158 E2F9                   <1>         loop _delay_x_ms
0001115A C3                     <1>       retn
                                <1>
                                <1> ;       delay1_4ms - Delay for 1/4 millisecond.
                                <1> ;         1mS = 1000us
                                <1> ;       Entry:
                                <1> ;         None
                                <1> ;       Exit:
                                <1> ;       None
                                <1> ;
                                <1> ;       Modified:
                                <1> ;         None
                                <1> ;
                                <1>
                                <1>       ; 29/05/2017
                                <1>       ; 23/04/2017
                                <1>       ; 05/03/2017 (TRDOS 386)
                                <1>       ; ('UTILS.ASM')
                                <1> delay1_4ms:
0001115B 50                     <1>         push    eax
0001115C 51                     <1>         push    ecx
0001115D B110                   <1>         mov  cl, 16       ; close enough.
                                <1>
0001115F E461                   <1>       in    al, PORTB ; 61h
                                <1>
00011161 2410                   <1>       and   al, REFRESH_STATUS ; 10h
00011163 88C5                   <1>       mov   ch, al       ; Start toggle state
                                <1> _d4ms1:
00011165 E461                   <1>       in    al, PORTB    ; Read system control port
                                <1>
00011167 2410                   <1>       and   al, REFRESH_STATUS ; Refresh toggles 15.085 microseconds
00011169 38C5                   <1>       cmp   ch, al
0001116B 74F8                   <1>       je    short _d4ms1 ; Wait for state change
                                <1>
0001116D 88C5                   <1>       mov   ch, al       ; Update with new state
0001116F FEC9                   <1>       dec   cl
00011171 75F2                   <1>       jnz   short _d4ms1
                                <1>
00011173 F8                     <1>       clc   ; 29/05/2017
                                <1>
00011174 59                     <1>         pop     ecx
00011175 58                     <1>         pop     eax
00011176 C3                     <1>         retn
                                <1>
                                <1> ; 10/04/2017 (TRDOS 386)
                                <1> ; 12/11/2016
                                <1>
                                <1> codec_io_w16: ;w32
                                <1>       ; ('codec.asm')
00011177 668B15[52610100]       <1>         mov  dx, [audio_io_base]
0001117E 6681C28000             <1>         add     dx, VIA_REG_AC97
00011183 EF                     <1>       out   dx, eax
00011184 C3                     <1>         retn
                                <1>
                                <1> codec_io_r16: ;r32
                                <1>       ; ('codec.asm')
00011185 668B15[52610100]       <1>         mov     dx, [audio_io_base]
0001118C 6681C28000             <1>         add     dx, VIA_REG_AC97
00011191 ED                     <1>         in  eax, dx
00011192 C3                     <1>         retn
                                <1>
                                <1> ctrl_io_w8:
                                <1>       ; ('codec.asm')
00011193 660315[52610100]       <1>         add     dx, [audio_io_base]
0001119A EE                     <1>         out  dx, al
0001119B C3                     <1>         retn
                                <1>
                                <1> ctrl_io_r8:
```

```
49562                                 <1>       ; ('codec.asm')
49563 0001119C 660315[52610100]       <1>         add     dx, [audio_io_base]
49564 000111A3 EC                     <1>         in   al, dx
49565 000111A4 C3                     <1>         retn
49566                                 <1>
49567                                 <1> ctrl_io_w32:
49568                                 <1>       ; ('codec.asm')
49569 000111A5 660315[52610100]       <1>         add     dx, [audio_io_base]
49570 000111AC EF                     <1>         out  dx, eax
49571 000111AD C3                     <1>         retn
49572                                 <1>
49573                                 <1> ctrl_io_r32:
49574                                 <1>       ; ('codec.asm')
49575 000111AE 660315[52610100]       <1>         add  dx, [audio_io_base]
49576 000111B5 ED                     <1>       in     eax, dx
49577 000111B6 C3                     <1>         retn
49578                                 <1>
49579                                 <1> codec_read:
49580                                 <1>       ; 12/11/2016 - Erdogan Tan (Ref: KolibriOS, vt823x.asm)
49581                                 <1>         ; Use only primary codec.
49582                                 <1>         ; eax = register
49583 000111B7 C1E010                <1>         shl     eax, VIA_REG_AC97_CMD_SHIFT
49584 000111BA 0D00008002            <1>         or      eax, VIA_REG_AC97_PRIMARY_VALID + VIA_REG_AC97_READ
49585                                 <1>
49586 000111BF E8B3FFFFFF            <1>       call    codec_io_w16
49587                                 <1>
49588                                 <1>             ; codec_valid
49589 000111C4 E831000000            <1>       call   codec_check_ready
49590 000111C9 7301                  <1>         jnc  short _cr_ok
49591                                 <1>
49592 000111CB C3                    <1>       retn
49593                                 <1>
49594                                 <1> _cr_ok:
49595                                 <1>       ; wait 25 ms
49596 000111CC B950000000            <1>       mov    ecx, 80 ; (100*0.25 ms)
49597                                 <1> _cr_wloop:
49598 000111D1 E885FFFFFF            <1>       call   delay1_4ms
49599 000111D6 E2F9                  <1>       loop   _cr_wloop
49600                                 <1>
49601 000111D8 E8A8FFFFFF            <1>         call    codec_io_r16
49602 000111DD 25FFFF0000            <1>         and     eax, 0FFFFh
49603 000111E2 C3                    <1>         retn
49604                                 <1>
49605                                 <1> codec_write:
49606                                 <1>       ; 12/11/2016 - Erdogan Tan (Ref: KolibriOS, vt823x.asm)
49607                                 <1>         ; Use only primary codec.
49608                                 <1>
49609                                 <1>       ; eax = data (volume)
49610                                 <1>       ; edx = register (mixer register)
49611                                 <1>
49612 000111E3 C1E210                <1>       shl     edx, VIA_REG_AC97_CMD_SHIFT
49613                                 <1>
49614 000111E6 C1E000                <1>         shl     eax, VIA_REG_AC97_DATA_SHIFT ; shl eax, 0
49615 000111E9 09C2                  <1>         or      edx, eax
49616                                 <1>
49617 000111EB B800000000            <1>         mov     eax, VIA_REG_AC97_CODEC_ID_PRIMARY
49618 000111F0 C1E01E                <1>         shl     eax, VIA_REG_AC97_CODEC_ID_SHIFT
49619 000111F3 09D0                  <1>         or      eax, edx
49620                                 <1>
49621 000111F5 E87DFFFFFF            <1>         call    codec_io_w16
49622                                 <1>         ;mov    [codec.regs+esi], ax
49623                                 <1>
49624                                 <1>         ;call      codec_check_ready
49625                                 <1>             ;retn
49626                                 <1>       ;jmp   short _codec_check_ready
49627                                 <1>
49628                                 <1> codec_check_ready:
49629                                 <1>       ; 12/11/2016 - Erdogan Tan (Ref: KolibriOS, vt823x.asm)
49630                                 <1>
49631                                 <1> _codec_check_ready:
49632 000111FA B914000000            <1>       mov    ecx, 20      ; total 2s
49633                                 <1> _ccr_wait:
49634 000111FF 51                    <1>       push   ecx
49635                                 <1>
49636 00011200 E880FFFFFF            <1>         call    codec_io_r16
49637 00011205 A900000001            <1>         test    eax, VIA_REG_AC97_BUSY
49638 0001120A 740B                  <1>         jz      short _ccr_ok
49639                                 <1>
49640 0001120C E83DFFFFFF            <1>       call   delay_100ms
49641                                 <1>
49642 00011211 59                    <1>       pop    ecx
49643                                 <1>
49644 00011212 49                    <1>       dec     ecx
49645 00011213 75EA                  <1>         jnz     short _ccr_wait
49646                                 <1>
49647 00011215 F9                    <1>         stc
49648 00011216 C3                    <1>         retn
49649                                 <1>
49650                                 <1> _ccr_ok:
49651 00011217 59                    <1>       pop    ecx
49652 00011218 25FFFF0000            <1>       and     eax, 0FFFFh
49653 0001121D C3                    <1>         retn
49654                                 <1>
49655                                 <1> codec_config:
49656                                 <1>       ; 10/06/2017
49657                                 <1>       ; 29/05/2017
49658                                 <1>       ; 24/04/2017
49659                                 <1>       ; 21/04/2017
49660                                 <1>       ; 16/04/2017 (TRDOS 386 Kernel)
49661                                 <1>       ; 15/11/2016 ('codec.asm', 'player.com')
49662                                 <1>       ; 14/11/2016
49663                                 <1>       ; 12/11/2016 - Erdogan Tan
```

```
49664                                  <1>       ;          (Ref: KolibriOS, 'setup_codec', codec.inc)
49665                                  <1>
49666 0001121E B802020000              <1>       mov     eax, 0202h
49667 00011223 66A3[82610100]          <1>       mov    [audio_master_volume], ax
49668 00011229 66B81F1F                <1>       mov    ax, 1F1Fh ; 31,31
49669 0001122D BA02000000              <1>       mov    edx, CODEC_MASTER_VOL_REG ; 02h ; Line Out
49670 00011232 E8ACFFFFFF              <1>       call   codec_write
49671                                  <1>       ;jc    short cconfig_error
49672                                  <1>
49673                                  <1>       ;mov    eax, 0202h
49674 00011237 66B80202                <1>       mov     ax, 0202h
49675 0001123B BA18000000              <1>       mov    edx, CODEC_PCM_OUT_REG ; 18h ; Wave Output (Stereo)
49676 00011240 E89EFFFFFF              <1>       call   codec_write
49677                                  <1>       ;jc    short cconfig_error
49678                                  <1>
49679                                  <1>       ;mov    eax, 0202h
49680 00011245 66B80202                <1>       mov    ax, 0202h
49681 00011249 BA04000000              <1>       mov    edx, CODEC_AUX_VOL ; 04h ; CODEC_HP_VOL_REG ; HeadPhone
49682 0001124E E890FFFFFF              <1>       call   codec_write
49683                                  <1>       ;jc    short cconfig_error
49684                                  <1>
49685                                  <1>       ;mov    eax, 08h
49686                                  <1>         ;mov    ax,  08h
49687 00011253 66B80880                <1>       mov    ax, 8008h ; Mute
49688 00011257 BA0C000000              <1>       mov    edx, 0Ch  ; AC97_PHONE_VOL ; TAD Input (Mono)
49689 0001125C E882FFFFFF              <1>       call   codec_write
49690                                  <1>       ;jc    short cconfig_error
49691                                  <1>
49692                                  <1>       ;mov    eax, 0808h
49693 00011261 66B80808                <1>         mov     ax,  0808h
49694 00011265 BA10000000              <1>         mov  edx, CODEC_LINE_IN_VOL_REG ; 10h ; Line Input (Stereo)
49695 0001126A E874FFFFFF              <1>       call   codec_write
49696                                  <1>       ;jc    short cconfig_error
49697                                  <1>
49698                                  <1>       ;mov    eax, 0808h
49699 0001126F 66B80808                <1>       mov     ax,  0808h
49700 00011273 BA12000000              <1>         mov  edx, CODEC_CD_VOL_REG ; 12h ; CR Input (Stereo)
49701 00011278 E866FFFFFF              <1>       call   codec_write
49702                                  <1>       ;jc    short cconfig_error
49703                                  <1>
49704                                  <1>       ;mov    eax, 0808h
49705 0001127D 66B80808                <1>       mov     ax,  0808h
49706 00011281 BA16000000              <1>         mov  edx, CODEC_AUX_VOL_REG ; 16h ; Aux Input (Stereo)
49707                                  <1>       ;call  codec_write
49708                                  <1>       ;;jc   short cconfig_error
49709 00011286 E958FFFFFF              <1>       jmp    codec_write ; 10/06/2017
49710                                  <1>
49711                                  <1> ;     ; Extended Audio Status (2Ah)
49712                                  <1> ;     mov    eax, CODEC_EXT_AUDIO_CTRL_REG ; 2Ah
49713                                  <1> ;     call   codec_read
49714                                  <1> ;     and     eax, 0FFFFh - 2          ; clear DRA (BIT1)
49715                                  <1> ;     ;or     eax, 1                   ; set VRA (BIT0)
49716                                  <1> ;     or     eax, 5       ; VRA (BIT0) & S/PDIF (BIT2) ; 14/11/2016
49717                                  <1> ;     mov    edx, CODEC_EXT_AUDIO_CTRL_REG
49718                                  <1> ;     call   codec_write
49719                                  <1> ;     ;jc    short cconfig_error
49720                                  <1> ;
49721                                  <1> ;set_sample_rate:
49722                                  <1> ;     ;movzx eax, word [audio_freq]
49723                                  <1> ;     mov    ax, [audio_freq]
49724                                  <1> ;     mov    edx, CODEC_PCM_FRONT_DACRATE_REG ; 2Ch ; PCM Front DAC Rate
49725                                  <1> ;     ;call  codec_write
49726                                  <1> ;     ;retn
49727                                  <1> ;     jmp    codec_write
49728                                  <1>
49729                                  <1> ;cconfig_error:
49730                                  <1> ;     retn
49731                                  <1>
49732                                  <1> vt8233_int_handler:
49733                                  <1>       ; Interrupt Handler for VIA VT8237R Audio Controller
49734                                  <1>       ; Note: called by 'dev_irq_service'
49735                                  <1>       ; 13/06/2017
49736                                  <1>       ; 21/04/2017 (TRDOS 386 kernel, 'audio.s')
49737                                  <1>       ; 24/03/2017 - 'PLAYER.COM' ('player.asm')
49738                                  <1>
49739 0001128B 50                      <1>       push   eax
49740 0001128C 52                      <1>       push   edx
49741 0001128D 51                      <1>       push   ecx
49742                                  <1>       ;push  ebx
49743 0001128E 56                      <1>       push   esi
49744 0001128F 57                      <1>       push   edi
49745                                  <1>
49746 00011290 803D[50610100]01        <1>       cmp    byte [audio_busy], 1
49747 00011297 733A                    <1>       jnb    short _ih1  ; busy !
49748                                  <1>
49749 00011299 C605[50610100]01        <1>       mov    byte [audio_busy], 1
49750                                  <1>
49751 000112A0 C605[81610100]00        <1>       mov    byte [audio_flag_eol], 0
49752                                  <1>
49753 000112A7 66BA0000                <1>         mov     dx, VIADEV_PLAYBACK + VIA_REG_OFFSET_STATUS
49754 000112AB E8ECFEFFFF              <1>         call    ctrl_io_r8
49755                                  <1>
49756 000112B0 A880                    <1>       test    al, VIA_REG_STAT_ACTIVE
49757 000112B2 741F                    <1>         jz      short _ih1
49758                                  <1>
49759 000112B4 2407                    <1>         and     al, VIA_REG_STAT_EOL + VIA_REG_STAT_FLAG + VIA_REG_STAT_STOPPED
49760 000112B6 A2[81610100]            <1>       mov    [audio_flag_eol], al
49761 000112BB 7416                    <1>         jz   short _ih1
49762                                  <1>
49763 000112BD E842000000              <1>       call   vt8233_tuneLoop
49764                                  <1>
49765                                  <1>       ; 13/06/2017
```

```
49766 000112C2 30C0                <1>       xor    al, al ; 0
49767 000112C4 3805[74610100]      <1>       cmp    [audio_flag], al ; 0
49768 000112CA 7702                <1>       ja     short _ih0
49769 000112CC FEC0                <1>       inc    al ; 1
49770                              <1> _ih0:
49771 000112CE A2[74610100]        <1>       mov    [audio_flag], al
49772                              <1> _ih1:
49773 000112D3 A0[81610100]        <1>         mov     al, [audio_flag_eol]   ;; ack ;;
49774 000112D8 66BA0000            <1>         mov     dx, VIADEV_PLAYBACK + VIA_REG_OFFSET_STATUS
49775 000112DC E8B2FEFFFF          <1>         call    ctrl_io_w8
49776                              <1>
49777 000112E1 F605[81610100]01    <1>       test   byte [audio_flag_eol], VIA_REG_STAT_FLAG
49778 000112E8 740D                <1>       jz     short _ih2
49779                              <1>
49780 000112EA B023                <1>       mov    al, VIA_REG_CTRL_INT
49781 000112EC 0C80                <1>              or     al, VIA_REG_CTRL_START
49782 000112EE 66BA0100            <1>              mov     dx, VIADEV_PLAYBACK + VIA_REG_OFFSET_CONTROL
49783 000112F2 E89CFEFFFF          <1>         call    ctrl_io_w8
49784                              <1>
49785                              <1>       ;call  codec_check_ready
49786                              <1> _ih2:
49787                              <1>       ; 28/05/2017
49788 000112F7 C605[50610100]00    <1>       mov    byte [audio_busy], 0
49789                              <1>       ;
49790 000112FE 5F                  <1>       pop    edi
49791 000112FF 5E                  <1>       pop    esi
49792                              <1>       ;pop   ebx
49793 00011300 59                  <1>       pop    ecx
49794 00011301 5A                  <1>       pop    edx
49795 00011302 58                  <1>       pop    eax
49796                              <1>
49797 00011303 C3                  <1>       retn
49798                              <1>
49799                              <1> vt8233_tuneLoop:
49800                              <1>       ; 24/06/2017
49801                              <1>       ; 13/06/2017
49802                              <1>       ; 28/05/2017
49803                              <1>       ; 24/04/2017
49804                              <1>       ; 21/04/2017 (TRDOS 386 kernel, 'audio.s')
49805                              <1>       ; 24/03/2017 - 'PLAYER.COM' ('via_wav.asm' - 29/11/2016)
49806                              <1>       ; 08/12/2016
49807                              <1>       ; 28/11/2016 - Erdogan Tan
49808                              <1>
49809 00011304 803D[80610100]01    <1>       cmp    byte [audio_play_cmd], 1
49810                              <1>       ;jb    short _tlp2 ; stop command !
49811 0001130B 7225                <1>       jb     short channel_reset ; 24/06/2017
49812                              <1>
49813 0001130D 8B3D[6C610100]      <1>       mov    edi, [audio_dma_buff]
49814 00011313 8B0D[70610100]      <1>       mov    ecx, [audio_dmabuff_size]
49815 00011319 D1E9                <1>       shr    ecx, 1 ; dma buff size / 2 = half buffer size
49816                              <1>
49817                              <1>       ;mov   byte [audio_flag], 0 ; Buffer 1 (next buffer)
49818                              <1>
49819                              <1>       ;test  byte [audio_flag_eol], VIA_REG_STAT_EOL
49820                              <1>       ;jz    short _tlp1 ; FLAG
49821                              <1>       ;; EOL
49822                              <1>       ;inc   byte [audio_flag] ; 1 ; Buffer 2 (next buffer)
49823                              <1>       ;add   edi, ecx
49824                              <1>
49825                              <1>       ; 13/06/2017
49826 0001131B F605[74610100]01    <1>       test   byte [audio_flag], 1  ; Current flag value
49827 00011322 7402                <1>       jz     short _tlp1 ; EOL (Half Buffer 1 must be filled)
49828                              <1>       ; FLAG (Half Buffer 2 must be filled)
49829 00011324 01CF                <1>       add    edi, ecx
49830                              <1> _tlp1:
49831 00011326 8B35[64610100]      <1>       mov    esi, [audio_p_buffer] ; phy addr of audio buff
49832                              <1>       ;rep   movsb
49833 0001132C C1E902              <1>       shr    ecx, 2 ; half buff size / 4
49834 0001132F F3A5                <1>       rep    movsd
49835                              <1>
49836 00011331 C3                  <1>       retn
49837                              <1>
49838                              <1> channel_reset:
49839                              <1>       ; 24/06/2017
49840                              <1>       ; 29/05/2017
49841                              <1>       ; 23/03/2017
49842                              <1>       ; 14/11/2016 - Erdogan Tan
49843                              <1>       ; 12/11/2016 - Erdogan Tan (Ref: KolibriOS, vt823x.asm)
49844 00011332 BA01000000          <1>         mov  edx, VIA_REG_OFFSET_CONTROL
49845                              <1>         ;mov eax, VIA_REG_CTRL_PAUSE + VIA_REG_CTRL_TERMINATE + VIA_REG_CTRL_RESET
49846 00011337 B848000000          <1>         mov  eax, VIA_REG_CTRL_PAUSE + VIA_REG_CTRL_TERMINATE ; 24/06/2017
49847 0001133C E852FEFFFF          <1>       call    ctrl_io_w8
49848                              <1>
49849                              <1>         ;mov edx, VIA_REG_OFFSET_CONTROL
49850                              <1>         ;call   ctrl_io_r8
49851                              <1>
49852                              <1>       ; wait for 50 ms
49853 00011341 B9A0000000          <1>       mov    ecx, 160 ; (200*0.25 ms) ; 29/05/2017
49854                              <1> _ch_rst_wait:
49855 00011346 E810FEFFFF          <1>       call   delay1_4ms
49856 0001134B 49                  <1>       dec    ecx
49857 0001134C 75F8                <1>       jnz    short _ch_rst_wait
49858                              <1>
49859                              <1>         ; disable interrupts
49860 0001134E BA01000000          <1>         mov  edx, VIA_REG_OFFSET_CONTROL
49861 00011353 31C0                <1>         xor     eax, eax
49862 00011355 E839FEFFFF          <1>         call    ctrl_io_w8
49863                              <1>
49864                              <1>         ; clear interrupts
49865 0001135A BA00000000          <1>         mov  edx, VIA_REG_OFFSET_STATUS
49866 0001135F B803000000          <1>       mov    eax, 3
49867 00011364 E82AFEFFFF          <1>         call ctrl_io_w8
```

```
49868                                  <1>
49869                                  <1>       ;mov   edx, VIA_REG_OFFSET_CURR_PTR
49870                                  <1>       ;xor   eax, eax
49871                                  <1>       ;call  ctrl_io_w32
49872                                  <1>
49873 00011369 C3                      <1>         retn
49874                                  <1>
49875                                  <1> vt8233_stop: ; 22/04/2017
49876 0001136A C605[80610100]00        <1>       mov    byte [audio_play_cmd], 0 ; stop !
49877                                  <1> _tlp2:
49878                                  <1>       ; 24/06/2017
49879                                  <1>         ; finished with song, stop everything
49880                                  <1>       ;mov   al, VIA_REG_CTRL_INT
49881                                  <1>         ;or  al, VIA_REG_CTRL_TERMINATE
49882                                  <1>       ;mov   dx, VIADEV_PLAYBACK + VIA_REG_OFFSET_CONTROL
49883                                  <1>         ;call      ctrl_io_w8
49884                                  <1>
49885                                  <1>         ;call      channel_reset
49886                                  <1>       ;retn
49887 00011371 EBBF                    <1>       jmp    short channel_reset
49888                                  <1>
49889                                  <1> set_vt8233_bdl: ; Set VT8237R Buffer Descriptor List
49890                                  <1>       ; 28/05/2017
49891                                  <1>       ; 21/04/2017 (TRDOS 386 kernel, 'audio.s')
49892                                  <1>       ; 24/03/2017 - 'PLAYER.COM' ('via_wav.asm' - 29/11/2016)
49893                                  <1>
49894                                  <1>       ; eax = dma buffer address = [audio_DMA_buff]
49895                                  <1>       ; ecx = dma buffer buffer size = [audio_dmabuff_size]
49896                                  <1>
49897 00011373 D1E9                    <1>       shr    ecx, 1 ; dma half buffer size
49898 00011375 89CE                    <1>       mov    esi, ecx
49899                                  <1>
49900 00011377 BF[88610100]            <1>         mov     edi, audio_bdl_buff    ; get BDL address
49901 0001137C B910000000              <1>         mov     ecx, 32 / 2            ; make 32 entries in BDL
49902                                  <1>
49903 00011381 EB05                    <1>       jmp    short s_vt8233_bdl1
49904                                  <1>
49905                                  <1> s_vt8233_bdl0:
49906                                  <1>       ; set buffer descriptor 0 to start of data file in memory
49907                                  <1>
49908 00011383 A1[6C610100]            <1>       mov    eax, [audio_dma_buff]     ; Physical address of DMA buffer
49909                                  <1>
49910                                  <1> s_vt8233_bdl1:
49911 00011388 AB                      <1>       stosd                    ; store dmabuffer1 address
49912                                  <1>
49913 00011389 89C2                    <1>       mov    edx, eax
49914                                  <1>
49915                                  <1> ; VIA VT8235.PDF: (Page 110) (Erdogan Tan, 29/11/2016)
49916                                  <1>       ;
49917                                  <1>       ;      Audio SGD Table Format
49918                                  <1>       ;      --------------------------------
49919                                  <1>       ;      63   62    61-56    55-32  31-0
49920                                  <1>       ;      --   --   --------  -----  ----
49921                                  <1>       ;      EOL FLAG -reserved- Base   Base
49922                                  <1>       ;                        Count  Address
49923                                  <1>       ;                         [23:0] [31:0]
49924                                  <1>       ;      EOL: End Of Link.
49925                                  <1>       ;           1 indicates this block is the last of the link.
49926                                  <1>       ;           If the channel “Interrupt on EOL” bit is set, then
49927                                  <1>       ;           an interrupt is generated at the end of the transfer.
49928                                  <1>       ;
49929                                  <1>       ;      FLAG: Block Flag. If set, transfer pauses at the end of this
49930                                  <1>       ;            block. If the channel “Interrupt on FLAG” bit is set,
49931                                  <1>       ;            then an interrupt is generated at the end of this block.
49932                                  <1>
49933 0001138B 89F0                    <1>       mov    eax, esi ; DMA half buffer size
49934 0001138D 01C2                    <1>       add    edx, eax
49935 0001138F 0D00000040              <1>       or     eax, FLAG
49936                                  <1>       ;or    eax, EOL
49937 00011394 AB                      <1>       stosd
49938                                  <1>
49939                                  <1> ; 2nd buffer:
49940                                  <1>
49941 00011395 89D0                    <1>         mov  eax, edx ; Physical address of the 2nd half of DMA buffer
49942 00011397 AB                      <1>       stosd         ; store dmabuffer2 address
49943                                  <1>
49944                                  <1> ; set length to [audio_dmabuff_size]/2
49945                                  <1> ; Set control (bits 31:16) to BUP, bits 15:0=number of samples
49946                                  <1> ;
49947 00011398 89F0                    <1>       mov    eax, esi ; DMA half buffer size
49948 0001139A 0D00000080              <1>       or     eax, EOL
49949                                  <1>       ;or    eax, FLAG
49950 0001139F AB                      <1>       stosd
49951                                  <1>
49952 000113A0 E2E1                    <1>       loop    s_vt8233_bdl0
49953                                  <1>
49954 000113A2 C3                      <1>       retn
49955                                  <1>
49956                                  <1> vt8233_start_play:
49957                                  <1>       ; start to play audio data via VT8233 audio controller
49958                                  <1>       ; 13/06/2017
49959                                  <1>       ; 10/06/2017
49960                                  <1>       ; 24/04/2017
49961                                  <1>       ; 21/04/2017 (TRDOS 386 kernel, 'audio.s')
49962                                  <1>       ; 24/03/2017 - 'PLAYER.COM' ('via_wav.asm' - 29/11/2016)
49963                                  <1>       ; write buffer descriptor list address
49964                                  <1>       ;
49965                                  <1>
49966                                  <1>       ; Extended Audio Status (2Ah)
49967 000113A3 B82A000000              <1>       mov    eax, CODEC_EXT_AUDIO_CTRL_REG ; 2Ah
49968 000113A8 E80AFEFFFF              <1>       call   codec_read
49969 000113AD 25FDFF0000              <1>       and     eax, 0FFFFh - 2          ; clear DRA (BIT1)
```

```
49970                                  <1>       ;or     eax, 1                  ; set VRA (BIT0)
49971 000113B2 83C805                  <1>       or      eax, 5        ; VRA (BIT0) & S/PDIF (BIT2) ; 14/11/2016
49972 000113B5 BA2A000000              <1>       mov     edx, CODEC_EXT_AUDIO_CTRL_REG
49973 000113BA E824FEFFFF              <1>       call    codec_write
49974                                  <1>       ;jc     short cconfig_error
49975                                  <1>
49976                                  <1> set_sample_rate:
49977                                  <1>       ;movzx eax, word [audio_freq]
49978 000113BF 66A1[7E610100]          <1>       mov     ax, [audio_freq]
49979 000113C5 BA2C000000              <1>       mov     edx, CODEC_PCM_FRONT_DACRATE_REG ; 2Ch ; PCM Front DAC Rate
49980 000113CA E814FEFFFF              <1>       call    codec_write
49981                                  <1>
49982 000113CF B8[88610100]            <1>         mov  eax, audio_bdl_buff
49983                                  <1>
49984                                  <1>       ; 12/11/2016 - Erdogan Tan
49985                                  <1>       ; (Ref: KolibriOS, vt823x.asm, 'create_primary_buff')
49986 000113D4 BA04000000              <1>       mov     edx, VIADEV_PLAYBACK + VIA_REG_OFFSET_TABLE_PTR
49987 000113D9 E8C7FDFFFF              <1>         call ctrl_io_w32
49988                                  <1>
49989                                  <1>       ;call  codec_check_ready
49990                                  <1>
49991 000113DE 66BA0200                <1>       mov     dx, VIADEV_PLAYBACK + VIA_REG_OFS_PLAYBACK_VOLUME_L
49992                                  <1>         ;mov eax, 2 ; 31
49993 000113E2 B01F                    <1>       mov     al, 31
49994 000113E4 2A05[82610100]          <1>         sub  al, [audio_master_volume_l]
49995 000113EA E8A4FDFFFF              <1>       call    ctrl_io_w8
49996                                  <1>
49997                                  <1>       ;call  codec_check_ready
49998                                  <1>
49999 000113EF 66BA0300                <1>         mov     dx, VIADEV_PLAYBACK + VIA_REG_OFS_PLAYBACK_VOLUME_R
50000                                  <1>         ;mov ax, 2  ; 31
50001 000113F3 B01F                    <1>       mov     al, 31
50002 000113F5 2A05[83610100]          <1>         sub  al, [audio_master_volume_r]
50003 000113FB E893FDFFFF              <1>       call    ctrl_io_w8
50004                                  <1>
50005                                  <1>       ;call  codec_check_ready
50006                                  <1> ;
50007                                  <1> ;
50008                                  <1> ; All set.  Let's play some music.
50009                                  <1> ;
50010                                  <1> ;
50011                                  <1>             ;mov    dx, VIADEV_PLAYBACK + VIA_REG_OFFSET_STOP_IDX
50012                                  <1>         ;mov    ax, VIA8233_REG_TYPE_16BIT or VIA8233_REG_TYPE_STEREO or 0xfffff or 0xff000000
50013                                  <1>         ;call   ctrl_io_w32
50014                                  <1>
50015                                  <1>       ;call  codec_check_ready
50016                                  <1>
50017                                  <1>       ; 08/12/2016
50018                                  <1>       ; 07/10/2016
50019                                  <1>         ;mov    al, 1
50020 00011400 B01F                    <1>         mov  al, 31
50021 00011402 E815000000              <1>       call    set_VT8233_LastValidIndex
50022                                  <1>
50023 00011407 C605[80610100]01        <1>       mov     byte [audio_play_cmd], 1 ; play command (do not stop) !
50024                                  <1>
50025                                  <1> vt8233_play: ; continue to play
50026                                  <1>       ; 22/04/2017
50027 0001140E B023                    <1>         mov  al, VIA_REG_CTRL_INT
50028 00011410 0C80                    <1>             or     al, VIA_REG_CTRL_START
50029                                  <1>         ;mov al, VIA_REG_CTRL_AUTOSTART + VIA_REG_CTRL_START
50030                                  <1>       ;mov   al, VIA_REG_CTRL_AUTOSTART + VIA_REG_CTRL_START + VIA_REG_CTRL_INT_FLAG
50031 00011412 66BA0100                <1>       mov     dx, VIADEV_PLAYBACK + VIA_REG_OFFSET_CONTROL
50032 00011416 E878FDFFFF              <1>         call    ctrl_io_w8
50033                                  <1>       ;call  codec_check_ready
50034                                  <1>       ;retn
50035                                  <1>       ;jmp   codec_check_ready
50036 0001141B C3                      <1>       retn
50037                                  <1>
50038                                  <1> ;input AL = index # to stop on
50039                                  <1> set_VT8233_LastValidIndex:
50040                                  <1>       ; 10/06/2017
50041                                  <1>       ; 21/04/2017 (TRDOS 386 kernel, 'audio.s')
50042                                  <1>       ; 24/03/2017 - 'PLAYER.COM' ('via_wav.asm' - 29/11/2016)
50043                                  <1>       ; 19/11/2016
50044                                  <1>       ; 14/11/2016 - Erdogan Tan (Ref: VIA VT8235.PDF, Page 110)
50045                                  <1>       ; 12/11/2016 - Erdogan Tan
50046                                  <1>       ; (Ref: KolibriOS, vt823x.asm, 'create_primary_buff')
50047                                  <1>       ;push  edx
50048 0001141C 6650                    <1>       push   ax
50049                                  <1>       ;push  ecx
50050 0001141E 0FB705[7E610100]        <1>       movzx  eax, word [audio_freq] ; Hertz
50051 00011425 BA00001000              <1>       mov    edx, 100000h ; 2^20 = 1048576
50052 0001142A F7E2                    <1>       mul    edx
50053 0001142C B980BB0000              <1>       mov    ecx, 48000
50054 00011431 F7F1                    <1>       div    ecx
50055                                  <1>       ;and   eax, 0FFFFFh
50056                                  <1>       ;pop   ecx
50057 00011433 665A                    <1>       pop    dx
50058 00011435 C1E218                  <1>       shl    edx, 24  ; STOP Index Setting: Bit 24 to 31
50059 00011438 09D0                    <1>       or     eax, edx
50060                                  <1>       ; 19/11/2016
50061 0001143A 803D[7C610100]10        <1>       cmp    byte [audio_bps], 16
50062 00011441 7505                    <1>       jne    short sLVI_1
50063 00011443 0D00002000              <1>       or     eax, VIA8233_REG_TYPE_16BIT
50064                                  <1> sLVI_1:
50065 00011448 803D[7D610100]02        <1>       cmp    byte [audio_stmo], 2
50066 0001144F 7505                    <1>       jne    short sLVI_2
50067 00011451 0D00001000              <1>       or     eax, VIA8233_REG_TYPE_STEREO
50068                                  <1> sLVI_2:
50069 00011456 BA08000000              <1>       mov     edx, VIADEV_PLAYBACK + VIA_REG_OFFSET_STOP_IDX
50070 0001145B E845FDFFFF              <1>         call    ctrl_io_w32
50071                                  <1>       ;call  codec_check_ready
```

```
50072                                  <1>       ;pop   edx
50073 00011460 C3                      <1>       retn
50074                                  <1>
50075                                  <1> vt8233_pause: ; pause
50076                                  <1>       ; 10/06/2017
50077                                  <1>       ; 22/04/2017
50078 00011461 B023                    <1>       mov     al, VIA_REG_CTRL_INT
50079 00011463 0C08                    <1>         or      al, VIA_REG_CTRL_PAUSE
50080 00011465 66BA0100                <1>       mov     dx, VIADEV_PLAYBACK + VIA_REG_OFFSET_CONTROL
50081 00011469 E825FDFFFF              <1>         call    ctrl_io_w8
50082                                  <1>       ;call  codec_check_ready
50083                                  <1>       ;retn
50084                                  <1>       ;jmp   codec_check_ready
50085 0001146E C3                      <1>       retn
50086                                  <1>
50087                                  <1> vt8233_reset:
50088                                  <1>       ; 22/04/2017
50089                                  <1>       ; reset VT8237R (vt8233) Audio Controller
50090                                  <1>       ;cmp   byte [audio_play_cmd], 1
50091                                  <1>       ;jna   short vt8233_rst_0
50092 0001146F C605[80610100]00        <1>       mov    byte [audio_play_cmd], 0 ; stop !
50093                                  <1> vt8233_rst_0:
50094 00011476 E883FCFFFF              <1>       call   reset_codec
50095 0001147B 720A                    <1>       jc     short vt8233_rst_1 ; codec error !
50096                                  <1>       ; eax = 1
50097 0001147D E8F5FCFFFF              <1>       call   codec_io_w16 ; w32
50098 00011482 E8ABFEFFFF              <1>       call   channel_reset
50099                                  <1> vt8233_rst_1:
50100 00011487 C3                      <1>       retn
50101                                  <1>
50102                                  <1> vt8233_volume:
50103                                  <1>       ; set VT8237R (vt8233) sound volume level
50104                                  <1>       ; 24/04/2017
50105                                  <1>       ; 22/04/2017
50106                                  <1>       ; bl = component (0 = master/playback/lineout volume)
50107                                  <1>       ; cl = left channel volume level (0 to 31)
50108                                  <1>       ; ch = right channel volume level (0 to 31)
50109                                  <1>
50110 00011488 08DB                    <1>       or     bl, bl
50111 0001148A 7520                    <1>       jnz    short vt8233_vol_1 ; temporary !
50112 0001148C 66B81F1F                <1>       mov    ax, 1F1Fh ; 31,31
50113 00011490 38C1                    <1>       cmp    cl, al
50114 00011492 7718                    <1>       ja     short vt8233_vol_1 ; temporary !
50115 00011494 38E5                    <1>       cmp    ch, ah
50116 00011496 7714                    <1>       ja     short vt8233_vol_1 ; temporary !
50117 00011498 66890D[82610100]        <1>       mov    [audio_master_volume], cx
50118 0001149F 6629C8                  <1>       sub    ax, cx
50119 000114A2 BA02000000              <1>       mov    edx, CODEC_MASTER_VOL_REG ; 02h ; Line Out
50120 000114A7 E837FDFFFF              <1>       call   codec_write
50121                                  <1> vt8233_vol_1:
50122 000114AC C3                      <1>       retn
50123                                  <1>
50124                                  <1> ; CODE for SOUND BLASTER 16
50125                                  <1>
50126                                  <1> DetectSB:
50127                                  <1>       ; 24/04/2017
50128                                  <1>       ;pushad
50129                                  <1> ScanPort:
50130 000114AD 66BB1002                <1>       mov     bx, 210h    ; start scanning ports
50131                                  <1>                           ; 210h, 220h, .. 260h
50132                                  <1> ResetDSP:
50133 000114B1 6689DA                  <1>       mov     dx, bx            ; try to reset the DSP.
50134 000114B4 6683C206                <1>       add     dx, 06h
50135 000114B8 B001                    <1>       mov    al, 1
50136 000114BA EE                      <1>       out    dx, al
50137                                  <1>
50138 000114BB EC                      <1>       in     al, dx
50139 000114BC EC                      <1>       in     al, dx
50140 000114BD EC                      <1>       in     al, dx
50141 000114BE EC                      <1>       in     al, dx
50142                                  <1>
50143 000114BF 30C0                    <1>       xor     al, al
50144 000114C1 EE                      <1>       out    dx, al
50145                                  <1>
50146 000114C2 6683C208                <1>       add     dx, 08h
50147 000114C6 66B96400                <1>       mov    cx, 100
50148                                  <1> WaitID:
50149 000114CA EC                      <1>       in     al, dx
50150 000114CB 08C0                    <1>       or      al, al
50151 000114CD 7804                    <1>       js      short GetID
50152 000114CF E2F9                    <1>       loop    WaitID
50153 000114D1 EB0F                    <1>       jmp     short NextPort
50154                                  <1> GetID:
50155 000114D3 6683EA04                <1>       sub     dx, 04h
50156 000114D7 EC                      <1>       in     al, dx
50157 000114D8 3CAA                    <1>       cmp     al, 0AAh
50158 000114DA 7413                    <1>       je      short Found
50159 000114DC 6683C204                <1>       add     dx, 04h
50160 000114E0 E2E8                    <1>       loop    WaitID
50161                                  <1> NextPort:
50162 000114E2 6683C310                <1>       add     bx, 10h           ; if not response,
50163 000114E6 6681FB6002              <1>       cmp     bx, 260h    ; try the next port.
50164 000114EB 76C4                    <1>       jbe     short ResetDSP
50165 000114ED F9                      <1>       stc
50166 000114EE C3                      <1>       retn
50167                                  <1> Found:
50168 000114EF 66891D[52610100]        <1>       mov     [audio_io_base], bx      ; SB Port Address Found!
50169                                  <1> ScanIRQ:
50170                                  <1> SetIrqs:
50171 000114F6 28C0                    <1>       sub    al, al ; 0
50172 000114F8 A2[4A610100]            <1>       mov    [IRQnum], al ; reset
50173 000114FD A2[4F610100]            <1>       mov    [audio_intr], al ; reset
```

```
50174                                  <1>
50175                                  <1>       ; ah > 0 -> set IRQ vector
50176                                  <1>       ; al = IRQ number
50177                                  <1>       ;mov   ax, 103h ; IRQ 3
50178                                  <1>       ;call  set_hardware_int_vector
50179                                  <1>       ;mov   ax, 104h ; IRQ 4
50180                                  <1>       ;call  set_hardware_int_vector
50181 00011502 66B80501                <1>       mov    ax, 105h ; IRQ 5
50182 00011506 E8E1E1FFFF              <1>       call   set_hardware_int_vector
50183 0001150B 66B80701                <1>       mov    ax, 107h ; IRQ 7
50184 0001150F E8D8E1FFFF              <1>       call   set_hardware_int_vector
50185                                  <1>
50186 00011514 668B15[52610100]        <1>       mov     dx, [audio_io_base] ; tells to the SB to
50187 0001151B 6683C20C                <1>       add     dx, 0Ch               ; generate a IRQ!
50188                                  <1> WaitSb:
50189 0001151F EC                      <1>       in     al, dx
50190 00011520 08C0                    <1>       or      al, al
50191 00011522 78FB                    <1>       js      short WaitSb
50192 00011524 B0F2                    <1>       mov     al, 0F2h
50193 00011526 EE                      <1>       out    dx, al
50194                                  <1>
50195 00011527 31C9                    <1>       xor     ecx, ecx    ; wait until IRQ level
50196                                  <1> WaitIRQ:
50197 00011529 A0[4A610100]            <1>       mov    al, [IRQnum]
50198 0001152E 3C00                    <1>       cmp     al, 0 ; is changed or timeout.
50199 00011530 7706                    <1>       ja     short IrqOk
50200 00011532 6649                    <1>       dec    cx
50201 00011534 75F3                    <1>       jnz    short WaitIRQ
50202 00011536 EB15                    <1>       jmp    short RestoreIrqs
50203                                  <1> IrqOk:
50204 00011538 A2[4F610100]            <1>       mov    [audio_intr], al ; set
50205 0001153D 668B15[52610100]        <1>       mov     dx, [audio_io_base]
50206 00011544 6683C20E                <1>       add     dx, 0Eh
50207 00011548 EC                      <1>       in     al, dx ; SB acknowledge.
50208 00011549 B020                    <1>       mov    al, 20h
50209 0001154B E620                    <1>       out    20h, al      ; Hardware acknowledge.
50210                                  <1>
50211                                  <1> RestoreIrqs:
50212                                  <1>       ; ah = 0 -> reset IRQ vector
50213                                  <1>       ; al = IRQ number
50214                                  <1>       ;mov   ax, 3 ; IRQ 3
50215                                  <1>       ;call  set_hardware_int_vector
50216                                  <1>       ;mov   ax, 4 ; IRQ 4
50217                                  <1>       ;call  set_hardware_int_vector
50218 0001154D 66B80500                <1>       mov    ax, 5 ; IRQ 5
50219 00011551 E896E1FFFF              <1>       call   set_hardware_int_vector
50220 00011556 66B80700                <1>       mov    ax, 7 ; IRQ 7
50221 0001155A E88DE1FFFF              <1>       call   set_hardware_int_vector
50222                                  <1>
50223 0001155F 31D2                    <1>       xor    edx, edx
50224 00011561 8915[54610100]          <1>       mov    [audio_dev_id], edx ; 0
50225 00011567 8915[58610100]          <1>       mov    [audio_vendor], edx ; 0
50226 0001156D 8915[5C610100]          <1>       mov    [audio_stats_cmd], edx ; 0
50227                                  <1>
50228                                  <1>       ;popad
50229                                  <1>
50230 00011573 803D[4F610100]01        <1>       cmp     byte [audio_intr], 1 ; IRQ level was changed?
50231                                  <1>
50232 0001157A C3                      <1>       retn
50233                                  <1>
50234                                  <1> %macro      SbOut  1
50235                                  <1> %%Wait:
50236                                  <1>       in     al, dx
50237                                  <1>       or     al, al
50238                                  <1>       js     short %%Wait
50239                                  <1>       mov    al, %1
50240                                  <1>       out    dx, al
50241                                  <1> %endmacro
50242                                  <1>
50243                                  <1> SbInit_play:
50244                                  <1>       ; 13/07/2017
50245                                  <1>       ; 24/06/2017
50246                                  <1>       ; 15/05/2017
50247                                  <1>       ; 24/04/2017
50248                                  <1>       ;pushad
50249                                  <1> SetBuffer:
50250                                  <1>       ;mov   byte [DmaFlag], 0
50251                                  <1>
50252 0001157B 8B1D[6C610100]          <1>       mov    ebx, [audio_dma_buff] ; physical addr of DMA buff
50253 00011581 89DF                    <1>       mov    edi, ebx
50254 00011583 8B0D[70610100]          <1>       mov     ecx, [audio_dmabuff_size]
50255 00011589 49                      <1>       dec     ecx
50256                                  <1>
50257 0001158A 803D[7C610100]10        <1>       cmp    byte [audio_bps], 16
50258 00011591 7529                    <1>       jne    short sbInit_0 ; set 8 bit DMA buffer
50259                                  <1>
50260                                  <1>       ; 16 bit DMA buffer setting (DMA channel 5)
50261 00011593 B005                    <1>       mov     al, 05h ; set mask bit for channel 5  (4+1)
50262 00011595 E6D4                    <1>       out    0D4h, al
50263                                  <1>
50264 00011597 30C0                    <1>       xor     al, al ; stops all DMA processes on selected channel
50265 00011599 E6D8                    <1>       out    0D8h, al ; clear selected channel register
50266                                  <1>
50267 0001159B 88D8                    <1>       mov     al, bl      ; byte 0 of DMA buffer address (physical)
50268 0001159D E6C4                    <1>       out    0C4h, al ; DMA channel 5 port number
50269                                  <1>
50270 0001159F 88F8                    <1>       mov     al, bh  ; byte 1 of DMA buffer address (physical)
50271 000115A1 E6C4                    <1>       out    0C4h, al
50272                                  <1>
50273 000115A3 C1EB10                  <1>       shr    ebx, 16
50274                                  <1>
50275 000115A6 88D8                    <1>       mov     al, bl ; byte 2 of DMA buffer address (physical)
```

```
50276 000115A8 E68B              <1>       out    8Bh, al ; page register port addr for channel 5 ; 13/07/2017
50277                            <1>
50278 000115AA 88C8              <1>       mov     al, cl ; low byte of DMA count - 1
50279 000115AC E6C6              <1>       out    0C6h, al ; count register port addr for channel 1
50280                            <1>
50281 000115AE 88E8              <1>       mov     al, ch ; high byte of DMA count - 1
50282 000115B0 E6C6              <1>       out    0C6h, al
50283                            <1>
50284                            <1>       ; channel 5, read, autoinitialized, single mode
50285 000115B2 B049              <1>       mov     al, 49h
50286 000115B4 E6D6              <1>       out    0D6h, al ; DMA mode register port address
50287                            <1>
50288 000115B6 B001              <1>       mov     al, 01h ; clear mask bit for channel 1
50289 000115B8 E6D4              <1>       out    0D4h, al ; DMA mask register port address
50290                            <1>
50291 000115BA EB27              <1>       jmp    short ClearBuffer
50292                            <1>
50293                            <1> sbInit_0:
50294                            <1>       ; 8 bit DMA buffer setting (DMA channel 1)
50295 000115BC B005              <1>       mov     al, 05h ; set mask bit for channel 1  (4+1)
50296 000115BE E60A              <1>       out    0Ah, al ; DMA mask register
50297                            <1>
50298 000115C0 30C0              <1>       xor     al, al ; stops all DMA processes on selected channel
50299 000115C2 E60C              <1>       out    0Ch, al ; clear selected channel register
50300                            <1>
50301 000115C4 88D8              <1>       mov     al, bl      ; byte 0 of DMA buffer address (physical)
50302 000115C6 E602              <1>       out    02h, al ; DMA channel 1 port number
50303                            <1>
50304 000115C8 88F8              <1>       mov     al, bh  ; byte 1 of DMA buffer address (physical)
50305 000115CA E602              <1>       out    02h, al
50306                            <1>
50307 000115CC C1EB10            <1>       shr    ebx, 16
50308                            <1>
50309 000115CF 88D8              <1>       mov     al, bl ; byte 2 of DMA buffer address (physical)
50310 000115D1 E683              <1>       out    83h, al ; page register port addr for channel 1
50311                            <1>
50312 000115D3 88C8              <1>       mov     al, cl ; low byte of DMA count - 1
50313 000115D5 E603              <1>       out    03h, al ; count register port addr for channel 1
50314                            <1>
50315 000115D7 88E8              <1>       mov     al, ch ; high byte of DMA count - 1
50316 000115D9 E603              <1>       out    03h, al
50317                            <1>
50318                            <1>       ; channel 1, read, autoinitialized, single mode
50319 000115DB B049              <1>       mov     al, 49h
50320 000115DD E60B              <1>       out    0Bh, al ; DMA mode register port address
50321                            <1>
50322 000115DF B001              <1>       mov     al, 01h ; clear mask bit for channel 1
50323 000115E1 E60A              <1>       out    0Ah, al ; DMA mask register port address
50324                            <1>
50325                            <1> ClearBuffer:
50326                            <1>       ;;mov  edi, [audio_dma_buff]
50327                            <1>       ;;mov  ecx, [audio_dmabuff_size]
50328                            <1>       ;inc   ecx
50329                            <1>       ;mov     al, 80h
50330                            <1>       ;;cld
50331                            <1>       ;rep     stosb
50332                            <1> SetIrq:
50333                            <1>       ;mov   ebx, SbIrqhandler
50334                            <1>       ;mov   al, [audio_intr] ; IRQ number
50335                            <1>       ;call  set_dev_IRQ_service
50336                            <1>       ;; SETUP (audio) INTERRUPT CALLBACK SERVICE
50337                            <1>       ;mov   bl, [audio_intr] ; IRQ number
50338                            <1>       ;mov   bh, [audio_cb_mode]
50339                            <1>       ;inc   bh  ; 1 = Signal Response Byte method (fixed value)
50340                            <1>       ;          ; 2 = Callback service method
50341                            <1>       ;          ; 3 = Auto Increment S.R.B. method
50342                            <1>       ;mov   cl, [audio_srb]
50343                            <1>       ;mov   edx, [audio_cb_addr]
50344                            <1>       ;mov   al, [audio_user]
50345                            <1>       ;call  set_irq_callback_service
50346                            <1> ResetDsp:
50347 000115E3 668B15[52610100]  <1>       mov     dx, [audio_io_base]
50348 000115EA 6683C206          <1>       add     dx, 06h
50349 000115EE B001              <1>       mov     al, 1
50350 000115F0 EE                <1>       out    dx, al
50351                            <1>
50352 000115F1 EC                <1>       in     al, dx
50353 000115F2 EC                <1>       in     al, dx
50354 000115F3 EC                <1>       in     al, dx
50355 000115F4 EC                <1>       in     al, dx
50356                            <1>
50357 000115F5 30C0              <1>       xor     al, al
50358 000115F7 EE                <1>       out    dx, al
50359                            <1>
50360 000115F8 66B96400          <1>       mov     cx, 100
50361 000115FC 28E4              <1>       sub    ah, ah ; 0
50362                            <1> WaitId:
50363 000115FE 668B15[52610100]  <1>       mov     dx, [audio_io_base]
50364 00011605 6683C20E          <1>       add     dx, 0Eh
50365 00011609 EC                <1>       in     al, dx
50366 0001160A 08C0              <1>       or      al, al
50367 0001160C 7807              <1>       js      short sb_GetId
50368 0001160E E2EE              <1>       loop    WaitId
50369 00011610 E9A9000000        <1>       jmp     sb_Exit
50370                            <1> sb_GetId:
50371 00011615 668B15[52610100]  <1>       mov     dx, [audio_io_base]
50372 0001161C 6683C20A          <1>       add     dx, 0Ah
50373 00011620 EC                <1>       in     al, dx
50374 00011621 3CAA              <1>       cmp     al, 0AAh
50375 00011623 7407              <1>       je      short SbOk
50376 00011625 E2D7              <1>       loop    WaitId
50377 00011627 E992000000        <1>       jmp    sb_Exit
```

```
50378                                  <1> SbOk:
50379 0001162C 668B15[52610100]        <1>       mov     dx, [audio_io_base]
50380 00011633 6683C20C                <1>       add     dx, 0Ch
50381                                  <1>       SbOut   0D1h ; Turn on speaker
50382                                  <2> %%Wait:
50383 00011637 EC                      <2>  in al, dx
50384 00011638 08C0                    <2>  or al, al
50385 0001163A 78FB                    <2>  js short %%Wait
50386 0001163C B0D1                    <2>  mov al, %1
50387 0001163E EE                      <2>  out dx, al
50388                                  <1>       SbOut   41h ; 8 bit or 16 bit transfer
50389                                  <2> %%Wait:
50390 0001163F EC                      <2>  in al, dx
50391 00011640 08C0                    <2>  or al, al
50392 00011642 78FB                    <2>  js short %%Wait
50393 00011644 B041                    <2>  mov al, %1
50394 00011646 EE                      <2>  out dx, al
50395 00011647 668B1D[7E610100]        <1>       mov    bx, [audio_freq] ; sampling rate (Hz)
50396                                  <1>       SbOut  bh ; sampling rate high byte
50397                                  <2> %%Wait:
50398 0001164E EC                      <2>  in al, dx
50399 0001164F 08C0                    <2>  or al, al
50400 00011651 78FB                    <2>  js short %%Wait
50401 00011653 88F8                    <2>  mov al, %1
50402 00011655 EE                      <2>  out dx, al
50403                                  <1>       SbOut  bl ; sampling rate low byte
50404                                  <2> %%Wait:
50405 00011656 EC                      <2>  in al, dx
50406 00011657 08C0                    <2>  or al, al
50407 00011659 78FB                    <2>  js short %%Wait
50408 0001165B 88D8                    <2>  mov al, %1
50409 0001165D EE                      <2>  out dx, al
50410                                  <1>
50411                                  <1>       ; 20/05/2017
50412                                  <1>       ;call  sb16_volume_initial ; 15/05/2017
50413 0001165E E8CB000000              <1>       call   sb16_volume
50414                                  <1>
50415                                  <1> StartDma:
50416                                  <1>       ; autoinitialized mode
50417 00011663 803D[7C610100]10        <1>       cmp    byte [audio_bps], 16 ; 16 bit samples
50418 0001166A 7411                    <1>       je     short sb_play_1
50419                                  <1>       ; 8 bit samples
50420 0001166C 66BBC600                <1>       mov    bx, 0C6h ; 8 bit output (0C6h)
50421 00011670 803D[7D610100]02        <1>       cmp    byte [audio_stmo], 2 ; 1 = mono, 2 = stereo
50422 00011677 7214                    <1>       jb     short sb_play_2
50423 00011679 B720                    <1>       mov    bh, 20h      ; 8 bit stereo (20h)
50424 0001167B EB10                    <1>       jmp    short sb_play_2
50425                                  <1> sb_play_1:
50426                                  <1>       ; 16 bit samples
50427 0001167D 66BBB610                <1>       mov    bx, 10B6h ; 16 bit output (0B6h)
50428 00011681 803D[7D610100]02        <1>       cmp    byte [audio_stmo], 2 ; 1 = mono, 2 = stereo
50429 00011688 7203                    <1>       jb     short sb_play_2
50430 0001168A 80C720                  <1>       add    bh, 20h      ; 16 bit stereo (30h)
50431                                  <1> sb_play_2:
50432                                  <1>       ; PCM output (8/16 bit mono autoinitialized transfer)
50433                                  <1>       SbOut   bl ; bCommand
50434                                  <2> %%Wait:
50435 0001168D EC                      <2>  in al, dx
50436 0001168E 08C0                    <2>  or al, al
50437 00011690 78FB                    <2>  js short %%Wait
50438 00011692 88D8                    <2>  mov al, %1
50439 00011694 EE                      <2>  out dx, al
50440                                  <1>       SbOut  bh ; bMode
50441                                  <2> %%Wait:
50442 00011695 EC                      <2>  in al, dx
50443 00011696 08C0                    <2>  or al, al
50444 00011698 78FB                    <2>  js short %%Wait
50445 0001169A 88F8                    <2>  mov al, %1
50446 0001169C EE                      <2>  out dx, al
50447 0001169D 8B1D[70610100]          <1>       mov    ebx, [audio_dmabuff_size]  ; 15/05/2017
50448 000116A3 D1EB                    <1>       shr    ebx, 1 ; half buffer size
50449 000116A5 664B                    <1>       dec    bx  ; wBlkSize is one less than the actual size
50450                                  <1>       SbOut   bl
50451                                  <2> %%Wait:
50452 000116A7 EC                      <2>  in al, dx
50453 000116A8 08C0                    <2>  or al, al
50454 000116AA 78FB                    <2>  js short %%Wait
50455 000116AC 88D8                    <2>  mov al, %1
50456 000116AE EE                      <2>  out dx, al
50457                                  <1>       SbOut   bh
50458                                  <2> %%Wait:
50459 000116AF EC                      <2>  in al, dx
50460 000116B0 08C0                    <2>  or al, al
50461 000116B2 78FB                    <2>  js short %%Wait
50462 000116B4 88F8                    <2>  mov al, %1
50463 000116B6 EE                      <2>  out dx, al
50464                                  <1>
50465 000116B7 C605[80610100]01        <1>       mov    byte [audio_play_cmd], 1 ; playing !
50466                                  <1>
50467                                  <1>       ;; Set Voice and master volumes
50468                                  <1>       ;mov   dx, [audio_io_base]
50469                                  <1>       ;add   dl, 4 ; Mixer chip Register Address Port
50470                                  <1>       ;SbOut 30h   ; select Master Volume Register (L)
50471                                  <1>       ;inc   dl    ; Mixer chip Register Data Port
50472                                  <1>       ;SbOut 0F8h  ; Max. volume value is 31 (31*8)
50473                                  <1>       ;dec   dl
50474                                  <1>       ;SbOut 31h   ; select Master Volume Register (R)
50475                                  <1>       ;inc   dl
50476                                  <1>       ;SbOut 0F8h  ; Max. volume value is 31 (31*8)
50477                                  <1>       ;dec   dl
50478                                  <1>       ;SbOut 32h   ; select Voice Volume Register (L)
50479                                  <1>       ;inc   dl
```

```
                                <1>       ;SbOut 0F8h  ; Max. volume value is 31 (31*8)
                                <1>       ;dec   dl
                                <1>       ;SbOut 33h   ; select Voice Volume Register (R)
                                <1>       ;inc   dl
                                <1>       ;SbOut 0F8h  ; Max. volume value is 31 (31*8)
                                <1>       ;;
                                <1>       ;dec   dl
                                <1>       ;SbOut 44h   ; select Treble Register (L)
                                <1>       ;inc   dl
                                <1>       ;SbOut 0F0h  ; Max. Treble value is 15 (15*16)
                                <1>       ;dec   dl
                                <1>       ;SbOut 45h   ; select Treble Register (R)
                                <1>       ;inc   dl
                                <1>       ;SbOut 0F0h  ; Max. Treble value is 15 (15*16)
                                <1>       ;dec   dl
                                <1>       ;SbOut 46h   ; select Bass Register (L)
                                <1>       ;inc   dl
                                <1>       ;SbOut 0F0h  ; Max. Bass value is 15 (15*16)
                                <1>       ;dec   dl
                                <1>       ;SbOut 47h   ; select Bass Register (R)
                                <1>       ;inc   dl
                                <1>       ;SbOut 0F0h  ; Max. Bass value is 15 (15*16)
                                <1>
                                <1> sb_Exit:
                                <1>       ;popad
000116BE 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>       ; 12/05/2017
                                <1>       ; 24/04/2017 (TRDOS 386 kernel, 'audio.s')
                                <1>       ; 10/03/2017 - 'PLAYWAV.PRG' ('playwav.s')
                                <1>
                                <1>       ;push  eax
                                <1>       ;pusb  ebx
                                <1>       ;push  ecx
                                <1>       ;push  edx
                                <1>       ;push  esi
                                <1>       ;push  edi
                                <1>
000116BF 668B15[52610100]       <1>       mov     dx, [audio_io_base]
000116C6 6683C20E               <1>       add     dx, 0Eh
000116CA EC                     <1>       in      al, dx
                                <1>
000116CB 803D[50610100]00       <1>       cmp     byte [audio_busy], 0
000116D2 772D                   <1>       ja      short sb_irq_h3
                                <1>
000116D4 C605[50610100]01       <1>       mov     byte [audio_busy], 1
                                <1>
000116DB 803D[80610100]01       <1>       cmp     byte [audio_play_cmd], 1
000116E2 7307                   <1>       jnb     short sb_irq_h1
                                <1> sb_irq_h0:
000116E4 E8B9000000             <1>       call    sb16_stop
000116E9 EB16                   <1>       jmp     short sb_irq_h3
                                <1> sb_irq_h1:
000116EB E819000000             <1>       call    sb16_tuneloop
                                <1>       ;
000116F0 30C0                   <1>       xor     al, al ; 0
000116F2 3805[74610100]         <1>       cmp     [audio_flag], al ; 0
000116F8 7702                   <1>       ja      short sb_irq_h2
000116FA FEC0                   <1>       inc     al ; 1
                                <1> sb_irq_h2:
000116FC A2[74610100]           <1>       mov     [audio_flag], al
                                <1> sb_irq_h3:
00011701 C605[50610100]00       <1>       mov     byte [audio_busy], 0
                                <1>
                                <1>       ;pop   edi
                                <1>       ;pop   esi
                                <1>       ;pop   edx
                                <1>       ;pop   ecx
                                <1>       ;pop   ebx
                                <1>       ;pop   eax
                                <1>
00011708 C3                     <1>       retn
                                <1>
                                <1> sb16_tuneloop:
                                <1>       ;cmp   byte [audio_play_cmd], 1
                                <1>       ;jz    short sb_tlp1 ; stop command !
                                <1>
00011709 8B3D[6C610100]         <1>       mov    edi, [audio_dma_buff]
0001170F 8B0D[70610100]         <1>       mov    ecx, [audio_dmabuff_size]
00011715 D1E9                   <1>       shr    ecx, 1 ; dma buff size / 2 = half buffer size
                                <1>
                                <1>       ; 22/05/2017
00011717 F605[74610100]01       <1>       test   byte [audio_flag], 1  ; Current flag value
0001171E 7402                   <1>       jz     short sb_tlp1 ; EOL (Half Buffer 1 must be filled)
                                <1>       ; FLAG (Half Buffer 2 must be filled)
00011720 01CF                   <1>       add    edi, ecx
                                <1>       ; 15/05/2017
                                <1> sb_tlp1:
00011722 8B35[64610100]         <1>       mov    esi, [audio_p_buffer] ; phy addr of audio buff
                                <1>       ;rep   movsb
00011728 C1E902                 <1>       shr    ecx, 2 ; half buff size / 4
0001172B F3A5                   <1>       rep    movsd
0001172D C3                     <1>       retn
                                <1>
                                <1> ;sb16_volume_initial:
                                <1> ;     ; 15/05/2017
                                <1> ;     mov    word [audio_master_volume], 1010h ; middle level
                                <1> sb16_volume:
0001172E 6652                   <1>       push   dx
```

```
50582 00011730 668B15[52610100]    <1>       mov    dx, [audio_io_base]
50583 00011737 6683C204            <1>       add    dx, 4 ; Mixer chip address port
50584 0001173B B022                <1>       mov    al, 22h ; master volume
50585 0001173D EE                  <1>       out    dx, al
50586 0001173E 6642                <1>       inc    dx
50587 00011740 8A25[82610100]      <1>       mov    ah, [audio_master_volume_l]
50588 00011746 C0EC02              <1>       shr    ah, 2 ; 32 -> 8 level
50589 00011749 C0E405              <1>       shl    ah, 5 ; bit 5 to 7
50590 0001174C A0[83610100]        <1>       mov    al, [audio_master_volume_r]
50591 00011751 C0E802              <1>       shr    al, 2 ; 32 -> 8 level
50592                              <1>       ;and   al, 0Fh
50593 00011754 D0E0                <1>       shl    al, 1 ; bit 1 to 3
50594 00011756 08E0                <1>       or     al, ah
50595 00011758 EE                  <1>       out    dx, al
50596 00011759 665A                <1>       pop    dx
50597 0001175B C3                  <1>       retn
50598                              <1>
50599                              <1> sb16_pause:
50600 0001175C 668B15[52610100]    <1>       mov    dx, [audio_io_base]
50601 00011763 6683C20C            <1>       add    dx, 0Ch ; Command & Data Port
50602 00011767 803D[7C610100]10    <1>       cmp    byte [audio_bps], 16 ; 16 bit samples
50603 0001176E 7404                <1>       je     short sb_pause_1
50604                              <1>       ; 8 bit samples
50605 00011770 B3D0                <1>       mov    bl, 0D0h ; 8 bit DMA mode
50606 00011772 EB02                <1>       jmp    short sb_pause_2
50607                              <1> sb_pause_1:
50608                              <1>       ; 16 bit samples
50609 00011774 B3D5                <1>       mov    bl, 0D5h ; 16 bit DMA mode
50610                              <1> sb_pause_2:
50611                              <1>       SbOut   bl ; bCommand
50612                              <2> %%Wait:
50613 00011776 EC                  <2>  in al, dx
50614 00011777 08C0                <2>  or al, al
50615 00011779 78FB                <2>  js short %%Wait
50616 0001177B 88D8                <2>  mov al, %1
50617 0001177D EE                  <2>  out dx, al
50618                              <1> sb_pause_3:
50619 0001177E C3                  <1>       retn
50620                              <1>
50621                              <1> sb16_continue:
50622 0001177F 668B15[52610100]    <1>       mov    dx, [audio_io_base]
50623 00011786 6683C20C            <1>       add    dx, 0Ch ; Command & Data Port
50624 0001178A 803D[7C610100]10    <1>       cmp    byte [audio_bps], 16 ; 16 bit samples
50625 00011791 7404                <1>       je     short sb_cont_1
50626                              <1>       ; 8 bit samples
50627 00011793 B3D4                <1>       mov    bl, 0D4h ; 8 bit DMA mode
50628 00011795 EB02                <1>       jmp    short sb_cont_2
50629                              <1> sb_cont_1:
50630                              <1>       ; 16 bit samples
50631 00011797 B3D6                <1>       mov    bl, 0D6h ; 16 bit DMA mode
50632                              <1> sb_cont_2:
50633                              <1>       SbOut   bl ; bCommand
50634                              <2> %%Wait:
50635 00011799 EC                  <2>  in al, dx
50636 0001179A 08C0                <2>  or al, al
50637 0001179C 78FB                <2>  js short %%Wait
50638 0001179E 88D8                <2>  mov al, %1
50639 000117A0 EE                  <2>  out dx, al
50640                              <1> sb_cont_3:
50641 000117A1 C3                  <1>       retn
50642                              <1>
50643                              <1> sb16_stop:
50644                              <1>       ; 24/04/2017
50645 000117A2 803D[80610100]00    <1>       cmp    byte [audio_play_cmd], 0
50646 000117A9 7648                <1>       jna    short sb16_stop_4
50647                              <1>
50648                              <1>       ; 22/05/2017
50649 000117AB 668B15[52610100]    <1>       mov    dx, [audio_io_base]
50650 000117B2 6683C20C            <1>       add    dx, 0Ch
50651                              <1>
50652 000117B6 B3D9                <1>       mov    bl, 0D9h ; exit auto-initialize 16 bit transfer
50653                              <1>       ; stop  autoinitialized DMA transfer mode
50654 000117B8 803D[7C610100]10    <1>       cmp    byte [audio_bps], 16 ; 16 bit samples
50655 000117BF 7402                <1>       je     short sb16_stop_1
50656                              <1>       ;mov   bl, 0DAh ; exit auto-initialize 8 bit transfer
50657 000117C1 FEC3                <1>       inc    bl
50658                              <1> sb16_stop_1:
50659                              <1>       SbOut  bl ; exit auto-initialize transfer command
50660                              <2> %%Wait:
50661 000117C3 EC                  <2>  in al, dx
50662 000117C4 08C0                <2>  or al, al
50663 000117C6 78FB                <2>  js short %%Wait
50664 000117C8 88D8                <2>  mov al, %1
50665 000117CA EE                  <2>  out dx, al
50666                              <1>
50667 000117CB 30C0                <1>       xor     al, al ; stops all DMA processes on selected channel
50668                              <1>
50669 000117CD 803D[7C610100]10    <1>       cmp    byte [audio_bps], 16 ; 16 bit samples
50670 000117D4 7404                <1>       je     short sb16_stop_2
50671 000117D6 E60C                <1>       out    0Ch, al ; clear selected channel register
50672 000117D8 EB02                <1>       jmp    short sb16_stop_3
50673                              <1>
50674                              <1> sb16_stop_2:
50675 000117DA E6D8                <1>       out    0D8h, al ; clear selected channel register
50676                              <1>
50677                              <1> sb16_stop_3:
50678 000117DC C605[80610100]00    <1>       mov    byte [audio_play_cmd], 0 ; stop !
50679                              <1> SbDone:
50680                              <1>       ;mov   dx, [audio_io_base]
50681                              <1>       ;add   dx, 0Ch
50682                              <1>       SbOut   0D0h
50683                              <2> %%Wait:
```

```
50684 000117E3 EC                  <2>  in al, dx
50685 000117E4 08C0                <2>  or al, al
50686 000117E6 78FB                <2>  js short %%Wait
50687 000117E8 B0D0                <2>  mov al, %1
50688 000117EA EE                  <2>  out dx, al
50689                              <1>       SbOut   0D3h
50690                              <2> %%Wait:
50691 000117EB EC                  <2>  in al, dx
50692 000117EC 08C0                <2>  or al, al
50693 000117EE 78FB                <2>  js short %%Wait
50694 000117F0 B0D3                <2>  mov al, %1
50695 000117F2 EE                  <2>  out dx, al
50696                              <1> sb16_stop_4:
50697 000117F3 C3                  <1>       retn
50698                              <1>
50699                              <1> sb16_reset:
50700                              <1>       ; 24/04/2017
50701 000117F4 668B15[52610100]    <1>       mov     dx, [audio_io_base] ; try to reset the DSP.
50702 000117FB 6683C206            <1>       add     dx, 06h
50703 000117FF B001                <1>       mov    al, 1
50704 00011801 EE                  <1>       out    dx, al
50705                              <1>
50706 00011802 EC                  <1>       in     al, dx
50707 00011803 EC                  <1>       in     al, dx
50708 00011804 EC                  <1>       in     al, dx
50709 00011805 EC                  <1>       in     al, dx
50710                              <1>
50711 00011806 30C0                <1>       xor     al, al
50712 00011808 EE                  <1>       out    dx, al
50713                              <1>
50714 00011809 6683C208            <1>       add     dx, 08h
50715 0001180D 66B96400            <1>       mov    cx, 100
50716                              <1> sbrstWaitID:
50717 00011811 EC                  <1>       in     al, dx
50718 00011812 08C0                <1>       or      al, al
50719 00011814 7804                <1>       js      short sbrstGetID
50720 00011816 E2F9                <1>       loop    sbrstWaitID
50721 00011818 F9                  <1>       stc
50722 00011819 C3                  <1>       retn
50723                              <1> sbrstGetID:
50724 0001181A 6683EA04            <1>       sub     dx, 04h
50725 0001181E EC                  <1>       in     al, dx
50726 0001181F 3CAA                <1>       cmp     al, 0AAh
50727 00011821 7406                <1>       je      short sb_rst_retn
50728 00011823 6683C204            <1>       add     dx, 04h
50729 00011827 E2E8                <1>       loop    sbrstWaitID
50730                              <1> sb_rst_retn:
50731 00011829 C3                  <1>       retn
50732                              <1>
50733                              <1> ac97_codec_config:
50734                              <1>       ; 10/06/2017
50735                              <1>       ; 05/06/2017
50736                              <1>       ; 29/05/2017
50737                              <1>       ; 28/05/2017 (TRDOS 386, 'audio.s')
50738                              <1>       ; 07/11/2016 (Erdogan Tan)
50739                              <1>       ; Derived from 'codecConfig' procedure in 'CODEC.ASM'
50740                              <1>       ; .wav player for DOS by Jeff Leyda (02/09/2002)
50741                              <1>
50742                              <1>       ;; 'PLAYER.ASM'
50743                              <1>       ;; get ICH base address regs for mixer and bus master
50744                              <1>
50745                              <1> init_ac97_controller: ; 10/06/2017
50746 0001182A A1[54610100]        <1>       mov    eax, [audio_dev_id]
50747                              <1>         ;mov al, NAMBAR_REG
50748                              <1>         ;;call  pciRegRead16                ; read PCI registers 10-11
50749                              <1>         ;call      pciRegRead32
50750                              <1>       ;and   dx, IO_ADDR_MASK          ; mask off BIT0
50751                              <1>       ;;and  edx, IO_ADDR_MASK
50752                              <1>
50753                              <1>         ;mov [NAMBAR], dx              ; save audio mixer base addr
50754                              <1>
50755                              <1>         ;mov    al, NABMBAR_REG
50756                              <1>         ;;call     pciRegRead16
50757                              <1>         ;call      pciRegRead32
50758                              <1>       ;and   dx, 0FFC0h ; IO_ADDR_MASK
50759                              <1>       ;;and  edx, 0FFC0h
50760                              <1>
50761                              <1>         ;mov    [NABMBAR], dx               ; save bus master base addr
50762                              <1>
50763                              <1>       ;mov   eax, [audio_dev_id]
50764 0001182F B004                <1>         mov     al, PCI_CMD_REG
50765                              <1>         ;call      pciRegRead8              ; read PCI command register
50766 00011831 E81AF8FFFF          <1>         call pciRegRead16
50767 00011836 80CA05              <1>       or      dl, IO_ENA+BM_ENA              ; enable IO and bus master
50768                              <1>         ;call      pciRegWrite8
50769 00011839 E87DF8FFFF          <1>       call   pciRegWrite16
50770                              <1>
50771                              <1>       ; 'CODEC.ASM'
50772                              <1>
50773                              <1>       ; enable codec, unmute stuff, set output rate
50774                              <1> ;     ; entry: [audio_freq] = desired sample rate
50775                              <1>
50776                              <1> ;     mov          dx, [NAMBAR]
50777                              <1> ;     add          dx, CODEC_EXT_AUDIO_CTRL_REG     ; 2Ah
50778                              <1> ;     in           ax, dx
50779                              <1> ;     or     ax, 1
50780                              <1> ;     out    dx, ax                          ; Enable variable rate audio
50781                              <1>
50782                              <1> ;       ;call    delay1_4ms
50783                              <1> ;       ;call    delay1_4ms
50784                              <1> ;       ;call    delay1_4ms
50785                              <1> ;       ;call    delay1_4ms
```

```
                                <1>
                                <1> ;     mov    ax, [audio_freq]          ; sample rate
                                <1>
                                <1> ;     mov          dx, [NAMBAR]
                                <1> ;     add          dx, CODEC_PCM_FRONT_DACRATE_REG  ; 2Ch
                                <1> ;     out    dx, ax                          ; out sample rate
                                <1>
                                <1> ;       ;call      delay1_4ms
                                <1> ;       ;call      delay1_4ms
                                <1> ;       ;call      delay1_4ms
                                <1> ;       ;call      delay1_4ms
                                <1>
                                <1>       ;mov   dx, [NAMBAR]              ; mixer base address
                                <1>         ;add dx, CODEC_RESET_REG              ; reset register
                                <1>         ;mov ax, 42
                                <1>       ;out   dx, ax                          ; reset
                                <1>
                                <1>       ;mov    dx, [NABMBAR]                  ; bus master base address
                                <1>         ;add dx, GLOB_STS_REG
                                <1>         ;mov ax, 2
                                <1>       ;out   dx, ax
                                <1>
0001183E E80BF9FFFF             <1>         call    delay_100ms ; 29/05/2017
                                <1>
                                <1> init_ac97_codec:
                                <1>       ; 10/06/2017
                                <1>       ; 29/05/2017
                                <1>       ; 28/05/2017 - Erdogan Tan (Ref: KolibriOS, intelac97.asm)
                                <1>       ;
00011843 66BA2C00               <1>       mov    dx, GLOB_CNT_REG ; 2Ch
00011847 660315[86610100]       <1>       add    dx, [NABMBAR]
0001184E ED                     <1>       in     eax, dx
                                <1>       ; ?
0001184F 66BA3000               <1>       mov    dx, GLOB_STS_REG ; 30h
00011853 660315[86610100]       <1>       add    dx, [NABMBAR]
0001185A ED                     <1>       in     eax, dx
                                <1>
0001185B 83F8FF                 <1>       cmp    eax, 0FFFFFFFFh ; -1
0001185E 744B                   <1>       je     short init_ac97_codec_err1
                                <1>
00011860 A900030010             <1>       test   eax, CTRL_ST_CREADY
00011865 7507                   <1>       jnz    short _ac97_codec_ready
                                <1>
00011867 E8F1020000             <1>       call   reset_ac97_codec
0001186C 723E                   <1>       jc     short init_ac97_codec_err2
                                <1>
                                <1> _ac97_codec_ready:
0001186E 668B15[84610100]       <1>       mov    dx, [NAMBAR]
                                <1>       ;add   dx, 0 ; ac_reg_0 ; reset register
00011875 66EF                   <1>       out    dx, ax
                                <1>
00011877 31C0                   <1>       xor    eax, eax ; 0
00011879 668B15[84610100]       <1>       mov    dx, [NAMBAR]
00011880 6683C226               <1>       add    dx, CODEC_REG_POWERDOWN
00011884 66EF                   <1>       out    dx, ax
                                <1>
                                <1>       ; 10/06/2017
                                <1>       ; 29/05/2017
                                <1>       ; wait for 1 second
00011886 B9E8030000             <1>       mov    ecx, 1000 ; 1000*0.25ms = 1s
                                <1> _ac97_codec_rloop:
0001188B E8CBF8FFFF             <1>       call   delay1_4ms
00011890 E8C6F8FFFF             <1>       call   delay1_4ms
00011895 E8C1F8FFFF             <1>       call   delay1_4ms
0001189A E8BCF8FFFF             <1>       call   delay1_4ms
                                <1>       ;mov   dx, [NAMBAR]
                                <1>       ;add   dx, CODEC_REG_POWERDOWN
0001189F 66ED                   <1>       in     ax, dx
000118A1 6683E00F               <1>       and    ax, 0Fh
000118A5 3C0F                   <1>       cmp    al, 0Fh
000118A7 7404                   <1>       je     short _ac97_codec_init_ok
000118A9 E2E0                   <1>       loop   _ac97_codec_rloop
                                <1>
                                <1> init_ac97_codec_err1:
000118AB F9                     <1>       stc
                                <1> init_ac97_codec_err2:
000118AC C3                     <1>       retn
                                <1>
                                <1> _ac97_codec_init_ok:
000118AD B002                   <1>         mov     al, 2 ; force set 16-bit 2-channel PCM
000118AF 66BA2C00               <1>       mov    dx, GLOB_CNT_REG ; 2Ch
000118B3 660315[86610100]       <1>       add    dx, [NABMBAR]
000118BA EF                     <1>       out    dx, eax
                                <1>
                                <1>       ;call  delay1_4ms
                                <1>
                                <1>       ; 10/06/2017
000118BB E84B020000             <1>       call   reset_ac97_controller
                                <1>
                                <1> ;     call   setup_ac97_codec
                                <1> ;
                                <1> ;detect_ac97_codec:
                                <1> ;     retn
                                <1>
                                <1> setup_ac97_codec:
                                <1>       ; 10/06/2017
                                <1>       ; 29/05/2017
000118C0 B802020000             <1>       mov     eax, 0202h
000118C5 66A3[82610100]         <1>       mov    [audio_master_volume], ax
000118CB 66B81F1F               <1>       mov    ax, 1F1Fh ; 31, 31
                                <1>
000118CF 668B15[84610100]       <1>       mov     dx, [NAMBAR]
```

```
50888 000118D6 6683C202            <1>       add     dx, CODEC_MASTER_VOL_REG        ;02h
50889 000118DA 6631C0              <1>       xor    ax, ax       ; volume attenuation = 0 (max. volume)
50890 000118DD 66EF                <1>       out     dx, ax
50891                              <1>
50892 000118DF 668B15[84610100]    <1>       mov     dx, [NAMBAR]
50893 000118E6 6683C206            <1>       add     dx, CODEC_MASTER_MONO_VOL_REG   ;06h
50894                              <1>       ;xor   ax, ax
50895 000118EA 66EF                <1>       out     dx, ax
50896                              <1>
50897 000118EC 668B15[84610100]    <1>       mov     dx, [NAMBAR]
50898 000118F3 6683C20A            <1>       add     dx, CODEC_PCBEEP_VOL_REG        ;0Ah
50899                              <1>       ;xor    ax, ax
50900 000118F7 66EF                <1>       out     dx, ax
50901                              <1>
50902 000118F9 668B15[84610100]    <1>       mov     dx, [NAMBAR]
50903 00011900 6683C218            <1>       add     dx, CODEC_PCM_OUT_REG           ;18h
50904                              <1>       ;xor    ax, ax
50905 00011904 66EF                <1>       out     dx, ax
50906                              <1>
50907 00011906 66B80880            <1>       mov     ax,  8008h ; Mute
50908 0001190A 668B15[84610100]    <1>       mov     dx, [NAMBAR]
50909 00011911 6683C20C            <1>       add    dx, 0Ch       ; AC97_PHONE_VOL ; TAD Input (Mono)
50910 00011915 66EF                <1>       out     dx, ax
50911                              <1>
50912 00011917 66B80808            <1>         mov  ax,  0808h
50913 0001191B 668B15[84610100]    <1>       mov     dx, [NAMBAR]
50914 00011922 6683C210            <1>         add  dx, CODEC_LINE_IN_VOL_REG ; 10h ; Line Input (Stereo)
50915 00011926 66EF                <1>       out     dx, ax
50916                              <1>
50917                              <1>       ;mov   ax,  0808h
50918 00011928 668B15[84610100]    <1>       mov     dx, [NAMBAR]
50919 0001192F 6683C212            <1>         add  dx, CODEC_CD_VOL_REG ; 12h ; CR Input (Stereo)
50920 00011933 66EF                <1>       out     dx, ax
50921                              <1>
50922                              <1>       ;mov   ax,  0808h
50923 00011935 668B15[84610100]    <1>       mov     dx, [NAMBAR]
50924 0001193C 6683C216            <1>         add  dx, CODEC_AUX_VOL_REG ; 16h ; Aux Input (Stereo)
50925 00011940 66EF                <1>       out     dx, ax
50926                              <1>
50927                              <1>         ;call    delay1_4ms
50928                              <1>         ;call    delay1_4ms
50929                              <1>         ;call    delay1_4ms
50930                              <1>         ;call    delay1_4ms
50931                              <1>
50932                              <1> detect_ac97_codec:
50933 00011942 C3                  <1>         retn
50934                              <1>
50935                              <1> set_ac97_bdl: ; Set AC97 (ICH) Buffer Descriptor List
50936                              <1>       ; 17/06/2017
50937                              <1>       ; 11/06/2017
50938                              <1>       ; 28/05/2017
50939                              <1>       ; eax = dma buffer address = [audio_DMA_buff]
50940                              <1>       ; ecx = dma buffer buffer size = [audio_dmabuff_size]
50941                              <1>
50942 00011943 D1E9                <1>       shr    ecx, 1 ; dma half buffer size
50943 00011945 89CE                <1>       mov    esi, ecx
50944                              <1>
50945 00011947 BF[88610100]        <1>         mov     edi, audio_bdl_buff    ; get BDL address
50946 0001194C B910000000          <1>         mov     ecx, 32 / 2            ; make 32 entries in BDL
50947                              <1>
50948 00011951 EB05                <1>       jmp    short s_ac97_bdl1
50949                              <1>
50950                              <1> s_ac97_bdl0:
50951                              <1>       ; set buffer descriptor 0 to start of data file in memory
50952                              <1>
50953 00011953 A1[6C610100]        <1>       mov    eax, [audio_dma_buff]     ; Physical address of DMA buffer
50954                              <1>
50955                              <1> s_ac97_bdl1:
50956 00011958 AB                  <1>       stosd                     ; store dmabuffer1 address
50957                              <1>
50958 00011959 89C2                <1>       mov    edx, eax
50959                              <1>
50960                              <1> ;
50961                              <1> ; Buffer Descriptors List
50962                              <1> ; As stated earlier, each buffer descriptor list is a set of (up to) 32
50963                              <1> ; descriptors, each 8 bytes in length.  Bytes 0-3 of a descriptor entry point
50964                              <1> ; to a chunk of memory to either play from or record to.  Bytes 4-7 of an
50965                              <1> ; entry describe various control things detailed below.
50966                              <1> ;
50967                              <1> ; Buffer pointers must always be aligned on a Dword boundry.
50968                              <1> ;
50969                              <1> ;
50970                              <1>
50971                              <1> ;IOC                     equ     BIT31 ; Fire an interrupt whenever this
50972                              <1>                                        ; buffer is complete.
50973                              <1>
50974                              <1> ;BUP                     equ     BIT30  ; Buffer Underrun Policy.
50975                              <1>                                        ; if this buffer is the last buffer
50976                              <1>                                        ; in a playback, fill the remaining
50977                              <1>                                        ; samples with 0 (silence) or not.
50978                              <1>                                        ; It's a good idea to set this to 1
50979                              <1>                                        ; for the last buffer in playback,
50980                              <1>                                        ; otherwise you're likely to get a lot
50981                              <1>                                        ; of noise at the end of the sound.
50982                              <1>
50983                              <1> ;
50984                              <1> ; Bits 15:0 contain the length of the buffer, in number of samples, which
50985                              <1> ; are 16 bits each, coupled in left and right pairs, or 32bits each.
50986                              <1> ; Luckily for us, that's the same format as .wav files.
50987                              <1> ;
50988                              <1> ; A value of FFFF is 65536 samples.  Running at 44.1Khz, that's just about
50989                              <1> ; 1.5 seconds of sample time.  FFFF * 32bits is 1FFFFh bytes or 128k of data.
```

```
50990                                  <1> ;
50991                                  <1> ; A value of 0 in these bits means play no samples.
50992                                  <1> ;
50993                                  <1>
50994 0001195B 89F0                    <1>       mov    eax, esi ; DMA half buffer size
50995 0001195D 01C2                    <1>       add    edx, eax
50996 0001195F D1E8                    <1>       shr    eax, 1 ; count of 16 bit samples
50997                                  <1>       ;or    eax, IOC+BUP
50998 00011961 0D00000080              <1>       or     eax, IOC ; 11/06/2017
50999 00011966 AB                      <1>       stosd
51000                                  <1>
51001                                  <1> ; 2nd buffer:
51002                                  <1>
51003 00011967 89D0                    <1>         mov  eax, edx ; Physical address of the 2nd half of DMA buffer
51004 00011969 AB                      <1>       stosd         ; store dmabuffer2 address
51005                                  <1>
51006                                  <1> ; set length to [audio_dmabuff_size]/2
51007                                  <1> ; Set control (bits 31:16) to BUP, bits 15:0=number of samples
51008                                  <1> ;
51009 0001196A 89F0                    <1>       mov    eax, esi ; DMA half buffer size
51010 0001196C D1E8                    <1>       shr    eax, 1 ; count of 16 bit samples
51011                                  <1>       ;or    eax, IOC+BUP
51012 0001196E 0D00000080              <1>       or     eax, IOC ; 11/06/2017
51013 00011973 AB                      <1>       stosd
51014                                  <1>
51015 00011974 E2DD                    <1>       loop    s_ac97_bdl0
51016                                  <1>
51017 00011976 C3                      <1>       retn
51018                                  <1>
51019                                  <1> ac97_start_play:
51020                                  <1>       ; 28/05/2017
51021                                  <1>       ; Derived from 'playWav' procedure in 'ICHWAV.ASM'
51022                                  <1>       ; .wav player for DOS by Jeff Leyda (02/09/2002)
51023                                  <1>
51024                                  <1>       ; set output rate
51025                                  <1>       ; entry: [audio_freq] = desired sample rate
51026                                  <1>
51027 00011977 668B15[84610100]        <1>       mov          dx, [NAMBAR]
51028 0001197E 6683C22A                <1>       add          dx, CODEC_EXT_AUDIO_CTRL_REG     ; 2Ah
51029 00011982 66ED                    <1>       in           ax, dx
51030 00011984 6683C801                <1>       or     ax, 1
51031 00011988 66EF                    <1>       out    dx, ax                          ; Enable variable rate audio
51032                                  <1>
51033                                  <1>        ;call    delay1_4ms
51034                                  <1>        ;call    delay1_4ms
51035                                  <1>        ;call    delay1_4ms
51036                                  <1>        ;call    delay1_4ms
51037                                  <1>
51038 0001198A 66A1[7E610100]          <1>       mov    ax, [audio_freq]          ; sample rate
51039                                  <1>
51040 00011990 668B15[84610100]        <1>       mov          dx, [NAMBAR]
51041 00011997 6683C22C                <1>       add          dx, CODEC_PCM_FRONT_DACRATE_REG  ; 2Ch
51042 0001199B 66EF                    <1>       out    dx, ax                          ; out sample rate
51043                                  <1>
51044                                  <1>        ;call    delay1_4ms
51045                                  <1>        ;call    delay1_4ms
51046                                  <1>        ;call    delay1_4ms
51047                                  <1>        ;call    delay1_4ms
51048                                  <1>
51049                                  <1> ;
51050                                  <1> ; register reset the DMA engine.  This may cause a pop noise on the output
51051                                  <1> ; lines when the device is reset.  Prolly a better idea to mute output, then
51052                                  <1> ; reset.
51053                                  <1> ;
51054 0001199D 668B15[86610100]        <1>         mov     dx, [NABMBAR]
51055 000119A4 6683C21B                <1>         add     dx, PO_CR_REG                 ; set pointer to Cntl reg
51056 000119A8 B002                    <1>         mov     al, RR                        ; set reset
51057 000119AA EE                      <1>         out     dx, al                        ; self clearing bit
51058                                  <1> ;
51059                                  <1> ;     mov    edi, audio_bdl_buff
51060                                  <1> ;     mov    edx, [audio_dmabuff_size]
51061                                  <1> ;     shr    edx, 1
51062                                  <1> ;     mov    ecx, 32/2
51063                                  <1> ;ac97_set_bdl_buffer:
51064                                  <1> ;     ; 1st half of DMA buffer
51065                                  <1> ;     mov    eax, [audio_dma_buff]
51066                                  <1> ;     push   eax
51067                                  <1> ;     stosd
51068                                  <1> ;     mov    eax, edx ; dma buffer size / 2
51069                                  <1> ;     or     eax, IOC+BUP
51070                                  <1> ;     stosd
51071                                  <1> ;     pop    eax
51072                                  <1> ;     ; 2nd half of DMA buffer
51073                                  <1> ;     add    eax, edx
51074                                  <1> ;     stosd
51075                                  <1> ;     mov    eax, edx ; dma buffer size / 2
51076                                  <1> ;     or     eax, IOC+BUP
51077                                  <1> ;     stosd
51078                                  <1> ;     loop   ac97_set_bdl_buffer
51079                                  <1>
51080                                  <1> ; tell the DMA engine where to find our list of Buffer Descriptors.
51081                                  <1> ; this 32bit value is a flat mode memory offset (ie no segment:offset)
51082                                  <1> ;
51083                                  <1> ; write NABMBAR+10h with offset of buffer descriptor list
51084                                  <1> ;
51085 000119AB B8[88610100]           <1>         mov  eax, audio_bdl_buff
51086 000119B0 668B15[86610100]        <1>       mov    dx, [NABMBAR]
51087 000119B7 6683C210                <1>       add    dx, PO_BDBAR_REG
51088 000119BB EF                      <1>       out    dx, eax
51089                                  <1> ;
51090                                  <1> ; All set.  Let's play some music.
51091                                  <1> ;
```

```
51092                                  <1> ;
51093 000119BC B81F000000              <1>       mov    eax, 31
51094 000119C1 E816000000              <1>       call    set_ac97_LastValidIndex
51095                                  <1>
51096 000119C6 C605[80610100]01        <1>       mov    byte [audio_play_cmd], 1 ; play command (do not stop) !
51097                                  <1>
51098                                  <1> ac97_play: ; continue to play (after pause)
51099                                  <1>       ; 11/06/2017
51100                                  <1>       ; 29/05/2017
51101                                  <1>       ; 28/05/2017
51102 000119CD 668B15[86610100]        <1>         mov     dx, [NABMBAR]
51103 000119D4 6683C21B                <1>         add     dx, PO_CR_REG          ; PCM out control register
51104 000119D8 B011                    <1>         mov  al, IOCE+RPBM ; 29/05/2017
51105                                  <1>         ;mov al, 1Dh ; (Ref: KolibriOS, intelac97.asm, 'play:')
51106 000119DA EE                      <1>       out     dx, al                   ; set start!
51107                                  <1>
51108                                  <1>       ;mov   byte [audio_play_cmd], 1 ; play command (do not stop) !
51109                                  <1>
51110 000119DB C3                      <1>       retn
51111                                  <1>
51112                                  <1> ;input AL = index # to stop on
51113                                  <1> set_ac97_LastValidIndex:
51114                                  <1>       ; 28/05/2017
51115                                  <1>       ; Derived from 'setLastValidIndex' procedure in 'ICHWAV.ASM'
51116                                  <1>       ; .wav player for DOS by Jeff Leyda (02/09/2002)
51117 000119DC 668B15[86610100]        <1>       mov    dx, [NABMBAR]
51118 000119E3 6683C215                <1>       add    dx, PO_LVI_REG
51119 000119E7 EE                      <1>         out     dx, al
51120                                  <1>       ;mov   [audio_lvi], al ; for ac97_int_handler
51121 000119E8 C3                      <1>       retn
51122                                  <1>
51123                                  <1> ac97_volume:
51124                                  <1>       ; 28/05/2017
51125                                  <1>       ; bl = component (0 = master/playback/lineout volume)
51126                                  <1>       ; cl = left channel volume level (0 to 31)
51127                                  <1>       ; ch = right channel volume level (0 to 31)
51128                                  <1>
51129 000119E9 08DB                    <1>       or     bl, bl
51130 000119EB 7523                    <1>       jnz    short ac97_vol_1 ; temporary !
51131 000119ED 66B81F1F                <1>       mov    ax, 1F1Fh ; 31,31
51132 000119F1 38C1                    <1>       cmp    cl, al
51133 000119F3 771B                    <1>       ja     short ac97_vol_1 ; temporary !
51134 000119F5 38E5                    <1>       cmp    ch, ah
51135 000119F7 7717                    <1>       ja     short ac97_vol_1 ; temporary !
51136 000119F9 66890D[82610100]        <1>       mov    [audio_master_volume], cx
51137 00011A00 6629C8                  <1>       sub    ax, cx
51138 00011A03 668B15[84610100]        <1>       mov     dx, [NAMBAR]
51139 00011A0A 6683C202                <1>       add     dx, CODEC_MASTER_VOL_REG  ; 02h ; Line Out
51140 00011A0E 66EF                    <1>       out     dx, ax
51141                                  <1> ac97_vol_1:
51142 00011A10 C3                      <1>       retn
51143                                  <1>
51144                                  <1> ac97_int_handler:
51145                                  <1>       ; 17/06/2017
51146                                  <1>       ; 13/06/2017
51147                                  <1>       ; 10/06/2017, 11/06/2017
51148                                  <1>       ; Interrupt Handler for AC97 (ICH) Audio Controller
51149                                  <1>       ; Note: called by 'dev_irq_service'
51150                                  <1>       ; 28/05/2017
51151                                  <1>
51152 00011A11 50                      <1>       push   eax
51153 00011A12 52                      <1>       push   edx
51154 00011A13 51                      <1>       push   ecx
51155                                  <1>       ;push  ebx
51156 00011A14 56                      <1>       push   esi
51157 00011A15 57                      <1>       push   edi
51158                                  <1>
51159 00011A16 803D[50610100]01        <1>       cmp    byte [audio_busy], 1
51160 00011A1D 737F                    <1>       jnb    _ac97_ih3  ; busy !
51161                                  <1>
51162 00011A1F 66BA3000                <1>         mov     dx, GLOB_STS_REG
51163 00011A23 660315[86610100]        <1>         add  dx, [NABMBAR]
51164 00011A2A ED                      <1>       in     eax, dx
51165                                  <1>
51166 00011A2B 83F8FF                  <1>         cmp     eax, 0FFFFFFFFh ; -1
51167 00011A2E 7475                    <1>         je      _ac97_ih4 ; exit
51168                                  <1>
51169 00011A30 A940000000              <1>         test    eax, 40h ; PCM Out Interrupt
51170 00011A35 7507                    <1>         jnz     short _ac97_ih0
51171                                  <1>
51172 00011A37 85C0                    <1>       test   eax, eax
51173 00011A39 746A                    <1>       jz     short _ac97_ih4 ; exit
51174                                  <1>
51175                                  <1>       ;mov   dx, GLOB_STS_REG
51176                                  <1>         ;add dx, [NABMBAR]
51177 00011A3B EF                      <1>       out    dx, eax
51178                                  <1>
51179 00011A3C EB67                    <1>       jmp    short _ac97_ih4 ; exit
51180                                  <1>
51181                                  <1> _ac97_ih0:
51182 00011A3E 50                      <1>       push   eax
51183                                  <1>       ;
51184 00011A3F C605[50610100]01        <1>       mov    byte [audio_busy], 1
51185                                  <1>
51186                                  <1>       ;mov   al, 10h
51187                                  <1>       ;mov   dx, PO_CR_REG
51188                                  <1>         ;add dx, [NABMBAR]
51189                                  <1>       ;out   dx, al
51190                                  <1>
51191 00011A46 66B81C00                <1>       mov    ax, 1Ch ; FIFOE(=16)+BCIS(=8)+LVBCI(=4)
51192 00011A4A 66BA1600                <1>       mov    dx, PO_SR_REG
51193 00011A4E 660315[86610100]        <1>         add  dx, [NABMBAR]
```

```
51194 00011A55 66EF                <1>       out    dx, ax
51195                              <1>
51196 00011A57 66BA1400            <1>       mov    dx, PO_CIV_REG
51197 00011A5B 660315[86610100]    <1>         add  dx, [NABMBAR]
51198 00011A62 EC                  <1>       in     al, dx
51199                              <1>
51200                              <1>       ;cmp   al, [audio_civ] ; [audio_flag]
51201                              <1>       ;je    short _ac97_ih2
51202                              <1>
51203 00011A63 A2[81610100]        <1>       mov    [audio_civ], al
51204 00011A68 FEC8                <1>       dec    al
51205                              <1>       ;inc   al ; 11/06/2017
51206 00011A6A 241F                <1>       and    al, 1Fh
51207                              <1>
51208 00011A6C 66BA1500            <1>         mov     dx, PO_LVI_REG
51209 00011A70 660315[86610100]    <1>         add  dx, [NABMBAR]
51210 00011A77 EE                  <1>         out  dx, al
51211                              <1>
51212                              <1>       ;mov   al, [audio_civ]
51213                              <1>       ;inc   al
51214                              <1>       ;and   al, 1
51215                              <1>       ;mov   [audio_flag], al
51216                              <1>       ;; [audio_flag] : 0 = Buffer 1, 1 = Buffer 2
51217                              <1>
51218 00011A78 E82E000000          <1>       call   ac97_tuneLoop
51219                              <1>
51220                              <1>       ; 13/06/2017
51221 00011A7D 30C0                <1>       xor    al, al ; 0
51222 00011A7F 3805[74610100]      <1>       cmp    [audio_flag], al ; 0
51223 00011A85 7702                <1>       ja     short _ac97_ih1
51224 00011A87 FEC0                <1>       inc    al ; 1
51225                              <1> _ac97_ih1:
51226 00011A89 A2[74610100]        <1>       mov    [audio_flag], al
51227                              <1> _ac97_ih2:
51228 00011A8E 58                  <1>       pop    eax
51229                              <1>       ;
51230 00011A8F 83E040              <1>       and     eax, 40h
51231 00011A92 668B15[86610100]    <1>         mov  dx, [NABMBAR]
51232 00011A99 6683C230            <1>       add    dx, GLOB_STS_REG
51233 00011A9D EF                  <1>       out    dx, eax
51234                              <1>
51235                              <1>       ;; 13/06/2017
51236                              <1>       ;mov   al, 11h ; IOCE + RPBM
51237                              <1>       ;dx, PO_CR_REG
51238                              <1>         ;add dx, [NABMBAR]
51239                              <1>       ;out   dx, al
51240                              <1>
51241                              <1> _ac97_ih3:
51242 00011A9E C605[50610100]00    <1>       mov    byte [audio_busy], 0
51243                              <1> _ac97_ih4:
51244 00011AA5 5F                  <1>       pop    edi
51245 00011AA6 5E                  <1>       pop    esi
51246                              <1>       ;pop   ebx
51247 00011AA7 59                  <1>       pop    ecx
51248 00011AA8 5A                  <1>       pop    edx
51249 00011AA9 58                  <1>       pop    eax
51250                              <1>
51251 00011AAA C3                  <1>       retn
51252                              <1>
51253                              <1> ac97_tuneLoop:
51254                              <1>       ; 13/06/2017
51255                              <1>       ; 10/06/2017
51256                              <1>       ; 28/05/2017
51257 00011AAB 803D[80610100]01    <1>       cmp    byte [audio_play_cmd], 1
51258 00011AB2 7230                <1>       jb     short _ac97_tlp3 ; stop command !
51259                              <1>
51260 00011AB4 8B3D[6C610100]      <1>       mov    edi, [audio_dma_buff]
51261 00011ABA 8B0D[70610100]      <1>       mov    ecx, [audio_dmabuff_size]
51262 00011AC0 D1E9                <1>       shr    ecx, 1 ; dma buff size / 2 = half buffer size
51263                              <1>
51264                              <1>       ; 13/06/2017
51265 00011AC2 F605[74610100]01    <1>       test   byte [audio_flag], 1
51266 00011AC9 7402                <1>       jz     short _ac97_tlp1
51267                              <1>       ;test  byte [audio_civ], 1  ; Current (Buff) Index
51268                              <1>       ;jnz   short _ac97_tlp1  ; Next: Buffer 1
51269                              <1>       ;;jz   short _ac97_tlp1  ; Next: Buffer 1 ; 10/06/2017
51270                              <1>       ; Next: Buffer 2
51271 00011ACB 01CF                <1>       add    edi, ecx
51272                              <1> _ac97_tlp1:
51273 00011ACD 8B35[64610100]      <1>       mov    esi, [audio_p_buffer] ; phy addr of audio buff
51274                              <1>       ;rep   movsb
51275 00011AD3 C1E902              <1>       shr    ecx, 2 ; half buff size / 4
51276 00011AD6 F3A5                <1>       rep    movsd
51277                              <1> _ac97_tlp2:
51278 00011AD8 C3                  <1>       retn
51279                              <1>
51280                              <1> ac97_pause:
51281                              <1>       ; 11/06/2017
51282                              <1>       ; 29/05/2017
51283 00011AD9 B010                <1>       mov    al, IOCE
51284 00011ADB EB21                <1>       jmp    short ac97_po_cmd
51285                              <1>
51286                              <1> ac97_stop:
51287                              <1>       ; 28/05/2017
51288 00011ADD C605[80610100]00    <1>       mov    byte [audio_play_cmd], 0 ; stop !
51289                              <1> _ac97_tlp3:
51290                              <1>       ; 29/05/2017
51291                              <1>       ;mov   dx, [NABMBAR]
51292                              <1>       ;add   dx, PO_CR_REG
51293                              <1>       ;mov   al, 0
51294                              <1>       ;out   dx, al
51295                              <1>
```

```
51296                                  <1>       ; 11/06/2017
51297 00011AE4 30C0                    <1>       xor    al, al ; 0
51298 00011AE6 E813000000              <1>       call   ac97_po_cmd
51299                                  <1>
51300                                  <1>       ; (Ref: KolibriOS, intelac97.asm, 'stop:')
51301                                  <1>       ; Clear FIFOE, BCIS, LVBCI (Ref: Intel ICH hub manual)
51302 00011AEB 66B81C00                <1>       mov     ax, 1Ch
51303 00011AEF 668B15[86610100]        <1>       mov     dx, [NABMBAR]
51304 00011AF6 6683C216                <1>       add     dx, PO_SR_REG
51305 00011AFA 66EF                    <1>       out     dx, ax
51306                                  <1>
51307                                  <1>       ;retn
51308                                  <1>
51309                                  <1>       ; 11/06/2017
51310 00011AFC B002                    <1>       mov     al, RR
51311                                  <1> ac97_po_cmd:
51312                                  <1>        ;11/06/2017
51313                                  <1>       ; 29/05/2017
51314 00011AFE 668B15[86610100]        <1>       mov     dx, [NABMBAR]
51315 00011B05 6683C21B                <1>         add     dx, PO_CR_REG          ; PCM out control register
51316 00011B09 EE                      <1>       out    dx, al
51317 00011B0A C3                      <1>       retn
51318                                  <1>
51319                                  <1> reset_ac97_controller:
51320                                  <1>       ; 10/06/2017
51321                                  <1>       ; 29/05/2017
51322                                  <1>       ; 28/05/2017
51323                                  <1>       ; reset AC97 audio controller registers
51324 00011B0B 31C0                    <1>       xor     eax, eax
51325 00011B0D 66BA0B00                <1>         mov  dx, PI_CR_REG
51326 00011B11 660315[86610100]        <1>       add    dx, [NABMBAR]
51327 00011B18 EE                      <1>       out     dx, al
51328                                  <1>
51329 00011B19 66BA1B00                <1>         mov     dx, PO_CR_REG
51330 00011B1D 660315[86610100]        <1>       add    dx, [NABMBAR]
51331 00011B24 EE                      <1>       out     dx, al
51332                                  <1>
51333 00011B25 66BA2B00                <1>         mov     dx, MC_CR_REG
51334 00011B29 660315[86610100]        <1>       add    dx, [NABMBAR]
51335 00011B30 EE                      <1>       out     dx, al
51336                                  <1>
51337 00011B31 B002                    <1>         mov     al, RR
51338 00011B33 66BA0B00                <1>         mov     dx, PI_CR_REG
51339 00011B37 660315[86610100]        <1>       add    dx, [NABMBAR]
51340 00011B3E EE                      <1>       out     dx, al
51341                                  <1>
51342 00011B3F 66BA1B00                <1>         mov     dx, PO_CR_REG
51343 00011B43 660315[86610100]        <1>       add    dx, [NABMBAR]
51344 00011B4A EE                      <1>       out     dx, al
51345                                  <1>
51346 00011B4B 66BA2B00                <1>         mov     dx, MC_CR_REG
51347 00011B4F 660315[86610100]        <1>       add    dx, [NABMBAR]
51348 00011B56 EE                      <1>       out     dx, al
51349                                  <1>
51350 00011B57 C3                      <1>       retn
51351                                  <1>
51352                                  <1> ac97_reset:
51353                                  <1>       ; 10/06/2017
51354                                  <1>       ; 29/05/2017
51355                                  <1>       ; 28/05/2017
51356 00011B58 E8AEFFFFFF              <1>       call   reset_ac97_controller
51357                                  <1>       ; 29/05/2017
51358                                  <1>       ;jmp   reset_ac97_codec
51359                                  <1> reset_ac97_codec:
51360                                  <1>       ; 28/05/2017 - Erdogan Tan (Ref: KolibriOS, intelac97.asm)
51361 00011B5D 66BA2C00                <1>       mov    dx, GLOB_CNT_REG ; 2Ch
51362 00011B61 660315[86610100]        <1>       add    dx, [NABMBAR]
51363 00011B68 ED                      <1>       in     eax, dx
51364                                  <1>
51365 00011B69 A902000000              <1>       test   eax, 2
51366 00011B6E 7407                    <1>       jz     short _r_ac97codec_cold
51367                                  <1>
51368 00011B70 E80F000000              <1>       call   warm_ac97codec_reset
51369 00011B75 7308                    <1>       jnc    short _r_ac97codec_ok
51370                                  <1> _r_ac97codec_cold:
51371 00011B77 E83D000000              <1>         call    cold_ac97codec_reset
51372 00011B7C 7301                    <1>         jnc     short _r_ac97codec_ok
51373                                  <1>
51374                                  <1>       ; 16/04/2017
51375                                  <1>         ;xor     eax, eax         ; timeout error
51376                                  <1>             ;stc
51377 00011B7E C3                      <1>       retn
51378                                  <1>
51379                                  <1> _r_ac97codec_ok:
51380 00011B7F 31C0                    <1>         xor     eax, eax
51381                                  <1>         ;mov al, VIA_ACLINK_C00_READY ; 1
51382 00011B81 FEC0                    <1>         inc  al
51383 00011B83 C3                      <1>       retn
51384                                  <1>
51385                                  <1> warm_ac97codec_reset:
51386                                  <1>       ; 28/05/2017 - Erdogan Tan (Ref: KolibriOS, intelac97.asm)
51387 00011B84 B806000000              <1>         mov     eax, 6
51388 00011B89 66BA2C00                <1>       mov    dx, GLOB_CNT_REG ; 2Ch
51389 00011B8D 660315[86610100]        <1>       add    dx, [NABMBAR]
51390 00011B94 EF                      <1>       out    dx, eax
51391                                  <1>
51392 00011B95 B90A000000              <1>       mov    ecx, 10      ; total 1s
51393                                  <1> _warm_ac97c_rst_wait:
51394 00011B9A 51                      <1>       push   ecx
51395 00011B9B E8AEF5FFFF              <1>       call   delay_100ms
51396 00011BA0 59                      <1>       pop    ecx
51397                                  <1>
```

```
51398 00011BA1 66BA3000            <1>       mov    dx, GLOB_STS_REG ; 30h
51399 00011BA5 660315[86610100]    <1>       add    dx, [NABMBAR]
51400 00011BAC ED                  <1>       in     eax, dx
51401                              <1>
51402 00011BAD A900030010          <1>       test   eax, CTRL_ST_CREADY
51403 00011BB2 7504                <1>       jnz    short _warm_ac97c_rst_ok
51404                              <1>
51405 00011BB4 49                  <1>         dec     ecx
51406 00011BB5 75E3                <1>         jnz     short _warm_ac97c_rst_wait
51407                              <1>
51408                              <1> _warm_ac97c_rst_fail:
51409 00011BB7 F9                  <1>         stc
51410                              <1> _warm_ac97c_rst_ok:
51411 00011BB8 C3                  <1>       retn
51412                              <1>
51413                              <1> cold_ac97codec_reset:
51414                              <1>       ; 28/05/2017 - Erdogan Tan (Ref: KolibriOS, intelac97.asm)
51415 00011BB9 B802000000          <1>         mov     eax, 2
51416 00011BBE 66BA2C00            <1>       mov    dx, GLOB_CNT_REG ; 2Ch
51417 00011BC2 660315[86610100]    <1>       add    dx, [NABMBAR]
51418 00011BC9 EF                  <1>       out    dx, eax
51419                              <1>
51420 00011BCA E87FF5FFFF          <1>       call   delay_100ms  ; wait 100 ms
51421 00011BCF E87AF5FFFF          <1>       call   delay_100ms  ; wait 100 ms
51422 00011BD4 E875F5FFFF          <1>       call   delay_100ms  ; wait 100 ms
51423 00011BD9 E870F5FFFF          <1>       call   delay_100ms  ; wait 100 ms
51424                              <1>
51425 00011BDE B910000000          <1>       mov    ecx, 16      ; total 20*100 ms = 2s
51426                              <1> _cold_ac97c_rst_wait:
51427 00011BE3 66BA3000            <1>       mov    dx, GLOB_STS_REG ; 30h
51428 00011BE7 660315[86610100]    <1>       add    dx, [NABMBAR]
51429 00011BEE ED                  <1>       in     eax, dx
51430                              <1>
51431 00011BEF A900030010          <1>       test   eax, CTRL_ST_CREADY
51432 00011BF4 750B                <1>       jnz    short _cold_ac97c_rst_ok
51433                              <1>
51434 00011BF6 51                  <1>       push   ecx
51435 00011BF7 E852F5FFFF          <1>       call   delay_100ms
51436 00011BFC 59                  <1>       pop    ecx
51437                              <1>
51438 00011BFD 49                  <1>         dec     ecx
51439 00011BFE 75E3                <1>         jnz     short _cold_ac97c_rst_wait
51440                              <1>
51441                              <1> _cold_ac97c_rst_fail:
51442 00011C00 F9                  <1>         stc
51443                              <1> _cold_ac97c_rst_ok:
51444 00011C01 C3                  <1>       retn
51445                              <1>
51446                              <1> sb16_current_sound_data:
51447                              <1>       ; 24/06/2017
51448                              <1>       ; 22/06/2017
51449                              <1>       ; get current sound (PCM out) data for graphics
51450                              <1>       ; (for Sound Blaster 16)
51451                              <1>       ; ebx = Physical address (on page boundary)
51452                              <1>       ; ecx = Byte count
51453                              <1>       ; [audio_buff_size]
51454                              <1>
51455                              <1>       ;;mov  edi, [audio_buff_size]
51456                              <1>       ;mov   edi, [audio_dmabuff_size]
51457                              <1>       ;mov   esi, [audio_dma_buff]
51458 00011C02 39CF                <1>       cmp    edi, ecx
51459 00011C04 7302                <1>       jnb    short sb16_gcd_1
51460 00011C06 89F9                <1>       mov    ecx, edi
51461                              <1> sb16_gcd_1:
51462 00011C08 E403                <1>       in     al, 03h
51463 00011C0A 88C2                <1>       mov    dl, al
51464 00011C0C E403                <1>       in     al, 03h
51465 00011C0E 88C6                <1>       mov     dh, al
51466 00011C10 0FB7C2              <1>       movzx  eax, dx
51467 00011C13 EB41                <1>       jmp    short sb16_gcd_2
51468                              <1> ;sb16_gcd_2:
51469                              <1> ;     cmp    eax, ecx
51470                              <1> ;     jnb    short sb16_gcd_3
51471                              <1> ;     ; remain count < graphics bytes
51472                              <1> ;     mov    eax, ecx ; fix remain count to data size
51473                              <1> ;sb16_gcd_3:
51474                              <1> ;     sub    edi, eax
51475                              <1> ;     jna    short sb16_gcd_4
51476                              <1> ;     add    esi, edi ; dma buffer offset
51477                              <1> ;sb16_gcd_4:
51478                              <1> ;     mov    edi, ebx ; buffer address (for graphics)
51479                              <1> ;     mov    [u.r0], ecx
51480                              <1> ;     rep    movsb
51481                              <1> ;     retn
51482                              <1>
51483                              <1> get_current_sound_data:
51484                              <1>       ; 24/06/2017
51485                              <1>       ; 22/06/2017
51486                              <1>       ; get current sound (PCM out) data for graphics
51487                              <1>       ;
51488                              <1>       ; ebx = Physical address (on page boundary)
51489                              <1>       ; ecx = Byte count
51490                              <1>       ; [audio_buff_size]
51491                              <1>
51492                              <1>       ;mov   edi, [audio_buff_size]
51493 00011C15 8B3D[70610100]      <1>       mov    edi, [audio_dmabuff_size]
51494 00011C1B 8B35[6C610100]      <1>       mov    esi, [audio_dma_buff]
51495 00011C21 803D[4D610100]02    <1>       cmp    byte [audio_device], 2
51496 00011C28 72D8                <1>       jb     short sb16_current_sound_data ; = 1
51497 00011C2A D1EF                <1>       shr    edi, 1
51498 00011C2C 39CF                <1>       cmp    edi, ecx
51499 00011C2E 7302                <1>       jnb    short gcd_0
```

```
51500 00011C30 89F9                <1>       mov    ecx, edi
51501                              <1> gcd_0:
51502 00011C32 803D[4D610100]03    <1>       cmp    byte [audio_device], 3
51503 00011C39 7232                <1>       jb     short ac97_current_sound_data ; = 2
51504                              <1>       ; = 3
51505                              <1> vt8233_current_sound_data:
51506                              <1>       ; 22/06/2017
51507                              <1>       ; 21/06/2017
51508                              <1>       ; get current sound (PCM out) data for graphics
51509                              <1>       ; (for VT 8233, VT 8237R)
51510                              <1>       ; ebx = Physical address (on page boundary)
51511                              <1>       ; ecx = Byte count
51512                              <1>       ; [audio_buff_size]
51513                              <1>
51514                              <1>       ;;mov  edi, [audio_buff_size]
51515                              <1>       ;mov   edi, [audio_dmabuff_size]
51516                              <1>       ;mov   esi, [audio_dma_buff]
51517                              <1>       ;shr   edi, 1
51518                              <1>       ;cmp   edi, ecx
51519                              <1>       ;jnb   short vt8233_gcd_1
51520                              <1>       ;mov   ecx, edi
51521                              <1> vt8233_gcd_1:
51522 00011C3B BA0C000000         <1>       mov    edx, VIA_REG_OFFSET_CURR_COUNT
51523 00011C40 E869F5FFFF         <1>       call   ctrl_io_r32
51524 00011C45 89C2               <1>       mov    edx, eax ; remain count (bits 23-0),
51525                              <1>                       ; SGD index (bits 31-24)
51526 00011C47 81E200000001       <1>       and    edx, 1000000h ; SGD index (0 = 1st half)
51527 00011C4D 7402               <1>       jz     short vt8233_gcd_2
51528                              <1>       ; the second half of DMA buffer
51529 00011C4F 01FE               <1>       add    esi, edi
51530                              <1> vt8233_gcd_2:
51531 00011C51 25FFFFFF00         <1>       and    eax, 0FFFFFFh ; bits 23-0
51532                              <1> ac97_gcd_2:
51533                              <1> sb16_gcd_2:
51534 00011C56 39C8               <1>       cmp    eax, ecx
51535 00011C58 7302               <1>       jnb    short vt8233_gcd_3
51536                              <1>       ; remain count < graphics bytes
51537 00011C5A 89C8               <1>       mov    eax, ecx ; fix remain count to data size
51538                              <1> vt8233_gcd_3:
51539 00011C5C 29C7               <1>       sub    edi, eax
51540 00011C5E 7602               <1>       jna    short vt8233_gcd_4
51541 00011C60 01FE               <1>       add    esi, edi ; dma buffer offset
51542                              <1> vt8233_gcd_4:
51543 00011C62 89DF               <1>       mov    edi, ebx ; buffer address (for graphics)
51544 00011C64 890D[64030600]     <1>       mov    [u.r0], ecx
51545 00011C6A F3A4               <1>       rep    movsb
51546                              <1> vt8233_gcd_5:
51547 00011C6C C3                 <1>       retn
51548                              <1>
51549                              <1> ac97_current_sound_data:
51550                              <1>       ; 23/06/2017
51551                              <1>       ; 22/06/2017
51552                              <1>       ; get current sound (PCM out) data for graphics
51553                              <1>       ; (for AC'97, ICH)
51554                              <1>       ; ebx = Physical address (on page boundary)
51555                              <1>       ; ecx = Byte count
51556                              <1>       ; [audio_buff_size]
51557                              <1>
51558                              <1>       ;;mov  edi, [audio_buff_size]
51559                              <1>       ;mov   edi, [audio_dmabuff_size]
51560                              <1>       ;mov   esi, [audio_dma_buff]
51561                              <1>       ;shr   edi, 1
51562                              <1>       ;cmp   edi, ecx
51563                              <1>       ;jnb   short ac97_gcd_0
51564                              <1>       ;mov   ecx, edi
51565                              <1> ac97_gcd_0:
51566 00011C6D 66BA1400           <1>       mov    dx, PO_CIV_REG ; Position In Current Buff Reg
51567 00011C71 660315[86610100]    <1>       add    dx, [NABMBAR]
51568 00011C78 EC                 <1>       in     al, dx ; current index value
51569 00011C79 A801               <1>       test   al, 1
51570 00011C7B 7402               <1>       jz     short ac97_gcd_1
51571 00011C7D 01FE               <1>       add    esi, edi
51572                              <1> ac97_gcd_1:
51573 00011C7F 31C0               <1>       xor    eax, eax
51574 00011C81 66BA1800           <1>       mov    dx, PO_PICB_REG ; Position In Current Buff Reg
51575 00011C85 660315[86610100]    <1>       add    dx, [NABMBAR]
51576 00011C8C 66ED               <1>       in     ax, dx ; remain dwords
51577 00011C8E C1E002             <1>       shl    eax, 2 ; remain bytes ; 23/06/2017
51578 00011C91 EBC3               <1>       jmp    short ac97_gcd_2
51579                              <1> ;     cmp    eax, ecx
51580                              <1> ;     jnb    short ac97_gcd_2
51581                              <1> ;     ; remain count < graphics bytes
51582                              <1> ;     mov    eax, ecx ; fix remain count to data size
51583                              <1> ;ac97_gcd_2:
51584                              <1> ;     sub    edi, eax
51585                              <1> ;     jna    short ac97_gcd_3
51586                              <1> ;     add    esi, edi ; dma buffer offset
51587                              <1> ;ac97_gcd_3:
51588                              <1> ;     mov    edi, ebx ; buffer address (for graphics)
51589                              <1> ;     mov    [u.r0], ecx
51590                              <1> ;     rep    movsb
51591                              <1> ;     retn
51592                              <1>
51593                              <1> sb16_get_dma_buff_off:
51594                              <1>       ; 24/06/2017
51595                              <1>       ; 22/06/2017
51596                              <1>       ; get current (PCM OUT DMA buffer) pointer
51597                              <1>       ; (for Sound Blaster 16)
51598                              <1>
51599                              <1>       ;mov   ecx, [audio_dmabuff_size]
51600                              <1>       ;xor   ebx, ebx
51601                              <1>       ;shr   ecx, 1
```

```
51602                                  <1> sb16_gdmabo_0:
51603 00011C93 E403                    <1>       in     al, 03h
51604 00011C95 88C2                    <1>       mov    dl, al
51605 00011C97 E403                    <1>       in     al, 03h
51606 00011C99 88C6                    <1>       mov     dh, al
51607 00011C9B 0FB7C2                  <1>       movzx  eax, dx
51608 00011C9E EB30                    <1>       jmp    short sb16_gdmabo_1
51609                                  <1>
51610                                  <1> get_dma_buffer_offset:
51611                                  <1>       ; 24/06/2017
51612                                  <1>       ; 22/06/2017
51613                                  <1>       ; get current sound (PCM out) data for graphics
51614                                  <1>       ;
51615                                  <1>       ; ebx = Physical address (on page boundary)
51616                                  <1>       ; ecx = Byte count
51617                                  <1>       ; [audio_buff_size]
51618                                  <1>
51619 00011CA0 8B0D[70610100]          <1>       mov    ecx, [audio_dmabuff_size]
51620 00011CA6 31DB                    <1>       xor    ebx, ebx
51621                                  <1> gdmabo_0:
51622 00011CA8 803D[4D610100]02        <1>       cmp    byte [audio_device], 2
51623 00011CAF 72E2                    <1>       jb     short sb16_get_dma_buff_off
51624 00011CB1 742A                    <1>       je     short ac97_get_dma_buff_off
51625                                  <1>
51626                                  <1> vt8233_get_dma_buff_off:
51627                                  <1>       ; 24/06/2017
51628                                  <1>       ; 22/06/2017
51629                                  <1>       ; get current (PCM OUT DMA buffer) pointer
51630                                  <1>       ; (for VT 8233, VT 8237R)
51631                                  <1>
51632                                  <1>       ;mov   ecx, [audio_dmabuff_size]
51633                                  <1>       ;xor   ebx, ebx
51634 00011CB3 D1E9                    <1>       shr    ecx, 1
51635                                  <1> vt8233_gdmabo_0:
51636 00011CB5 BA0C000000              <1>       mov    edx, VIA_REG_OFFSET_CURR_COUNT
51637 00011CBA E8EFF4FFFF              <1>       call   ctrl_io_r32
51638 00011CBF 89C2                    <1>       mov    edx, eax ; remain count (bits 23-0),
51639                                  <1>                       ; SGD index (bits 31-24)
51640 00011CC1 81E200000001            <1>       and    edx, 1000000h ; SGD index (0 = 1st half)
51641 00011CC7 7402                    <1>       jz     short vt8233_gdmabo_1
51642                                  <1>       ; the second half of DMA buffer
51643 00011CC9 89CB                    <1>       mov    ebx, ecx
51644                                  <1> vt8233_gdmabo_1:
51645 00011CCB 25FFFFFF00              <1>       and    eax, 0FFFFFFh ; bits 23-0
51646                                  <1> sb16_gdmabo_1:
51647                                  <1> ac97_gdmabo_2:
51648 00011CD0 29C1                    <1>       sub    ecx, eax
51649 00011CD2 7602                    <1>       jna    short vt8233_gdmabo_2
51650 00011CD4 01CB                    <1>       add    ebx, ecx ; dma buffer offset
51651                                  <1> vt8233_gdmabo_2:
51652 00011CD6 891D[64030600]          <1>       mov    [u.r0], ebx
51653 00011CDC C3                      <1>       retn
51654                                  <1>
51655                                  <1> ac97_get_dma_buff_off:
51656                                  <1>       ; 24/06/2017
51657                                  <1>       ; 22/06/2017
51658                                  <1>       ; get current (PCM OUT DMA buffer) pointer
51659                                  <1>       ; (for AC'97, ICH)
51660                                  <1>       ; ebx = Physical address (on page boundary)
51661                                  <1>       ; ecx = Byte count
51662                                  <1>       ; [audio_buff_size]
51663                                  <1>
51664                                  <1>       ;mov   ecx, [audio_dmabuff_size]
51665                                  <1>       ;xor   ebx, ebx
51666 00011CDD D1E9                    <1>       shr    ecx, 1
51667                                  <1> ac97_gdmabo_0:
51668 00011CDF 66BA1400                <1>       mov    dx, PO_CIV_REG ; Position In Current Buff Reg
51669 00011CE3 660315[86610100]        <1>       add    dx, [NABMBAR]
51670 00011CEA EC                      <1>       in     al, dx ; current index value
51671 00011CEB A801                    <1>       test   al, 1
51672 00011CED 7402                    <1>       jz     short ac97_gdmabo_1
51673 00011CEF 89CB                    <1>       mov    ebx, ecx
51674                                  <1> ac97_gdmabo_1:
51675 00011CF1 31C0                    <1>       xor    eax, eax
51676 00011CF3 66BA1800                <1>       mov    dx, PO_PICB_REG ; Position In Current Buff Reg
51677 00011CF7 660315[86610100]        <1>       add    dx, [NABMBAR]
51678 00011CFE 66ED                    <1>       in     ax, dx ; remain dwords
51679 00011D00 EBCE                    <1>       jmp    short ac97_gdmabo_2
51680
51681 00011D02 90<rept>                    align 4
51682
51683                                      %include 'vgadata.s' ; 04/07/2016
51684                                  <1> ;
*******************************************************************************
51685                                  <1> ; TRDOS386.ASM (TRDOS 386 Kernel) - v2.0.0 - vgadata.s  (palette and fond data)
51686                                  <1> ; ----------------------------------------------------------------------------
51687                                  <1> ; Last Update: 04/07/2016
51688                                  <1> ; ----------------------------------------------------------------------------
51689                                  <1> ; Beginning: 16/01/2016
51690                                  <1> ; ----------------------------------------------------------------------------
51691                                  <1> ; Assembler: NASM version 2.11 (trdos386.s)
51692                                  <1> ; ----------------------------------------------------------------------------
51693                                  <1> ; Turkish Rational DOS
51694                                  <1> ; Operating System Project v2.0 by ERDOGAN TAN (Beginning: 04/01/2016)
51695                                  <1> ;
51696                                  <1> ; Derived from 'Plex86/Bochs VGABios' source code, vgabios-0.7a (2011)
51697                                  <1> ; by the LGPL VGABios Developers Team (2001-2008), 'vgatables.h'
51698                                  <1> ;
51699                                  <1> ; Oracle VirtualBox 5.0.24 VGABios Source Code
51700                                  <1> ; ('vgabios.c', 'vgatables.h', 'vgafonts.h', 'vgarom.asm')
51701                                  <1> ;
51702                                  <1> ; Palette and font data in assembly language format:
```

```
 51703                                  <1> ; 'VBoxVgaBiosAlternative.asm'
 51704                                  <1>
 51705                                  <1> ;
********************************************************************************************
 51706                                  <1>
 51707                                  <1> ; 04/07/2016
 51708                                  <1> ; COLOR DATA
 51709                                  <1>
 51710                                  <1> palette0:
 51711 00011D04 00000000000000000-      <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 51712 00011D0D 00000000000000          <1>
 51713 00011D14 0000000000000002A-      <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah,
02ah, 02ah
 51714 00011D1D 2A2A2A2A2A2A            <1>
 51715 00011D24 2A2A2A2A2A2A2A2A2A-      <1>     db  02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah,
02ah
 51716 00011D2D 2A2A2A2A2A2A            <1>
 51717 00011D34 2A2A2A2A2A2A2A2A2A-      <1>     db  02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah,
02ah
 51718 00011D3D 2A2A2A2A2A2A            <1>
 51719 00011D44 2A2A2A2A2A2A2A2A3F-      <1>     db  02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh,
03fh
 51720 00011D4D 3F3F3F3F3F3F            <1>
 51721 00011D54 3F3F3F3F3F3F3F3F3F-      <1>     db  03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh,
03fh
 51722 00011D5D 3F3F3F3F3F3F            <1>
 51723 00011D64 0000000000000000000-      <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 51724 00011D6D 00000000000000          <1>
 51725 00011D74 0000000000000002A-      <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah,
02ah
 51726 00011D7D 2A2A2A2A2A2A            <1>
 51727 00011D84 2A2A2A2A2A2A2A2A2A-      <1>     db  02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah,
02ah
 51728 00011D8D 2A2A2A2A2A2A            <1>
 51729 00011D94 2A2A2A2A2A2A2A2A2A-      <1>     db  02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah,
02ah
 51730 00011D9D 2A2A2A2A2A2A            <1>
 51731 00011DA4 2A2A2A2A2A2A2A2A3F-      <1>     db  02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh,
03fh
 51732 00011DAD 3F3F3F3F3F3F            <1>
 51733 00011DB4 3F3F3F3F3F3F3F3F3F-      <1>     db  03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh,
03fh
 51734 00011DBD 3F3F3F3F3F3F            <1>
 51735                                  <1> palette1:
 51736 00011DC4 00000000002A002A00-      <1>     db  000h, 000h, 000h, 000h, 000h, 02ah, 000h, 02ah, 000h, 000h, 02ah, 02ah, 02ah, 000h, 000h,
02ah
 51737 00011DCD 002A2A2A00002A          <1>
 51738 00011DD4 002A2A15002A2A2A00-      <1>     db  000h, 02ah, 02ah, 015h, 000h, 02ah, 02ah, 02ah, 000h, 000h, 000h, 000h, 000h, 02ah, 000h,
02ah
 51739 00011DDD 000000002A002A          <1>
 51740 00011DE4 00002A2A2A00002A00-      <1>     db  000h, 000h, 02ah, 02ah, 02ah, 000h, 000h, 02ah, 000h, 02ah, 02ah, 015h, 000h, 02ah, 02ah,
02ah
 51741 00011DED 2A2A15002A2A2A          <1>
 51742 00011DF4 15151515153F153F15-      <1>     db  015h, 015h, 015h, 015h, 015h, 03fh, 015h, 03fh, 015h, 015h, 03fh, 03fh, 03fh, 015h, 015h,
03fh
 51743 00011DFD 153F3F3F15153F          <1>
 51744 00011E04 153F3F3F153F3F3F15-      <1>     db  015h, 03fh, 03fh, 03fh, 015h, 03fh, 03fh, 03fh, 015h, 015h, 015h, 015h, 015h, 03fh, 015h,
03fh
 51745 00011E0D 151515153F153F          <1>
 51746 00011E14 15153F3F3F15153F15-      <1>     db  015h, 015h, 03fh, 03fh, 03fh, 015h, 015h, 03fh, 015h, 03fh, 03fh, 03fh, 015h, 03fh, 03fh,
03fh
 51747 00011E1D 3F3F3F153F3F3F          <1>
 51748 00011E24 00000000002A002A00-      <1>     db  000h, 000h, 000h, 000h, 000h, 02ah, 000h, 02ah, 000h, 000h, 02ah, 02ah, 02ah, 000h, 000h,
02ah
 51749 00011E2D 002A2A2A00002A          <1>
 51750 00011E34 002A2A15002A2A2A00-      <1>     db  000h, 02ah, 02ah, 015h, 000h, 02ah, 02ah, 02ah, 000h, 000h, 000h, 000h, 000h, 02ah, 000h,
02ah
 51751 00011E3D 000000002A002A          <1>
 51752 00011E44 00002A2A2A00002A00-      <1>     db  000h, 000h, 02ah, 02ah, 02ah, 000h, 000h, 02ah, 000h, 02ah, 02ah, 015h, 000h, 02ah, 02ah,
02ah
 51753 00011E4D 2A2A15002A2A2A          <1>
 51754 00011E54 15151515153F153F15-      <1>     db  015h, 015h, 015h, 015h, 015h, 03fh, 015h, 03fh, 015h, 015h, 03fh, 03fh, 03fh, 015h, 015h,
03fh
 51755 00011E5D 153F3F3F15153F          <1>
 51756 00011E64 153F3F3F153F3F3F15-      <1>     db  015h, 03fh, 03fh, 03fh, 015h, 03fh, 03fh, 03fh, 015h, 015h, 015h, 015h, 015h, 03fh, 015h,
03fh
 51757 00011E6D 151515153F153F          <1>
 51758 00011E74 15153F3F3F15153F15-      <1>     db  015h, 015h, 03fh, 03fh, 03fh, 015h, 015h, 03fh, 015h, 03fh, 03fh, 03fh, 015h, 03fh, 03fh,
03fh
 51759 00011E7D 3F3F3F153F3F3F          <1>
 51760                                  <1> palette2:
 51761 00011E84 00000000002A002A00-      <1>     db  000h, 000h, 000h, 000h, 000h, 02ah, 000h, 02ah, 000h, 000h, 02ah, 02ah, 02ah, 000h, 000h,
02ah
 51762 00011E8D 002A2A2A00002A          <1>
 51763 00011E94 002A2A2A002A2A2A00-      <1>     db  000h, 02ah, 02ah, 02ah, 000h, 02ah, 02ah, 02ah, 000h, 000h, 015h, 000h, 000h, 03fh, 000h,
02ah
 51764 00011E9D 001500003F002A          <1>
 51765 00011EA4 15002A3F2A00152A00-      <1>     db  015h, 000h, 02ah, 03fh, 02ah, 000h, 015h, 02ah, 000h, 03fh, 02ah, 02ah, 015h, 02ah, 02ah,
03fh
 51766 00011EAD 3F2A2A152A2A3F          <1>
 51767 00011EB4 00150000152A003F00-      <1>     db  000h, 015h, 000h, 000h, 015h, 02ah, 000h, 03fh, 000h, 000h, 03fh, 02ah, 02ah, 015h, 000h,
02ah
 51768 00011EBD 003F2A2A15002A          <1>
 51769 00011EC4 152A2A3F002A3F2A00-      <1>     db  015h, 02ah, 02ah, 03fh, 000h, 02ah, 03fh, 02ah, 000h, 015h, 015h, 000h, 015h, 03fh, 000h,
03fh
 51770 00011ECD 151500153F003F          <1>
 51771 00011ED4 15003F3F2A15152A15-      <1>     db  015h, 000h, 03fh, 03fh, 02ah, 015h, 015h, 02ah, 015h, 03fh, 02ah, 03fh, 015h, 02ah, 03fh,
03fh
 51772 00011EDD 3F2A3F152A3F3F          <1>
```

```
 51773 00011EE4 15000015002A152A00- <1>     db  015h, 000h, 000h, 015h, 000h, 02ah, 015h, 02ah, 000h, 015h, 02ah, 02ah, 03fh, 000h, 000h,
03fh
 51774 00011EED 152A2A3F00003F      <1>
 51775 00011EF4 002A3F2A003F2A2A15- <1>     db  000h, 02ah, 03fh, 02ah, 000h, 03fh, 02ah, 02ah, 015h, 000h, 015h, 015h, 000h, 03fh, 015h,
02ah
 51776 00011EFD 001515003F152A      <1>
 51777 00011F04 15152A3F3F00153F00- <1>     db  015h, 015h, 02ah, 03fh, 03fh, 000h, 015h, 03fh, 000h, 03fh, 03fh, 02ah, 015h, 03fh, 02ah,
03fh
 51778 00011F0D 3F3F2A153F2A3F      <1>
 51779 00011F14 15150015152A153F00- <1>     db  015h, 015h, 000h, 015h, 015h, 02ah, 015h, 03fh, 000h, 015h, 03fh, 02ah, 03fh, 015h, 000h,
03fh
 51780 00011F1D 153F2A3F15003F      <1>
 51781 00011F24 152A3F3F003F3F2A15- <1>     db  015h, 02ah, 03fh, 03fh, 000h, 03fh, 03fh, 02ah, 015h, 015h, 015h, 015h, 015h, 03fh,
015h, 03fh
 51782 00011F2D 151515153F153F      <1>
 51783 00011F34 15153F3F3F15153F15- <1>     db  015h, 015h, 03fh, 03fh, 03fh, 015h, 015h, 03fh, 015h, 03fh, 03fh, 03fh, 015h, 03fh, 03fh,
03fh
 51784 00011F3D 3F3F3F153F3F3F      <1>
 51785                              <1> palette3:
 51786 00011F44 00000000002A002A00- <1>     db  000h, 000h, 000h, 000h, 000h, 02ah, 000h, 02ah, 000h, 000h, 02ah, 02ah, 02ah, 000h, 000h,
02ah
 51787 00011F4D 002A2A2A00002A      <1>
 51788 00011F54 002A2A15002A2A2A15- <1>     db  000h, 02ah, 02ah, 015h, 000h, 02ah, 02ah, 02ah, 015h, 015h, 015h, 015h, 015h, 03fh, 015h,
03fh
 51789 00011F5D 151515153F153F      <1>
 51790 00011F64 15153F3F3F15153F15- <1>     db  015h, 015h, 03fh, 03fh, 03fh, 015h, 015h, 03fh, 015h, 03fh, 03fh, 03fh, 015h, 03fh, 03fh,
03fh
 51791 00011F6D 3F3F3F153F3F3F      <1>
 51792 00011F74 000000050505080808- <1>     db  000h, 000h, 000h, 005h, 005h, 005h, 008h, 008h, 008h, 00bh, 00bh, 00bh, 00eh, 00eh, 00eh,
011h
 51793 00011F7D 0B0B0B0E0E0E11      <1>
 51794 00011F84 111114141418181C- <1>     db  011h, 011h, 014h, 014h, 014h, 018h, 018h, 018h, 01ch, 01ch, 01ch, 020h, 020h, 020h, 024h,
024h
 51795 00011F8D 1C1C2020202424      <1>
 51796 00011F94 242828282D2D2D3232- <1>     db  024h, 028h, 028h, 028h, 02dh, 02dh, 02dh, 032h, 032h, 032h, 038h, 038h, 038h, 03fh, 03fh,
03fh
 51797 00011F9D 323838383F3F3F      <1>
 51798 00011FA4 00003F10003F1F003F- <1>     db  000h, 000h, 03fh, 010h, 000h, 03fh, 01fh, 000h, 03fh, 02fh, 000h, 03fh, 03fh, 000h, 03fh,
03fh
 51799 00011FAD 2F003F3F003F3F      <1>
 51800 00011FB4 002F3F001F3F00103F- <1>     db  000h, 02fh, 03fh, 000h, 01fh, 03fh, 000h, 010h, 03fh, 000h, 000h, 03fh, 010h, 000h, 03fh,
01fh
 51801 00011FBD 00003F10003F1F      <1>
 51802 00011FC4 003F2F003F3F002F3F- <1>     db  000h, 03fh, 02fh, 000h, 03fh, 03fh, 000h, 02fh, 03fh, 000h, 01fh, 03fh, 000h, 010h, 03fh,
000h
 51803 00011FCD 001F3F00103F00      <1>
 51804 00011FD4 003F00003F10003F1F- <1>     db  000h, 03fh, 000h, 000h, 03fh, 010h, 000h, 03fh, 01fh, 000h, 03fh, 02fh, 000h, 03fh, 03fh,
000h
 51805 00011FDD 003F2F003F3F00      <1>
 51806 00011FE4 2F3F001F3F00103F1F- <1>     db  02fh, 03fh, 000h, 01fh, 03fh, 000h, 010h, 03fh, 01fh, 01fh, 03fh, 027h, 01fh, 03fh, 02fh,
01fh
 51807 00011FED 1F3F271F3F2F1F      <1>
 51808 00011FF4 3F371F3F3F1F3F3F1F- <1>     db  03fh, 037h, 01fh, 03fh, 03fh, 01fh, 03fh, 03fh, 01fh, 037h, 03fh, 01fh, 02fh, 03fh, 01fh,
027h
 51809 00011FFD 373F1F2F3F1F27      <1>
 51810 00012004 3F1F1F3F271F3F2F1F- <1>     db  03fh, 01fh, 01fh, 03fh, 027h, 01fh, 03fh, 02fh, 01fh, 03fh, 037h, 01fh, 03fh, 03fh, 01fh,
037h
 51811 0001200D 3F371F3F3F1F37      <1>
 51812 00012014 3F1F2F3F1F273F1F1F- <1>     db  03fh, 01fh, 02fh, 03fh, 01fh, 027h, 03fh, 01fh, 01fh, 03fh, 01fh, 01fh, 03fh, 027h, 01fh,
03fh
 51813 0001201D 3F1F1F3F271F3F      <1>
 51814 00012024 2F1F3F371F3F3F1F37- <1>     db  02fh, 01fh, 03fh, 037h, 01fh, 03fh, 03fh, 01fh, 037h, 03fh, 01fh, 02fh, 03fh, 01fh, 027h,
03fh
 51815 0001202D 3F1F2F3F1F273F      <1>
 51816 00012034 2D2D3F312D3F362D3F- <1>     db  02dh, 02dh, 03fh, 031h, 02dh, 03fh, 036h, 02dh, 03fh, 03ah, 02dh, 03fh, 03fh, 02dh, 03fh,
03fh
 51817 0001203D 3A2D3F3F2D3F3F      <1>
 51818 00012044 2D3A3F2D363F2D313F- <1>     db  02dh, 03ah, 03fh, 02dh, 036h, 03fh, 02dh, 031h, 03fh, 02dh, 02dh, 03fh, 031h, 02dh, 03fh,
036h
 51819 0001204D 2D2D3F312D3F36      <1>
 51820 00012054 2D3F3A2D3F3F2D3A3F- <1>     db  02dh, 03fh, 03ah, 02dh, 03fh, 03fh, 02dh, 03ah, 03fh, 02dh, 036h, 03fh, 02dh, 031h, 03fh,
02dh
 51821 0001205D 2D363F2D313F2D      <1>
 51822 00012064 2D3F2D2D3F312D3F36- <1>     db  02dh, 03fh, 02dh, 02dh, 03fh, 031h, 02dh, 03fh, 036h, 02dh, 03fh, 03ah, 02dh, 03fh, 03fh,
02dh
 51823 0001206D 2D3F3A2D3F3F2D      <1>
 51824 00012074 3A3F2D363F2D313F00- <1>     db  03ah, 03fh, 02dh, 036h, 03fh, 02dh, 031h, 03fh, 000h, 000h, 01ch, 007h, 000h, 01ch, 00eh,
000h
 51825 0001207D 001C07001C0E00      <1>
 51826 00012084 1C15001C1C001C1C00- <1>     db  01ch, 015h, 000h, 01ch, 01ch, 000h, 01ch, 01ch, 000h, 015h, 01ch, 000h, 00eh, 01ch, 000h,
007h
 51827 0001208D 151C000E1C0007      <1>
 51828 00012094 1C00001C07001C0E00- <1>     db  01ch, 000h, 000h, 01ch, 007h, 000h, 01ch, 00eh, 000h, 01ch, 015h, 000h, 01ch, 01ch, 000h,
015h
 51829 0001209D 1C15001C1C0015      <1>
 51830 000120A4 1C000E1C00071C0000- <1>     db  01ch, 000h, 00eh, 01ch, 000h, 007h, 01ch, 000h, 000h, 01ch, 000h, 000h, 01ch, 007h, 000h,
01ch
 51831 000120AD 1C00001C07001C      <1>
 51832 000120B4 0E001C15001C1C0015- <1>     db  00eh, 000h, 01ch, 015h, 000h, 01ch, 01ch, 000h, 015h, 01ch, 000h, 00eh, 01ch, 000h, 007h,
01ch
 51833 000120BD 1C000E1C00071C      <1>
 51834 000120C4 0E0E1C110E1C150E1C- <1>     db  00eh, 00eh, 01ch, 011h, 00eh, 01ch, 015h, 00eh, 01ch, 018h, 00eh, 01ch, 01ch, 00eh, 01ch,
01ch
 51835 000120CD 180E1C1C0E1C1C      <1>
 51836 000120D4 0E181C0E151C0E111C- <1>     db  00eh, 018h, 01ch, 00eh, 015h, 01ch, 00eh, 011h, 01ch, 00eh, 00eh, 01ch, 011h, 00eh, 01ch,
015h
 51837 000120DD 0E0E1C110E1C15      <1>
 51838 000120E4 0E1C180E1C1C0E181C- <1>     db  00eh, 01ch, 018h, 00eh, 01ch, 01ch, 00eh, 018h, 01ch, 00eh, 015h, 01ch, 00eh, 011h, 01ch,
00eh
 51839 000120ED 0E151C0E111C0E      <1>
 51840 000120F4 0E1C0E0E1C110E1C15- <1>     db  00eh, 01ch, 00eh, 00eh, 01ch, 011h, 00eh, 01ch, 015h, 00eh, 01ch, 018h, 00eh, 01ch, 01ch,
00eh
```

```
 51841 000120FD 0E1C180E1C1C0E      <1>
 51842 00012104 181C0E151C0E111C14- <1>     db  018h, 01ch, 00eh, 015h, 01ch, 00eh, 011h, 01ch, 014h, 014h, 01ch, 016h, 014h, 01ch, 018h,
014h
 51843 0001210D 141C16141C1814      <1>
 51844 00012114 1C1A141C1C141C1C14- <1>     db  01ch, 01ah, 014h, 01ch, 01ch, 014h, 01ch, 01ch, 014h, 01ah, 01ch, 014h, 018h, 01ch, 014h,
016h
 51845 0001211D 1A1C14181C1416      <1>
 51846 00012124 1C14141C16141C1814- <1>     db  01ch, 014h, 014h, 01ch, 016h, 014h, 01ch, 018h, 014h, 01ch, 01ah, 014h, 01ch, 01ch, 014h,
01ah
 51847 0001212D 1C1A141C1C141A      <1>
 51848 00012134 1C14181C14161C1414- <1>     db  01ch, 014h, 018h, 01ch, 014h, 016h, 01ch, 014h, 014h, 01ch, 014h, 014h, 01ch, 016h,
014h, 01ch
 51849 0001213D 1C14141C16141C      <1>
 51850 00012144 18141C1A141C1C141A- <1>     db  018h, 014h, 01ch, 01ah, 014h, 01ch, 01ch, 014h, 01ah, 01ch, 014h, 018h, 01ch, 014h, 016h,
01ch
 51851 0001214D 1C14181C14161C      <1>
 51852 00012154 000010040010080010- <1>     db  000h, 000h, 010h, 004h, 000h, 010h, 008h, 000h, 010h, 00ch, 000h, 010h, 010h, 000h, 010h,
010h
 51853 0001215D 0C001010001010      <1>
 51854 00012164 000C10000810000410- <1>     db  000h, 00ch, 010h, 000h, 008h, 010h, 000h, 004h, 010h, 000h, 000h, 010h, 004h, 000h, 010h,
008h
 51855 0001216D 00001004001008      <1>
 51856 00012174 00100C001010000C10- <1>     db  000h, 010h, 00ch, 000h, 010h, 010h, 000h, 00ch, 010h, 000h, 008h, 010h, 000h, 004h, 010h,
000h
 51857 0001217D 00081000041000      <1>
 51858 00012184 001000001004001008- <1>     db  000h, 010h, 000h, 000h, 010h, 004h, 000h, 010h, 008h, 000h, 010h, 00ch, 000h, 010h, 010h,
000h
 51859 0001218D 00100C00101000      <1>
 51860 00012194 0C1000081000041008- <1>     db  00ch, 010h, 000h, 008h, 010h, 000h, 004h, 010h, 008h, 008h, 010h, 00ah, 008h, 010h, 00ch,
008h
 51861 0001219D 08100A08100C08      <1>
 51862 000121A4 100E08101008101008- <1>     db  010h, 00eh, 008h, 010h, 010h, 008h, 010h, 010h, 008h, 00eh, 010h, 008h, 00ch, 010h, 008h,
00ah
 51863 000121AD 0E10080C10080A      <1>
 51864 000121B4 100808100A08100C08- <1>     db  010h, 008h, 008h, 010h, 00ah, 008h, 010h, 00ch, 008h, 010h, 00eh, 008h, 010h, 010h, 008h,
00eh
 51865 000121BD 100E081010080E      <1>
 51866 000121C4 10080C10080A100808- <1>     db  010h, 008h, 00ch, 010h, 008h, 00ah, 010h, 008h, 008h, 010h, 008h, 008h, 010h, 00ah, 008h,
010h
 51867 000121CD 100808100A0810      <1>
 51868 000121D4 0C08100E081010080E- <1>     db  00ch, 008h, 010h, 00eh, 008h, 010h, 010h, 008h, 00eh, 010h, 008h, 00ch, 010h, 008h, 00ah,
010h
 51869 000121DD 10080C10080A10      <1>
 51870 000121E4 0B0B100C0B100D0B10- <1>     db  00bh, 00bh, 010h, 00ch, 00bh, 010h, 00dh, 00bh, 010h, 00fh, 00bh, 010h, 010h, 00bh, 010h,
010h
 51871 000121ED 0F0B10100B1010      <1>
 51872 000121F4 0B0F100B0D100B0C10- <1>     db  00bh, 00fh, 010h, 00bh, 00dh, 010h, 00bh, 00ch, 010h, 00bh, 00bh, 010h, 00ch, 00bh, 010h,
00dh
 51873 000121FD 0B0B100C0B100D      <1>
 51874 00012204 0B100F0B10100B0F10- <1>     db  00bh, 010h, 00fh, 00bh, 010h, 010h, 00bh, 00fh, 010h, 00bh, 00dh, 010h, 00bh, 00ch, 010h,
00bh
 51875 0001220D 0B0D100B0C100B      <1>
 51876 00012214 0B100B0B100C0B100D- <1>     db  00bh, 010h, 00bh, 00bh, 010h, 00ch, 00bh, 010h, 00dh, 00bh, 010h, 00fh, 00bh, 010h, 010h,
00bh
 51877 0001221D 0B100F0B10100B      <1>
 51878 00012224 0F100B0D100B0C1000- <1>     db  00fh, 010h, 00bh, 00dh, 010h, 00bh, 00ch, 010h, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 51879 0001222D 00000000000000      <1>
 51880 00012234 0000000000000000- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 51881 0001223D 00000000000000      <1>
 51882                              <1>
 51883                              <1>
 51884                              <1> ; 04/07/2016
 51885                              <1> ; FONT DATA
 51886                              <1>
 51887                              <1> CRT_CHAR_GEN:
 51888                              <1> vgafont8:
 51889 00012244 0000000000000007E- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 07eh, 081h, 0a5h, 081h, 0bdh, 099h, 081h,
07eh
 51890 0001224D 81A581BD99817E      <1>
 51891 00012254 7EFFDBFFC3E7FF7E6C- <1>     db  07eh, 0ffh, 0dbh, 0ffh, 0c3h, 0e7h, 0ffh, 07eh, 06ch, 0feh, 0feh, 0feh, 07ch, 038h, 010h,
000h
 51892 0001225D FEFEFE7C381000      <1>
 51893 00012264 10387CFE7C38100038- <1>     db  010h, 038h, 07ch, 0feh, 07ch, 038h, 010h, 000h, 038h, 07ch, 038h, 0feh, 0feh, 07ch, 038h,
07ch
 51894 0001226D 7C38FEFE7C387C      <1>
 51895 00012274 1010387CFE7C387C00- <1>     db  010h, 010h, 038h, 07ch, 0feh, 07ch, 038h, 07ch, 000h, 000h, 018h, 03ch, 03ch, 018h, 000h,
000h
 51896 0001227D 00183C3C180000      <1>
 51897 00012284 FFFFE7C3C3E7FFFF00- <1>     db  0ffh, 0ffh, 0e7h, 0c3h, 0c3h, 0e7h, 0ffh, 0ffh, 000h, 03ch, 066h, 042h, 042h, 066h, 03ch,
000h
 51898 0001228D 3C664242663C00      <1>
 51899 00012294 FFC399BDBD99C3FF0F- <1>     db  0ffh, 0c3h, 099h, 0bdh, 0bdh, 099h, 0c3h, 0ffh, 00fh, 007h, 00fh, 07dh, 0cch, 0cch, 0cch,
078h
 51900 0001229D 070F7DCCCCCC78      <1>
 51901 000122A4 3C6666663C187E183F- <1>     db  03ch, 066h, 066h, 066h, 03ch, 018h, 07eh, 018h, 03fh, 033h, 03fh, 030h, 030h, 070h, 0f0h,
0e0h
 51902 000122AD 333F303070F0E0      <1>
 51903 000122B4 7F637F636367E6C099- <1>     db  07fh, 063h, 07fh, 063h, 063h, 067h, 0e6h, 0c0h, 099h, 05ah, 03ch, 0e7h, 0e7h, 03ch, 05ah,
099h
 51904 000122BD 5A3CE7E73C5A99      <1>
 51905 000122C4 80E0F8FEF8E0800002- <1>     db  080h, 0e0h, 0f8h, 0feh, 0f8h, 0e0h, 080h, 000h, 002h, 00eh, 03eh, 0feh, 03eh, 00eh, 002h,
000h
 51906 000122CD 0E3EFE3E0E0200      <1>
 51907 000122D4 183C7E18187E3C1866- <1>     db  018h, 03ch, 07eh, 018h, 018h, 07eh, 03ch, 018h, 066h, 066h, 066h, 066h, 066h, 000h, 066h,
000h
 51908 000122DD 66666666006600      <1>
 51909 000122E4 7FDBDB7B1B1B1B003E- <1>     db  07fh, 0dbh, 0dbh, 07bh, 01bh, 01bh, 01bh, 000h, 03eh, 063h, 038h, 06ch, 06ch, 038h, 0cch,
078h
 51910 000122ED 63386C6C38CC78      <1>
```

```
 51911 000122F4 000000007E7E7E0018- <1>     db  000h, 000h, 000h, 000h, 07eh, 07eh, 07eh, 000h, 018h, 03ch, 07eh, 018h, 07eh, 03ch, 018h,
0ffh
 51912 000122FD 3C7E187E3C18FF      <1>
 51913 00012304 183C7E181818180018- <1>     db  018h, 03ch, 07eh, 018h, 018h, 018h, 018h, 000h, 018h, 018h, 018h, 018h, 07eh, 03ch, 018h,
000h
 51914 0001230D 1818187E3C1800      <1>
 51915 00012314 00180CFE0C18000000- <1>     db  000h, 018h, 00ch, 0feh, 00ch, 018h, 000h, 000h, 000h, 030h, 060h, 0feh, 060h, 030h, 000h,
000h
 51916 0001231D 3060FE60300000      <1>
 51917 00012324 0000C0C0C0FE000000- <1>     db  000h, 000h, 0c0h, 0c0h, 0c0h, 0feh, 000h, 000h, 000h, 024h, 066h, 0ffh, 066h, 024h, 000h,
000h
 51918 0001232D 2466FF66240000      <1>
 51919 00012334 00183C7EFFFF000000- <1>     db  000h, 018h, 03ch, 07eh, 0ffh, 0ffh, 000h, 000h, 000h, 0ffh, 0ffh, 07eh, 03ch, 018h,
000h, 000h
 51920 0001233D FFFF7E3C180000      <1>
 51921 00012344 000000000000000030- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 030h, 078h, 078h, 030h, 030h, 000h, 030h,
000h
 51922 0001234D 78783030003000      <1>
 51923 00012354 6C6C6C00000000006C- <1>     db  06ch, 06ch, 06ch, 000h, 000h, 000h, 000h, 000h, 06ch, 06ch, 0feh, 06ch, 0feh, 06ch, 06ch,
000h
 51924 0001235D 6CFE6CFE6C6C00      <1>
 51925 00012364 307CC0780CF8300000- <1>     db  030h, 07ch, 0c0h, 078h, 00ch, 0f8h, 030h, 000h, 000h, 0c6h, 0cch, 018h, 030h, 066h, 0c6h,
000h
 51926 0001236D C6CC183066C600      <1>
 51927 00012374 386C3876DCCC760060- <1>     db  038h, 06ch, 038h, 076h, 0dch, 0cch, 076h, 000h, 060h, 060h, 0c0h, 000h, 000h, 000h, 000h,
000h
 51928 0001237D 60C00000000000      <1>
 51929 00012384 183060606030180060- <1>     db  018h, 030h, 060h, 060h, 060h, 030h, 018h, 000h, 060h, 030h, 018h, 018h, 018h, 030h, 060h,
000h
 51930 0001238D 30181818306000      <1>
 51931 00012394 00663CFF3C66000000- <1>     db  000h, 066h, 03ch, 0ffh, 03ch, 066h, 000h, 000h, 000h, 030h, 030h, 0fch, 030h, 030h, 000h,
000h
 51932 0001239D 3030FC30300000      <1>
 51933 000123A4 000000000030306000- <1>     db  000h, 000h, 000h, 000h, 000h, 030h, 030h, 060h, 000h, 000h, 000h, 0fch, 000h, 000h, 000h,
000h
 51934 000123AD 0000FC00000000      <1>
 51935 000123B4 000000000030300006- <1>     db  000h, 000h, 000h, 000h, 000h, 030h, 030h, 000h, 006h, 00ch, 018h, 030h, 060h, 0c0h, 080h,
000h
 51936 000123BD 0C183060C08000      <1>
 51937 000123C4 7CC6CEDEF6E67C0030- <1>     db  07ch, 0c6h, 0ceh, 0deh, 0f6h, 0e6h, 07ch, 000h, 030h, 070h, 030h, 030h, 030h, 030h, 0fch,
000h
 51938 000123CD 7030303030FC00      <1>
 51939 000123D4 78CC0C3860CCFC0078- <1>     db  078h, 0cch, 00ch, 038h, 060h, 0cch, 0fch, 000h, 078h, 0cch, 00ch, 038h, 00ch, 0cch, 078h,
000h
 51940 000123DD CC0C380CCC7800      <1>
 51941 000123E4 1C3C6CCCFE0C1E00FC- <1>     db  01ch, 03ch, 06ch, 0cch, 0feh, 00ch, 01eh, 000h, 0fch, 0c0h, 0f8h, 00ch, 00ch, 0cch, 078h,
000h
 51942 000123ED C0F80C0CCC7800      <1>
 51943 000123F4 3860C0F8CCCC7800FC- <1>     db  038h, 060h, 0c0h, 0f8h, 0cch, 0cch, 078h, 000h, 0fch, 0cch, 00ch, 018h, 030h, 030h, 030h,
000h
 51944 000123FD CC0C1830303000      <1>
 51945 00012404 78CCCC78CCCC780078- <1>     db  078h, 0cch, 0cch, 078h, 0cch, 0cch, 078h, 000h, 078h, 0cch, 0cch, 07ch, 00ch, 018h, 070h,
000h
 51946 0001240D CCCC7C0C187000      <1>
 51947 00012414 003030000030300000- <1>     db  000h, 030h, 030h, 000h, 000h, 030h, 030h, 000h, 000h, 030h, 030h, 000h, 000h, 030h, 030h,
060h
 51948 0001241D 30300000303060      <1>
 51949 00012424 183060C06030180000- <1>     db  018h, 030h, 060h, 0c0h, 060h, 030h, 018h, 000h, 000h, 000h, 0fch, 000h, 000h, 0fch, 000h,
000h
 51950 0001242D 00FC0000FC0000      <1>
 51951 00012434 6030180C1830600078- <1>     db  060h, 030h, 018h, 00ch, 018h, 030h, 060h, 000h, 078h, 0cch, 00ch, 018h, 030h, 000h, 030h,
000h
 51952 0001243D CC0C1830003000      <1>
 51953 00012444 7CC6DEDEDEC0780030- <1>     db  07ch, 0c6h, 0deh, 0deh, 0deh, 0c0h, 078h, 000h, 030h, 078h, 0cch, 0cch, 0fch, 0cch, 0cch,
000h
 51954 0001244D 78CCCCFCCCCC00      <1>
 51955 00012454 FC66667C6666FC003C- <1>     db  0fch, 066h, 066h, 07ch, 066h, 066h, 0fch, 000h, 03ch, 066h, 0c0h, 0c0h, 0c0h, 066h, 03ch,
000h
 51956 0001245D 66C0C0C0663C00      <1>
 51957 00012464 F86C6666666CF800FE- <1>     db  0f8h, 06ch, 066h, 066h, 066h, 06ch, 0f8h, 000h, 0feh, 062h, 068h, 078h, 068h, 062h, 0feh,
000h
 51958 0001246D 6268786862FE00      <1>
 51959 00012474 FE6268786860F0003C- <1>     db  0feh, 062h, 068h, 078h, 068h, 060h, 0f0h, 000h, 03ch, 066h, 0c0h, 0c0h, 0ceh, 066h, 03eh,
000h
 51960 0001247D 66C0C0CE663E00      <1>
 51961 00012484 CCCCCCFCCCCCCC0078- <1>     db  0cch, 0cch, 0cch, 0fch, 0cch, 0cch, 0cch, 000h, 078h, 030h, 030h, 030h, 030h, 030h, 078h,
000h
 51962 0001248D 30303030307800      <1>
 51963 00012494 1E0C0C0CCCCC7800E6- <1>     db  01eh, 00ch, 00ch, 00ch, 0cch, 0cch, 078h, 000h, 0e6h, 066h, 06ch, 078h, 06ch, 066h, 0e6h,
000h
 51964 0001249D 666C786C66E600      <1>
 51965 000124A4 F06060606266FE00C6- <1>     db  0f0h, 060h, 060h, 060h, 062h, 066h, 0feh, 000h, 0c6h, 0eeh, 0feh, 0feh, 0d6h, 0c6h, 0c6h,
000h
 51966 000124AD EEFEFED6C6C600      <1>
 51967 000124B4 C6E6F6DECEC6C60038- <1>     db  0c6h, 0e6h, 0f6h, 0deh, 0ceh, 0c6h, 0c6h, 000h, 038h, 06ch, 0c6h, 0c6h, 0c6h, 06ch, 038h,
000h
 51968 000124BD 6CC6C6C66C3800      <1>
 51969 000124C4 FC66667C6060F00078- <1>     db  0fch, 066h, 066h, 07ch, 060h, 060h, 0f0h, 000h, 078h, 0cch, 0cch, 0cch, 0dch, 078h, 01ch,
000h
 51970 000124CD CCCCCCDC781C00      <1>
 51971 000124D4 FC66667C6C66E60078- <1>     db  0fch, 066h, 066h, 07ch, 06ch, 066h, 0e6h, 000h, 078h, 0cch, 0e0h, 070h, 01ch, 0cch, 078h,
000h
 51972 000124DD CCE0701CCC7800      <1>
 51973 000124E4 FCB4303030307800CC- <1>     db  0fch, 0b4h, 030h, 030h, 030h, 030h, 078h, 000h, 0cch, 0cch, 0cch, 0cch, 0cch, 0cch, 0fch,
000h
 51974 000124ED CCCCCCCCCCFC00      <1>
 51975 000124F4 CCCCCCCCCC783000C6- <1>     db  0cch, 0cch, 0cch, 0cch, 0cch, 078h, 030h, 000h, 0c6h, 0c6h, 0c6h, 0d6h, 0feh, 0eeh, 0c6h,
000h
 51976 000124FD C6C6D6FEEEC600      <1>
 51977 00012504 C6C66C38386CC600CC- <1>     db  0c6h, 0c6h, 06ch, 038h, 038h, 06ch, 0c6h, 000h, 0cch, 0cch, 0cch, 078h, 030h, 030h, 078h,
000h
 51978 0001250D CCCC7830307800      <1>
```

```
 51979 00012514 FEC68C183266FE0078- <1>     db  0feh, 0c6h, 08ch, 018h, 032h, 066h, 0feh, 000h, 078h, 060h, 060h, 060h, 060h, 060h, 078h,
000h
 51980 0001251D 60606060607800      <1>
 51981 00012524 C06030180C06020078- <1>     db  0c0h, 060h, 030h, 018h, 00ch, 006h, 002h, 000h, 078h, 018h, 018h, 018h, 018h, 018h, 078h,
000h
 51982 0001252D 18181818187800      <1>
 51983 00012534 10386CC60000000000- <1>     db  010h, 038h, 06ch, 0c6h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
0ffh
 51984 0001253D 000000000000FF      <1>
 51985 00012544 303018000000000000- <1>     db  030h, 030h, 018h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 078h, 00ch, 07ch, 0cch, 076h,
000h
 51986 0001254D 00780C7CCC7600      <1>
 51987 00012554 E060607C6666DC0000- <1>     db  0e0h, 060h, 060h, 07ch, 066h, 066h, 0dch, 000h, 000h, 000h, 078h, 0cch, 0c0h, 0cch,
078h, 000h
 51988 0001255D 0078CCC0CC7800      <1>
 51989 00012564 1C0C0C7CCCCC760000- <1>     db  01ch, 00ch, 00ch, 07ch, 0cch, 0cch, 076h, 000h, 000h, 000h, 078h, 0cch, 0fch, 0c0h, 078h,
000h
 51990 0001256D 0078CCFCC07800      <1>
 51991 00012574 386C60F06060F00000- <1>     db  038h, 06ch, 060h, 0f0h, 060h, 060h, 0f0h, 000h, 000h, 000h, 076h, 0cch, 0cch, 07ch, 00ch,
0f8h
 51992 0001257D 0076CCCC7C0CF8      <1>
 51993 00012584 E0606C766666E60030- <1>     db  0e0h, 060h, 06ch, 076h, 066h, 066h, 0e6h, 000h, 030h, 000h, 070h, 030h, 030h, 030h, 078h,
000h
 51994 0001258D 00703030307800      <1>
 51995 00012594 0C000C0C0CCCCC78E0- <1>     db  00ch, 000h, 00ch, 00ch, 00ch, 0cch, 0cch, 078h, 0e0h, 060h, 066h, 06ch, 078h, 06ch, 0e6h,
000h
 51996 0001259D 60666C786CE600      <1>
 51997 000125A4 703030303030780000- <1>     db  070h, 030h, 030h, 030h, 030h, 030h, 078h, 000h, 000h, 000h, 0cch, 0feh, 0feh, 0d6h, 0c6h,
000h
 51998 000125AD 00CCFEFED6C600      <1>
 51999 000125B4 0000F8CCCCCCCC0000- <1>     db  000h, 000h, 0f8h, 0cch, 0cch, 0cch, 0cch, 000h, 000h, 000h, 078h, 0cch, 0cch, 0cch, 078h,
000h
 52000 000125BD 0078CCCCCC7800      <1>
 52001 000125C4 0000DC66667C60F000- <1>     db  000h, 000h, 0dch, 066h, 066h, 07ch, 060h, 0f0h, 000h, 000h, 076h, 0cch, 0cch, 07ch, 00ch,
01eh
 52002 000125CD 0076CCCC7C0C1E      <1>
 52003 000125D4 0000DC766660F00000- <1>     db  000h, 000h, 0dch, 076h, 066h, 060h, 0f0h, 000h, 000h, 000h, 07ch, 0c0h, 078h, 00ch, 0f8h,
000h
 52004 000125DD 007CC0780CF800      <1>
 52005 000125E4 10307C303034180000- <1>     db  010h, 030h, 07ch, 030h, 030h, 034h, 018h, 000h, 000h, 000h, 0cch, 0cch, 0cch, 0cch, 076h,
000h
 52006 000125ED 00CCCCCCCC7600      <1>
 52007 000125F4 0000CCCCCC78300000- <1>     db  000h, 000h, 0cch, 0cch, 0cch, 078h, 030h, 000h, 000h, 000h, 0c6h, 0d6h, 0feh, 0feh, 06ch,
000h
 52008 000125FD 00C6D6FEFE6C00      <1>
 52009 00012604 0000C66C386CC60000- <1>     db  000h, 000h, 0c6h, 06ch, 038h, 06ch, 0c6h, 000h, 000h, 000h, 0cch, 0cch, 0cch, 07ch, 00ch,
0f8h
 52010 0001260D 00CCCCCC7C0CF8      <1>
 52011 00012614 0000FC983064FC001C- <1>     db  000h, 000h, 0fch, 098h, 030h, 064h, 0fch, 000h, 01ch, 030h, 030h, 0e0h, 030h, 030h, 01ch,
000h
 52012 0001261D 3030E030301C00      <1>
 52013 00012624 1818180018181800E0- <1>     db  018h, 018h, 018h, 000h, 018h, 018h, 018h, 000h, 0e0h, 030h, 030h, 01ch, 030h, 030h, 0e0h,
000h
 52014 0001262D 30301C3030E000      <1>
 52015 00012634 76DC00000000000000- <1>     db  076h, 0dch, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 010h, 038h, 06ch, 0c6h, 0c6h, 0feh,
000h
 52016 0001263D 10386CC6C6FE00      <1>
 52017 00012644 78CCC0CC78180C7800- <1>     db  078h, 0cch, 0c0h, 0cch, 078h, 018h, 00ch, 078h, 000h, 0cch, 000h, 0cch, 0cch, 0cch, 07eh,
000h
 52018 0001264D CC00CCCCCC7E00      <1>
 52019 00012654 1C0078CCFCC078007E- <1>     db  01ch, 000h, 078h, 0cch, 0fch, 0c0h, 078h, 000h, 07eh, 0c3h, 03ch, 006h, 03eh, 066h, 03fh,
000h
 52020 0001265D C33C063E663F00      <1>
 52021 00012664 CC00780C7CCC7E00E0- <1>     db  0cch, 000h, 078h, 00ch, 07ch, 0cch, 07eh, 000h, 0e0h, 000h, 078h, 00ch, 07ch, 0cch, 07eh,
000h
 52022 0001266D 00780C7CCC7E00      <1>
 52023 00012674 3030780C7CCC7E0000- <1>     db  030h, 030h, 078h, 00ch, 07ch, 0cch, 07eh, 000h, 000h, 000h, 078h, 0c0h, 0c0h, 078h, 00ch,
038h
 52024 0001267D 0078C0C0780C38      <1>
 52025 00012684 7EC33C667E603C00CC- <1>     db  07eh, 0c3h, 03ch, 066h, 07eh, 060h, 03ch, 000h, 0cch, 000h, 078h, 0cch, 0fch, 0c0h, 078h,
000h
 52026 0001268D 0078CCFCC07800      <1>
 52027 00012694 E00078CCFCC07800CC- <1>     db  0e0h, 000h, 078h, 0cch, 0fch, 0c0h, 078h, 000h, 0cch, 000h, 070h, 030h, 030h, 030h, 078h,
000h
 52028 0001269D 00703030307800      <1>
 52029 000126A4 7CC6381818183C00E0- <1>     db  07ch, 0c6h, 038h, 018h, 018h, 018h, 03ch, 000h, 0e0h, 000h, 070h, 030h, 030h, 030h, 078h,
000h
 52030 000126AD 00703030307800      <1>
 52031 000126B4 C6386CC6FEC6C60030- <1>     db  0c6h, 038h, 06ch, 0c6h, 0feh, 0c6h, 0c6h, 000h, 030h, 030h, 000h, 078h, 0cch, 0fch, 0cch,
000h
 52032 000126BD 300078CCFCCC00      <1>
 52033 000126C4 1C00FC607860FC0000- <1>     db  01ch, 000h, 0fch, 060h, 078h, 060h, 0fch, 000h, 000h, 000h, 07fh, 00ch, 07fh, 0cch, 07fh,
000h
 52034 000126CD 007F0C7FCC7F00      <1>
 52035 000126D4 3E6CCCFECCCCCE0078- <1>     db  03eh, 06ch, 0cch, 0feh, 0cch, 0cch, 0ceh, 000h, 078h, 0cch, 000h, 078h, 0cch, 0cch, 078h,
000h
 52036 000126DD CC0078CCCC7800      <1>
 52037 000126E4 00CC0078CCCC780000- <1>     db  000h, 0cch, 000h, 078h, 0cch, 0cch, 078h, 000h, 000h, 0e0h, 000h, 078h, 0cch, 0cch, 078h,
000h
 52038 000126ED E00078CCCC7800      <1>
 52039 000126F4 78CC00CCCCCC7E0000- <1>     db  078h, 0cch, 000h, 0cch, 0cch, 0cch, 07eh, 000h, 000h, 0e0h, 000h, 0cch, 0cch, 0cch, 07eh,
000h
 52040 000126FD E000CCCCCC7E00      <1>
 52041 00012704 00CC00CCCC7C0CF8C3- <1>     db  000h, 0cch, 000h, 0cch, 0cch, 07ch, 00ch, 0f8h, 0c3h, 018h, 03ch, 066h, 066h, 03ch, 018h,
000h
 52042 0001270D 183C66663C1800      <1>
 52043 00012714 CC00CCCCCCCC780018- <1>     db  0cch, 000h, 0cch, 0cch, 0cch, 0cch, 078h, 000h, 018h, 018h, 07eh, 0c0h, 0c0h, 07eh, 018h,
018h
 52044 0001271D 187EC0C07E1818      <1>
 52045 00012724 386C64F060E6FC00CC- <1>     db  038h, 06ch, 064h, 0f0h, 060h, 0e6h, 0fch, 000h, 0cch, 0cch, 078h, 0fch, 030h, 0fch, 030h,
030h
 52046 0001272D CC78FC30FC3030      <1>
```

```
 52047 00012734 F8CCCCFAC6CFC6C70E- <1>     db  0f8h, 0cch, 0cch, 0fah, 0c6h, 0cfh, 0c6h, 0c7h, 00eh, 01bh, 018h, 03ch, 018h, 018h, 0d8h,
070h
 52048 0001273D 1B183C1818D870      <1>
 52049 00012744 1C00780C7CCC7E0038- <1>     db  01ch, 000h, 078h, 00ch, 07ch, 0cch, 07eh, 000h, 038h, 000h, 070h, 030h, 030h, 030h, 078h,
000h
 52050 0001274D 00703030307800      <1>
 52051 00012754 001C0078CCCC780000- <1>     db  000h, 01ch, 000h, 078h, 0cch, 0cch, 078h, 000h, 000h, 01ch, 000h, 0cch, 0cch, 0cch, 07eh,
000h
 52052 0001275D 1C00CCCCCC7E00      <1>
 52053 00012764 00F800F8CCCCCC00FC- <1>     db  000h, 0f8h, 000h, 0f8h, 0cch, 0cch, 0cch, 000h, 0fch, 000h, 0cch, 0ech, 0fch, 0dch, 0cch,
000h
 52054 0001276D 00CCECFCDCCC00      <1>
 52055 00012774 3C6C6C3E007E000038- <1>     db  03ch, 06ch, 06ch, 03eh, 000h, 07eh, 000h, 000h, 038h, 06ch, 06ch, 038h, 000h, 07ch,
000h, 000h
 52056 0001277D 6C6C38007C0000      <1>
 52057 00012784 30003060C0CC780000- <1>     db  030h, 000h, 030h, 060h, 0c0h, 0cch, 078h, 000h, 000h, 000h, 000h, 0fch, 0c0h, 0c0h, 000h,
000h
 52058 0001278D 0000FCC0C00000      <1>
 52059 00012794 000000FC0C0C0000C3- <1>     db  000h, 000h, 000h, 0fch, 00ch, 00ch, 000h, 000h, 0c3h, 0c6h, 0cch, 0deh, 033h, 066h, 0cch,
00fh
 52060 0001279D C6CCDE3366CC0F      <1>
 52061 000127A4 C3C6CCDB376FCF0318- <1>     db  0c3h, 0c6h, 0cch, 0dbh, 037h, 06fh, 0cfh, 003h, 018h, 018h, 000h, 018h, 018h, 018h, 018h,
000h
 52062 000127AD 18001818181800      <1>
 52063 000127B4 003366CC6633000000- <1>     db  000h, 033h, 066h, 0cch, 066h, 033h, 000h, 000h, 000h, 0cch, 066h, 033h, 066h, 0cch, 000h,
000h
 52064 000127BD CC663366CC0000      <1>
 52065 000127C4 228822882288228855- <1>     db  022h, 088h, 022h, 088h, 022h, 088h, 022h, 088h, 055h, 0aah, 055h, 0aah, 055h, 0aah, 055h,
0aah
 52066 000127CD AA55AA55AA55AA      <1>
 52067 000127D4 DB77DBEEDB77DBEE18- <1>     db  0dbh, 077h, 0dbh, 0eeh, 0dbh, 077h, 0dbh, 0eeh, 018h, 018h, 018h, 018h, 018h, 018h, 018h,
018h
 52068 000127DD 18181818181818      <1>
 52069 000127E4 18181818F818181818- <1>     db  018h, 018h, 018h, 018h, 0f8h, 018h, 018h, 018h, 018h, 018h, 0f8h, 018h, 0f8h, 018h, 018h,
018h
 52070 000127ED 18F818F8181818      <1>
 52071 000127F4 36363636F636363600- <1>     db  036h, 036h, 036h, 036h, 0f6h, 036h, 036h, 036h, 000h, 000h, 000h, 000h, 0feh, 036h, 036h,
036h
 52072 000127FD 000000FE363636      <1>
 52073 00012804 0000F818F818181836- <1>     db  000h, 000h, 0f8h, 018h, 0f8h, 018h, 018h, 018h, 036h, 036h, 0f6h, 006h, 0f6h, 036h, 036h,
036h
 52074 0001280D 36F606F6363636      <1>
 52075 00012814 3636363636363600- <1>     db  036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 000h, 000h, 0feh, 006h, 0f6h, 036h, 036h,
036h
 52076 0001281D 00FE06F6363636      <1>
 52077 00012824 3636F606FE00000036- <1>     db  036h, 036h, 0f6h, 006h, 0feh, 000h, 000h, 000h, 036h, 036h, 036h, 036h, 0feh, 000h, 000h,
000h
 52078 0001282D 363636FE000000      <1>
 52079 00012834 1818F818F800000000- <1>     db  018h, 018h, 0f8h, 018h, 0f8h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0f8h, 018h, 018h,
018h
 52080 0001283D 000000F8181818      <1>
 52081 00012844 181818181F00000018- <1>     db  018h, 018h, 018h, 018h, 01fh, 000h, 000h, 000h, 018h, 018h, 018h, 018h, 0ffh, 000h, 000h,
000h
 52082 0001284D 181818FF000000      <1>
 52083 00012854 00000000FF18181818- <1>     db  000h, 000h, 000h, 000h, 0ffh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 01fh, 018h, 018h,
018h
 52084 0001285D 1818181F181818      <1>
 52085 00012864 00000000FF00000018- <1>     db  000h, 000h, 000h, 000h, 0ffh, 000h, 000h, 000h, 018h, 018h, 018h, 018h, 0ffh, 018h, 018h,
018h
 52086 0001286D 181818FF181818      <1>
 52087 00012874 18181F181F18181836- <1>     db  018h, 018h, 01fh, 018h, 01fh, 018h, 018h, 018h, 036h, 036h, 036h, 036h, 037h, 036h, 036h,
036h
 52088 0001287D 36363637363636      <1>
 52089 00012884 363637303F00000000- <1>     db  036h, 036h, 037h, 030h, 03fh, 000h, 000h, 000h, 000h, 000h, 03fh, 030h, 037h, 036h, 036h,
036h
 52090 0001288D 003F3037363636      <1>
 52091 00012894 3636F700FF00000000- <1>     db  036h, 036h, 0f7h, 000h, 0ffh, 000h, 000h, 000h, 000h, 000h, 0ffh, 000h, 0f7h, 036h, 036h,
036h
 52092 0001289D 00FF00F7363636      <1>
 52093 000128A4 363637303736363600- <1>     db  036h, 036h, 037h, 030h, 037h, 036h, 036h, 036h, 000h, 000h, 0ffh, 000h, 0ffh, 000h, 000h,
000h
 52094 000128AD 00FF00FF000000      <1>
 52095 000128B4 3636F700F736363618- <1>     db  036h, 036h, 0f7h, 000h, 0f7h, 036h, 036h, 036h, 018h, 018h, 0ffh, 000h, 0ffh, 000h, 000h,
000h
 52096 000128BD 18FF00FF000000      <1>
 52097 000128C4 36363636FF00000000- <1>     db  036h, 036h, 036h, 036h, 0ffh, 000h, 000h, 000h, 000h, 000h, 0ffh, 000h, 0ffh, 018h, 018h,
018h
 52098 000128CD 00FF00FF181818      <1>
 52099 000128D4 00000000FF36363636- <1>     db  000h, 000h, 000h, 000h, 0ffh, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 03fh, 000h, 000h,
000h
 52100 000128DD 3636363F000000      <1>
 52101 000128E4 18181F181F00000000- <1>     db  018h, 018h, 01fh, 018h, 01fh, 000h, 000h, 000h, 000h, 000h, 01fh, 018h, 01fh, 018h, 018h,
018h
 52102 000128ED 001F181F181818      <1>
 52103 000128F4 000000003F36363636- <1>     db  000h, 000h, 000h, 000h, 03fh, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 0ffh, 036h, 036h,
036h
 52104 000128FD 363636FF363636      <1>
 52105 00012904 1818FF18FF18181818- <1>     db  018h, 018h, 0ffh, 018h, 0ffh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 0f8h, 000h, 000h,
000h
 52106 0001290D 181818F8000000      <1>
 52107 00012914 000000001F181818FF- <1>     db  000h, 000h, 000h, 000h, 01fh, 018h, 018h, 018h, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh,
0ffh
 52108 0001291D FFFFFFFFFFFFFF      <1>
 52109 00012924 00000000FFFFFFFFF0- <1>     db  000h, 000h, 000h, 000h, 0ffh, 0ffh, 0ffh, 0ffh, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h,
0f0h
 52110 0001292D F0F0F0F0F0F0F0      <1>
 52111 00012934 0F0F0F0F0F0F0F0FFF- <1>     db  00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 0ffh, 0ffh, 0ffh, 0ffh, 000h, 000h, 000h,
000h
 52112 0001293D FFFFFF00000000      <1>
 52113 00012944 000076DCC8DC760000- <1>     db  000h, 000h, 076h, 0dch, 0c8h, 0dch, 076h, 000h, 000h, 078h, 0cch, 0f8h, 0cch, 0f8h, 0c0h,
0c0h
 52114 0001294D 78CCF8CCF8C0C0      <1>
```

```
 52115 00012954 00FCCCC0C0C0C00000- <1>     db  000h, 0fch, 0cch, 0c0h, 0c0h, 0c0h, 0c0h, 000h, 000h, 0feh, 06ch, 06ch, 06ch, 06ch, 06ch,
000h
 52116 0001295D FE6C6C6C6C6C00      <1>
 52117 00012964 FCCC603060CCFC0000- <1>     db  0fch, 0cch, 060h, 030h, 060h, 0cch, 0fch, 000h, 000h, 000h, 07eh, 0d8h, 0d8h, 0d8h, 070h,
000h
 52118 0001296D 007ED8D8D87000      <1>
 52119 00012974 00666666667C60C000- <1>     db  000h, 066h, 066h, 066h, 066h, 07ch, 060h, 0c0h, 000h, 076h, 0dch, 018h, 018h, 018h, 018h,
000h
 52120 0001297D 76DC1818181800      <1>
 52121 00012984 FC3078CCCC7830FC38- <1>     db  0fch, 030h, 078h, 0cch, 0cch, 078h, 030h, 0fch, 038h, 06ch, 0c6h, 0feh, 0c6h, 06ch, 038h,
000h
 52122 0001298D 6CC6FEC66C3800      <1>
 52123 00012994 386CC6C66C6CEE001C- <1>     db  038h, 06ch, 0c6h, 0c6h, 06ch, 06ch, 0eeh, 000h, 01ch, 030h, 018h, 07ch, 0cch, 0cch,
078h, 000h
 52124 0001299D 30187CCCCC7800      <1>
 52125 000129A4 00007EDBDB7E000006- <1>     db  000h, 000h, 07eh, 0dbh, 0dbh, 07eh, 000h, 000h, 006h, 00ch, 07eh, 0dbh, 0dbh, 07eh, 060h,
0c0h
 52126 000129AD 0C7EDBDB7E60C0      <1>
 52127 000129B4 3860C0F8C060380078- <1>     db  038h, 060h, 0c0h, 0f8h, 0c0h, 060h, 038h, 000h, 078h, 0cch, 0cch, 0cch, 0cch, 0cch, 0cch,
000h
 52128 000129BD CCCCCCCCCCCC00      <1>
 52129 000129C4 00FC00FC00FC000030- <1>     db  000h, 0fch, 000h, 0fch, 000h, 0fch, 000h, 000h, 030h, 030h, 0fch, 030h, 030h, 000h, 0fch,
000h
 52130 000129CD 30FC303000FC00      <1>
 52131 000129D4 603018306000FC0018- <1>     db  060h, 030h, 018h, 030h, 060h, 000h, 0fch, 000h, 018h, 030h, 060h, 030h, 018h, 000h, 0fch,
000h
 52132 000129DD 3060301800FC00      <1>
 52133 000129E4 0E1B1B181818181818- <1>     db  00eh, 01bh, 01bh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 0d8h, 0d8h,
070h
 52134 000129ED 18181818D8D870      <1>
 52135 000129F4 303000FC0030300000- <1>     db  030h, 030h, 000h, 0fch, 000h, 030h, 030h, 000h, 000h, 076h, 0dch, 000h, 076h, 0dch, 000h,
000h
 52136 000129FD 76DC0076DC0000      <1>
 52137 00012A04 386C6C380000000000- <1>     db  038h, 06ch, 06ch, 038h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 018h, 018h, 000h, 000h,
000h
 52138 00012A0D 00001818000000      <1>
 52139 00012A14 0000000180000000F- <1>     db  000h, 000h, 000h, 000h, 018h, 000h, 000h, 000h, 00fh, 00ch, 00ch, 00ch, 0ech, 06ch, 03ch,
01ch
 52140 00012A1D 0C0C0CEC6C3C1C      <1>
 52141 00012A24 786C6C6C6C00000070- <1>     db  078h, 06ch, 06ch, 06ch, 06ch, 000h, 000h, 000h, 070h, 018h, 030h, 060h, 078h, 000h, 000h,
000h
 52142 00012A2D 18306078000000      <1>
 52143 00012A34 00003C3C3C3C000000- <1>     db  000h, 000h, 03ch, 03ch, 03ch, 03ch, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 52144 00012A3D 00000000000000      <1>
 52145                              <1> vgafont14:
 52146 00012A44 0000000000000000- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 52147 00012A4D 00000000000000      <1>
 52148 00012A54 7E81A58181BD99817E- <1>     db  07eh, 081h, 0a5h, 081h, 081h, 0bdh, 099h, 081h, 07eh, 000h, 000h, 000h, 000h, 000h, 07eh,
0ffh
 52149 00012A5D 00000000007EFF      <1>
 52150 00012A64 DBFFFFC3E7FF7E0000- <1>     db  0dbh, 0ffh, 0ffh, 0c3h, 0e7h, 0ffh, 07eh, 000h, 000h, 000h, 000h, 000h, 000h, 06ch, 0feh,
0feh
 52151 00012A6D 000000006CFEFE      <1>
 52152 00012A74 FEFE7C381000000000- <1>     db  0feh, 0feh, 07ch, 038h, 010h, 000h, 000h, 000h, 000h, 000h, 000h, 010h, 038h, 07ch, 0feh,
07ch
 52153 00012A7D 000010387CFE7C      <1>
 52154 00012A84 3810000000000000018- <1>     db  038h, 010h, 000h, 000h, 000h, 000h, 000h, 000h, 018h, 03ch, 03ch, 0e7h, 0e7h, 0e7h, 018h,
018h
 52155 00012A8D 3C3CE7E7E71818      <1>
 52156 00012A94 3C0000000000183C7E- <1>     db  03ch, 000h, 000h, 000h, 000h, 000h, 018h, 03ch, 07eh, 0ffh, 0ffh, 07eh, 018h, 018h, 03ch,
000h
 52157 00012A9D FFFF7E18183C00      <1>
 52158 00012AA4 0000000000000183C- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 018h, 03ch, 03ch, 018h, 000h, 000h, 000h, 000h,
000h
 52159 00012AAD 3C180000000000      <1>
 52160 00012AB4 FFFFFFFFFFE7C3C3E7- <1>     db  0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0e7h, 0c3h, 0c3h, 0e7h, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 000h,
000h
 52161 00012ABD FFFFFFFFFF0000      <1>
 52162 00012AC4 00003C664242663C00- <1>     db  000h, 000h, 03ch, 066h, 042h, 042h, 066h, 03ch, 000h, 000h, 000h, 000h, 0ffh, 0ffh, 0ffh,
0ffh
 52163 00012ACD 000000FFFFFFFF      <1>
 52164 00012AD4 C399BDBD99C3FFFFFF- <1>     db  0c3h, 099h, 0bdh, 0bdh, 099h, 0c3h, 0ffh, 0ffh, 0ffh, 0ffh, 000h, 000h, 01eh, 00eh, 01ah,
032h
 52165 00012ADD FF00001E0E1A32      <1>
 52166 00012AE4 78CCCCCC7800000000- <1>     db  078h, 0cch, 0cch, 0cch, 078h, 000h, 000h, 000h, 000h, 000h, 03ch, 066h, 066h, 066h, 03ch,
018h
 52167 00012AED 003C6666663C18      <1>
 52168 00012AF4 7E181800000000003F- <1>     db  07eh, 018h, 018h, 000h, 000h, 000h, 000h, 000h, 03fh, 033h, 03fh, 030h, 030h, 030h, 070h,
0f0h
 52169 00012AFD 333F30303070F0      <1>
 52170 00012B04 E000000000007F637F- <1>     db  0e0h, 000h, 000h, 000h, 000h, 000h, 07fh, 063h, 07fh, 063h, 063h, 063h, 067h, 0e7h, 0e6h,
0c0h
 52171 00012B0D 63636367E7E6C0      <1>
 52172 00012B14 000000001818DB3CE7- <1>     db  000h, 000h, 000h, 000h, 018h, 018h, 0dbh, 03ch, 0e7h, 03ch, 0dbh, 018h, 018h, 000h, 000h,
000h
 52173 00012B1D 3CDB1818000000      <1>
 52174 00012B24 000080C0E0F8FEF8E0- <1>     db  000h, 000h, 080h, 0c0h, 0e0h, 0f8h, 0feh, 0f8h, 0e0h, 0c0h, 080h, 000h, 000h, 000h, 000h,
000h
 52175 00012B2D C0800000000000      <1>
 52176 00012B34 02060E3EFE3E0E0602- <1>     db  002h, 006h, 00eh, 03eh, 0feh, 03eh, 00eh, 006h, 002h, 000h, 000h, 000h, 000h, 000h, 018h,
03ch
 52177 00012B3D 0000000000183C      <1>
 52178 00012B44 7E1818187E3C180000- <1>     db  07eh, 018h, 018h, 018h, 07eh, 03ch, 018h, 000h, 000h, 000h, 000h, 000h, 066h, 066h, 066h,
066h
 52179 00012B4D 00000066666666      <1>
 52180 00012B54 666600666600000000- <1>     db  066h, 066h, 000h, 066h, 066h, 000h, 000h, 000h, 000h, 000h, 07fh, 0dbh, 0dbh, 0dbh, 07bh,
01bh
 52181 00012B5D 007FDBDBDB7B1B      <1>
 52182 00012B64 1B1B1B000000007CC6- <1>     db  01bh, 01bh, 01bh, 000h, 000h, 000h, 000h, 07ch, 0c6h, 060h, 038h, 06ch, 0c6h, 0c6h, 06ch,
038h
```

```
 52183 00012B6D 60386CC6C66C38      <1>
 52184 00012B74 0CC67C000000000000- <1>     db  00ch, 0c6h, 07ch, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0feh, 0feh, 0feh,
000h
 52185 00012B7D 000000FEFEFE00      <1>
 52186 00012B84 00000000183C7E1818- <1>     db  000h, 000h, 000h, 000h, 018h, 03ch, 07eh, 018h, 018h, 018h, 07eh, 03ch, 018h, 07eh, 000h,
000h
 52187 00012B8D 187E3C187E0000      <1>
 52188 00012B94 0000183C7E18181818- <1>     db  000h, 000h, 018h, 03ch, 07eh, 018h, 018h, 018h, 018h, 018h, 018h, 000h, 000h, 000h, 000h,
000h
 52189 00012B9D 18180000000000      <1>
 52190 00012BA4 1818181818187E3C18- <1>     db  018h, 018h, 018h, 018h, 018h, 018h, 07eh, 03ch, 018h, 000h, 000h, 000h, 000h, 000h,
000h, 000h
 52191 00012BAD 00000000000000      <1>
 52192 00012BB4 180CFE0C1800000000- <1>     db  018h, 00ch, 0feh, 00ch, 018h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 030h,
060h
 52193 00012BBD 00000000003060      <1>
 52194 00012BC4 FE6030000000000000- <1>     db  0feh, 060h, 030h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0c0h, 0c0h,
0c0h
 52195 00012BCD 00000000C0C0C0      <1>
 52196 00012BD4 FE0000000000000000- <1>     db  0feh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 028h, 06ch, 0feh, 06ch, 028h,
000h
 52197 00012BDD 00286CFE6C2800      <1>
 52198 00012BE4 000000000000001038- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 010h, 038h, 038h, 07ch, 07ch, 0feh, 0feh, 000h,
000h
 52199 00012BED 387C7CFEFE0000      <1>
 52200 00012BF4 0000000000FEFE7C7C- <1>     db  000h, 000h, 000h, 000h, 000h, 0feh, 0feh, 07ch, 07ch, 038h, 038h, 010h, 000h, 000h, 000h,
000h
 52201 00012BFD 38381000000000      <1>
 52202 00012C04 000000000000000000- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 52203 00012C0D 00000000000000      <1>
 52204 00012C14 183C3C3C1818001818- <1>     db  018h, 03ch, 03ch, 03ch, 018h, 018h, 000h, 018h, 018h, 000h, 000h, 000h, 000h, 066h, 066h,
066h
 52205 00012C1D 00000000666666      <1>
 52206 00012C24 240000000000000000- <1>     db  024h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 06ch, 06ch, 0feh,
06ch
 52207 00012C2D 0000006C6CFE6C      <1>
 52208 00012C34 6C6CFE6C6C00000018- <1>     db  06ch, 06ch, 0feh, 06ch, 06ch, 000h, 000h, 000h, 018h, 018h, 07ch, 0c6h, 0c2h, 0c0h, 07ch,
006h
 52209 00012C3D 187CC6C2C07C06      <1>
 52210 00012C44 86C67C181800000000- <1>     db  086h, 0c6h, 07ch, 018h, 018h, 000h, 000h, 000h, 000h, 000h, 0c2h, 0c6h, 00ch, 018h, 030h,
066h
 52211 00012C4D 00C2C60C183066      <1>
 52212 00012C54 C60000000000386C6C- <1>     db  0c6h, 000h, 000h, 000h, 000h, 000h, 038h, 06ch, 06ch, 038h, 076h, 0dch, 0cch, 0cch, 076h,
000h
 52213 00012C5D 3876DCCCCC7600      <1>
 52214 00012C64 000003030306000000- <1>     db  000h, 000h, 000h, 030h, 030h, 030h, 060h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 52215 00012C6D 00000000000000      <1>
 52216 00012C74 00000C183030303030- <1>     db  000h, 000h, 00ch, 018h, 030h, 030h, 030h, 030h, 030h, 018h, 00ch, 000h, 000h, 000h, 000h,
000h
 52217 00012C7D 180C0000000000      <1>
 52218 00012C84 30180C0C0C0C0C1830- <1>     db  030h, 018h, 00ch, 00ch, 00ch, 00ch, 00ch, 018h, 030h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 52219 00012C8D 00000000000000      <1>
 52220 00012C94 663CFF3C6600000000- <1>     db  066h, 03ch, 0ffh, 03ch, 066h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 018h,
018h
 52221 00012C9D 00000000001818      <1>
 52222 00012CA4 7E1818000000000000- <1>     db  07eh, 018h, 018h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 52223 00012CAD 00000000000000      <1>
 52224 00012CB4 181818300000000000- <1>     db  018h, 018h, 018h, 030h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0feh, 000h, 000h,
000h
 52225 00012CBD 000000FE000000      <1>
 52226 00012CC4 000000000000000000- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 018h, 018h,
000h
 52227 00012CCD 00000000181800      <1>
 52228 00012CD4 0000000002060C1830- <1>     db  000h, 000h, 000h, 000h, 002h, 006h, 00ch, 018h, 030h, 060h, 0c0h, 080h, 000h, 000h, 000h,
000h
 52229 00012CDD 60C08000000000      <1>
 52230 00012CE4 00007CC6CEDEF6E6C6- <1>     db  000h, 000h, 07ch, 0c6h, 0ceh, 0deh, 0f6h, 0e6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h,
000h
 52231 00012CED C67C0000000000      <1>
 52232 00012CF4 183878181818187E- <1>     db  018h, 038h, 078h, 018h, 018h, 018h, 018h, 018h, 07eh, 000h, 000h, 000h, 000h, 000h, 07ch,
0c6h
 52233 00012CFD 00000000007CC6      <1>
 52234 00012D04 060C183060C6FE0000- <1>     db  006h, 00ch, 018h, 030h, 060h, 0c6h, 0feh, 000h, 000h, 000h, 000h, 000h, 07ch, 0c6h, 006h,
006h
 52235 00012D0D 0000007CC60606      <1>
 52236 00012D14 3C0606C67C00000000- <1>     db  03ch, 006h, 006h, 0c6h, 07ch, 000h, 000h, 000h, 000h, 000h, 00ch, 01ch, 03ch, 06ch, 0cch,
0feh
 52237 00012D1D 000C1C3C6CCCFE      <1>
 52238 00012D24 0C0C1E0000000000FE- <1>     db  00ch, 00ch, 01eh, 000h, 000h, 000h, 000h, 000h, 0feh, 0c0h, 0c0h, 0c0h, 0fch, 006h, 006h,
0c6h
 52239 00012D2D C0C0C0FC0606C6      <1>
 52240 00012D34 7C00000000003860C0- <1>     db  07ch, 000h, 000h, 000h, 000h, 000h, 038h, 060h, 0c0h, 0c0h, 0fch, 0c6h, 0c6h, 0c6h, 07ch,
000h
 52241 00012D3D C0FCC6C6C67C00      <1>
 52242 00012D44 00000000FEC6060C18- <1>     db  000h, 000h, 000h, 000h, 0feh, 0c6h, 006h, 00ch, 018h, 030h, 030h, 030h, 030h, 000h, 000h,
000h
 52243 00012D4D 30303030000000      <1>
 52244 00012D54 00007CC6C6C67CC6C6- <1>     db  000h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 07ch, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h,
000h
 52245 00012D5D C67C0000000000      <1>
 52246 00012D64 7CC6C6C67E06060C78- <1>     db  07ch, 0c6h, 0c6h, 0c6h, 07eh, 006h, 006h, 00ch, 078h, 000h, 000h, 000h, 000h, 000h, 000h,
018h
 52247 00012D6D 00000000000018      <1>
 52248 00012D74 180000001818000000- <1>     db  018h, 000h, 000h, 000h, 018h, 018h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 018h, 018h,
000h
 52249 00012D7D 00000000181800      <1>
 52250 00012D84 000018183000000000- <1>     db  000h, 000h, 018h, 018h, 030h, 000h, 000h, 000h, 000h, 000h, 006h, 00ch, 018h, 030h, 060h,
030h
```

```
 52251 00012D8D 00060C18306030      <1>
 52252 00012D94 180C060000000000000- <1>     db  018h, 00ch, 006h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 07eh, 000h, 000h, 07eh,
000h
 52253 00012D9D 00007E00007E00      <1>
 52254 00012DA4 0000000000000603018- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 060h, 030h, 018h, 00ch, 006h, 00ch, 018h, 030h, 060h,
000h
 52255 00012DAD 0C060C18306000      <1>
 52256 00012DB4 000000007CC6C60C18- <1>     db  000h, 000h, 000h, 000h, 07ch, 0c6h, 0c6h, 00ch, 018h, 018h, 000h, 018h, 018h, 000h, 000h,
000h
 52257 00012DBD 18001818000000      <1>
 52258 00012DC4 00007CC6C6DEDEDEDC- <1>     db  000h, 000h, 07ch, 0c6h, 0c6h, 0deh, 0deh, 0deh, 0dch, 0c0h, 07ch, 000h, 000h, 000h,
000h, 000h
 52259 00012DCD C07C0000000000      <1>
 52260 00012DD4 10386CC6C6FEC6C6C6- <1>     db  010h, 038h, 06ch, 0c6h, 0c6h, 0feh, 0c6h, 0c6h, 0c6h, 000h, 000h, 000h, 000h, 000h, 0fch,
066h
 52261 00012DDD 0000000000FC66      <1>
 52262 00012DE4 66667C666666FC0000- <1>     db  066h, 066h, 07ch, 066h, 066h, 066h, 0fch, 000h, 000h, 000h, 000h, 000h, 03ch, 066h, 0c2h,
0c0h
 52263 00012DED 0000003C66C2C0      <1>
 52264 00012DF4 C0C0C2663C00000000- <1>     db  0c0h, 0c0h, 0c2h, 066h, 03ch, 000h, 000h, 000h, 000h, 000h, 0f8h, 06ch, 066h, 066h, 066h,
066h
 52265 00012DFD 00F86C66666666      <1>
 52266 00012E04 666CF800000000000FE- <1>     db  066h, 06ch, 0f8h, 000h, 000h, 000h, 000h, 000h, 0feh, 066h, 062h, 068h, 078h, 068h, 062h,
066h
 52267 00012E0D 66626878686266      <1>
 52268 00012E14 FE0000000000FE6662- <1>     db  0feh, 000h, 000h, 000h, 000h, 000h, 0feh, 066h, 062h, 068h, 078h, 068h, 060h, 060h, 0f0h,
000h
 52269 00012E1D 6878686060F000      <1>
 52270 00012E24 000000003C66C2C0C0- <1>     db  000h, 000h, 000h, 000h, 03ch, 066h, 0c2h, 0c0h, 0c0h, 0deh, 0c6h, 066h, 03ah, 000h, 000h,
000h
 52271 00012E2D DEC6663A000000      <1>
 52272 00012E34 0000C6C6C6C6FEC6C6- <1>     db  000h, 000h, 0c6h, 0c6h, 0c6h, 0c6h, 0feh, 0c6h, 0c6h, 0c6h, 0c6h, 000h, 000h, 000h, 000h,
000h
 52273 00012E3D C6C60000000000      <1>
 52274 00012E44 3C1818181818183C- <1>     db  03ch, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h, 000h, 000h, 01eh,
00ch
 52275 00012E4D 00000000001E0C      <1>
 52276 00012E54 0C0C0C0CCCCC780000- <1>     db  00ch, 00ch, 00ch, 00ch, 0cch, 0cch, 078h, 000h, 000h, 000h, 000h, 000h, 0e6h, 066h, 06ch,
06ch
 52277 00012E5D 000000E6666C6C      <1>
 52278 00012E64 786C6C66E600000000- <1>     db  078h, 06ch, 06ch, 066h, 0e6h, 000h, 000h, 000h, 000h, 000h, 0f0h, 060h, 060h, 060h, 060h,
060h
 52279 00012E6D 00F06060606060      <1>
 52280 00012E74 6266FE0000000000C6- <1>     db  062h, 066h, 0feh, 000h, 000h, 000h, 000h, 000h, 0c6h, 0eeh, 0feh, 0feh, 0d6h, 0c6h, 0c6h,
0c6h
 52281 00012E7D EEFEFED6C6C6C6      <1>
 52282 00012E84 C600000000000C6E6F6- <1>     db  0c6h, 000h, 000h, 000h, 000h, 000h, 0c6h, 0e6h, 0f6h, 0feh, 0deh, 0ceh, 0c6h, 0c6h, 0c6h,
000h
 52283 00012E8D FEDECEC6C6C600      <1>
 52284 00012E94 00000000386CC6C6C6- <1>     db  000h, 000h, 000h, 000h, 038h, 06ch, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 06ch, 038h, 000h, 000h,
000h
 52285 00012E9D C6C66C38000000      <1>
 52286 00012EA4 0000FC6666667C6060- <1>     db  000h, 000h, 0fch, 066h, 066h, 066h, 07ch, 060h, 060h, 060h, 0f0h, 000h, 000h, 000h, 000h,
000h
 52287 00012EAD 60F00000000000      <1>
 52288 00012EB4 7CC6C6C6C6D6DE7C0C- <1>     db  07ch, 0c6h, 0c6h, 0c6h, 0c6h, 0d6h, 0deh, 07ch, 00ch, 00eh, 000h, 000h, 000h, 000h, 0fch,
066h
 52289 00012EBD 0E00000000FC66      <1>
 52290 00012EC4 66667C6C6666E60000- <1>     db  066h, 066h, 07ch, 06ch, 066h, 066h, 0e6h, 000h, 000h, 000h, 000h, 000h, 07ch, 0c6h, 0c6h,
060h
 52291 00012ECD 0000007CC6C660      <1>
 52292 00012ED4 380CC6C67C00000000- <1>     db  038h, 00ch, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h, 000h, 07eh, 07eh, 05ah, 018h, 018h,
018h
 52293 00012EDD 007E7E5A181818      <1>
 52294 00012EE4 18183C0000000000C6- <1>     db  018h, 018h, 03ch, 000h, 000h, 000h, 000h, 000h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h,
0c6h
 52295 00012EED C6C6C6C6C6C6C6      <1>
 52296 00012EF4 7C0000000000C6C6C6- <1>     db  07ch, 000h, 000h, 000h, 000h, 000h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 06ch, 038h, 010h,
000h
 52297 00012EFD C6C6C66C381000      <1>
 52298 00012F04 00000000C6C6C6C6D6- <1>     db  000h, 000h, 000h, 000h, 0c6h, 0c6h, 0c6h, 0c6h, 0d6h, 0d6h, 0feh, 07ch, 06ch, 000h, 000h,
000h
 52299 00012F0D D6FE7C6C000000      <1>
 52300 00012F14 0000C6C66C3838386C- <1>     db  000h, 000h, 0c6h, 0c6h, 06ch, 038h, 038h, 038h, 06ch, 0c6h, 0c6h, 000h, 000h, 000h, 000h,
000h
 52301 00012F1D C6C60000000000      <1>
 52302 00012F24 666666663C1818183C- <1>     db  066h, 066h, 066h, 066h, 03ch, 018h, 018h, 018h, 03ch, 000h, 000h, 000h, 000h, 000h, 0feh,
0c6h
 52303 00012F2D 0000000000FEC6      <1>
 52304 00012F34 8C183060C2C6FE0000- <1>     db  08ch, 018h, 030h, 060h, 0c2h, 0c6h, 0feh, 000h, 000h, 000h, 000h, 000h, 03ch, 030h, 030h,
030h
 52305 00012F3D 0000003C303030      <1>
 52306 00012F44 303030303C00000000- <1>     db  030h, 030h, 030h, 030h, 03ch, 000h, 000h, 000h, 000h, 000h, 080h, 0c0h, 0e0h, 070h, 038h,
01ch
 52307 00012F4D 0080C0E070381C      <1>
 52308 00012F54 0E0602000000000003C- <1>     db  00eh, 006h, 002h, 000h, 000h, 000h, 000h, 000h, 03ch, 00ch, 00ch, 00ch, 00ch, 00ch, 00ch,
00ch
 52309 00012F5D 0C0C0C0C0C0C0C      <1>
 52310 00012F64 3C00000010386CC600- <1>     db  03ch, 000h, 000h, 000h, 010h, 038h, 06ch, 0c6h, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 52311 00012F6D 00000000000000      <1>
 52312 00012F74 0000000000000000000- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0ffh,
000h
 52313 00012F7D 0000000000FF00      <1>
 52314 00012F84 3030180000000000000- <1>     db  030h, 030h, 018h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 52315 00012F8D 00000000000000      <1>
 52316 00012F94 000000780C7CCCCC76- <1>     db  000h, 000h, 000h, 078h, 00ch, 07ch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h, 000h, 0e0h,
060h
 52317 00012F9D 0000000000E060      <1>
 52318 00012FA4 60786C6666667C0000- <1>     db  060h, 078h, 06ch, 066h, 066h, 066h, 07ch, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
07ch
```

```
 52319 00012FAD 0000000000007C      <1>
 52320 00012FB4 C6C0C0C67C00000000- <1>     db  0c6h, 0c0h, 0c0h, 0c6h, 07ch, 000h, 000h, 000h, 000h, 000h, 01ch, 00ch, 00ch, 03ch, 06ch,
0cch
 52321 00012FBD 001C0C0C3C6CCC      <1>
 52322 00012FC4 CCCC760000000000000- <1>     db  0cch, 0cch, 076h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 07ch, 0c6h, 0feh, 0c0h,
0c6h
 52323 00012FCD 00007CC6FEC0C6      <1>
 52324 00012FD4 7C0000000000386C64- <1>     db  07ch, 000h, 000h, 000h, 000h, 000h, 038h, 06ch, 064h, 060h, 0f0h, 060h, 060h, 060h, 0f0h,
000h
 52325 00012FDD 60F0606060F000      <1>
 52326 00012FE4 0000000000000076CC- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 076h, 0cch, 0cch, 0cch, 07ch, 00ch, 0cch,
078h, 000h
 52327 00012FED CCCC7C0CCC7800      <1>
 52328 00012FF4 0000E060606C766666- <1>     db  000h, 000h, 0e0h, 060h, 060h, 06ch, 076h, 066h, 066h, 066h, 0e6h, 000h, 000h, 000h, 000h,
000h
 52329 00012FFD 66E60000000000      <1>
 52330 00013004 181800381818183C- <1>     db  018h, 018h, 000h, 038h, 018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h, 000h, 000h, 006h,
006h
 52331 0001300D 00000000000606      <1>
 52332 00013014 000E06060666663C- <1>     db  000h, 00eh, 006h, 006h, 006h, 006h, 066h, 066h, 03ch, 000h, 000h, 000h, 0e0h, 060h, 060h,
066h
 52333 0001301D 000000E0606066      <1>
 52334 00013024 6C786C66E600000000- <1>     db  06ch, 078h, 06ch, 066h, 0e6h, 000h, 000h, 000h, 000h, 000h, 038h, 018h, 018h, 018h, 018h,
018h
 52335 0001302D 00381818181818      <1>
 52336 00013034 18183C000000000000- <1>     db  018h, 018h, 03ch, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0ech, 0feh, 0d6h, 0d6h,
0d6h
 52337 0001303D 0000ECFED6D6D6      <1>
 52338 00013044 C600000000000000000- <1>     db  0c6h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0dch, 066h, 066h, 066h, 066h, 066h,
000h
 52339 0001304D DC666666666600      <1>
 52340 00013054 0000000000000076CC- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h,
000h
 52341 0001305D C6C6C67C000000      <1>
 52342 00013064 0000000000DC666666- <1>     db  000h, 000h, 000h, 000h, 000h, 0dch, 066h, 066h, 066h, 07ch, 060h, 060h, 0f0h, 000h, 000h,
000h
 52343 0001306D 7C6060F0000000      <1>
 52344 00013074 00000076CCCCCC7C0C- <1>     db  000h, 000h, 000h, 076h, 0cch, 0cch, 0cch, 07ch, 00ch, 00ch, 01eh, 000h, 000h, 000h, 000h,
000h
 52345 0001307D 0C1E0000000000      <1>
 52346 00013084 00DC76666060F00000- <1>     db  000h, 0dch, 076h, 066h, 060h, 060h, 0f0h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
07ch
 52347 0001308D 0000000000007C      <1>
 52348 00013094 C6701CC67C00000000- <1>     db  0c6h, 070h, 01ch, 0c6h, 07ch, 000h, 000h, 000h, 000h, 000h, 010h, 030h, 030h, 0fch, 030h,
030h
 52349 0001309D 00103030FC3030      <1>
 52350 000130A4 30361C000000000000- <1>     db  030h, 036h, 01ch, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0cch, 0cch, 0cch, 0cch,
0cch
 52351 000130AD 0000CCCCCCCCCC      <1>
 52352 000130B4 7600000000000000000- <1>     db  076h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 066h, 066h, 066h, 066h, 03ch, 018h,
000h
 52353 000130BD 666666663C1800      <1>
 52354 000130C4 000000000000000C6C6- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 0c6h, 0c6h, 0d6h, 0d6h, 0feh, 06ch, 000h, 000h,
000h
 52355 000130CD D6D6FE6C000000      <1>
 52356 000130D4 0000000000C66C3838- <1>     db  000h, 000h, 000h, 000h, 000h, 0c6h, 06ch, 038h, 038h, 06ch, 0c6h, 000h, 000h, 000h, 000h,
000h
 52357 000130DD 6CC60000000000      <1>
 52358 000130E4 000000C6C6C6C67E06- <1>     db  000h, 000h, 000h, 0c6h, 0c6h, 0c6h, 0c6h, 07eh, 006h, 00ch, 0f8h, 000h, 000h, 000h, 000h,
000h
 52359 000130ED 0CF80000000000      <1>
 52360 000130F4 00FECC183066FE0000- <1>     db  000h, 0feh, 0cch, 018h, 030h, 066h, 0feh, 000h, 000h, 000h, 000h, 000h, 00eh, 018h, 018h,
018h
 52361 000130FD 0000000E181818      <1>
 52362 00013104 701818180E00000000- <1>     db  070h, 018h, 018h, 018h, 00eh, 000h, 000h, 000h, 000h, 000h, 018h, 018h, 018h, 018h, 000h,
018h
 52363 0001310D 00181818180018      <1>
 52364 00013114 1818180000000070- <1>     db  018h, 018h, 018h, 000h, 000h, 000h, 000h, 000h, 070h, 018h, 018h, 018h, 00eh, 018h, 018h,
018h
 52365 0001311D 1818180E181818      <1>
 52366 00013124 70000000000076DC00- <1>     db  070h, 000h, 000h, 000h, 000h, 000h, 076h, 0dch, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 52367 0001312D 00000000000000      <1>
 52368 00013134 0000000000010386C- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 010h, 038h, 06ch, 0c6h, 0c6h, 0feh, 000h, 000h, 000h,
000h
 52369 0001313D C6C6FE00000000      <1>
 52370 00013144 00003C66C2C0C0C266- <1>     db  000h, 000h, 03ch, 066h, 0c2h, 0c0h, 0c0h, 0c2h, 066h, 03ch, 00ch, 006h, 07ch, 000h, 000h,
000h
 52371 0001314D 3C0C067C000000      <1>
 52372 00013154 CCCC00CCCCCCCCCC76- <1>     db  0cch, 0cch, 000h, 0cch, 0cch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h, 00ch, 018h,
030h
 52373 0001315D 000000000C1830      <1>
 52374 00013164 007CC6FEC0C67C0000- <1>     db  000h, 07ch, 0c6h, 0feh, 0c0h, 0c6h, 07ch, 000h, 000h, 000h, 000h, 010h, 038h, 06ch, 000h,
078h
 52375 0001316D 000010386C0078      <1>
 52376 00013174 0C7CCCCC7600000000- <1>     db  00ch, 07ch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h, 000h, 0cch, 0cch, 000h, 078h, 00ch,
07ch
 52377 0001317D 00CCCC00780C7C      <1>
 52378 00013184 CCCC76000000006030- <1>     db  0cch, 0cch, 076h, 000h, 000h, 000h, 000h, 060h, 030h, 018h, 000h, 078h, 00ch, 07ch, 0cch,
0cch
 52379 0001318D 1800780C7CCCCC      <1>
 52380 00013194 7600000000386C3800- <1>     db  076h, 000h, 000h, 000h, 000h, 038h, 06ch, 038h, 000h, 078h, 00ch, 07ch, 0cch, 0cch, 076h,
000h
 52381 0001319D 780C7CCCCC7600      <1>
 52382 000131A4 0000000000003C6660- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 03ch, 066h, 060h, 066h, 03ch, 00ch, 006h, 03ch, 000h,
000h
 52383 000131AD 663C0C063C0000      <1>
 52384 000131B4 0010386C007CC6FEC0- <1>     db  000h, 010h, 038h, 06ch, 000h, 07ch, 0c6h, 0feh, 0c0h, 0c6h, 07ch, 000h, 000h, 000h, 000h,
000h
 52385 000131BD C67C0000000000      <1>
 52386 000131C4 CCCC007CC6FEC0C67C- <1>     db  0cch, 0cch, 000h, 07ch, 0c6h, 0feh, 0c0h, 0c6h, 07ch, 000h, 000h, 000h, 000h, 060h, 030h,
018h
```

```
 52387 000131CD 00000000603018      <1>
 52388 000131D4 007CC6FEC0C67C0000- <1>     db  000h, 07ch, 0c6h, 0feh, 0c0h, 0c6h, 07ch, 000h, 000h, 000h, 000h, 000h, 066h, 066h, 000h,
038h
 52389 000131DD 00000066660038      <1>
 52390 000131E4 181818183C00000000- <1>     db  018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h, 000h, 018h, 03ch, 066h, 000h, 038h, 018h,
018h
 52391 000131ED 183C6600381818      <1>
 52392 000131F4 18183C000000006030- <1>     db  018h, 018h, 03ch, 000h, 000h, 000h, 000h, 060h, 030h, 018h, 000h, 038h, 018h, 018h, 018h,
018h
 52393 000131FD 18003818181818      <1>
 52394 00013204 3C00000000C6C61038- <1>     db  03ch, 000h, 000h, 000h, 000h, 0c6h, 0c6h, 010h, 038h, 06ch, 0c6h, 0c6h, 0feh, 0c6h,
0c6h, 000h
 52395 0001320D 6CC6C6FEC6C600      <1>
 52396 00013214 0000386C3800386CC6- <1>     db  000h, 000h, 038h, 06ch, 038h, 000h, 038h, 06ch, 0c6h, 0c6h, 0feh, 0c6h, 0c6h, 000h, 000h,
000h
 52397 0001321D C6FEC6C6000000      <1>
 52398 00013224 18306000FE66607C60- <1>     db  018h, 030h, 060h, 000h, 0feh, 066h, 060h, 07ch, 060h, 066h, 0feh, 000h, 000h, 000h, 000h,
000h
 52399 0001322D 66FE0000000000      <1>
 52400 00013234 0000CC76367ED8D86E- <1>     db  000h, 000h, 0cch, 076h, 036h, 07eh, 0d8h, 0d8h, 06eh, 000h, 000h, 000h, 000h, 000h, 03eh,
06ch
 52401 0001323D 00000000003E6C      <1>
 52402 00013244 CCCCFECCCCCCCE0000- <1>     db  0cch, 0cch, 0feh, 0cch, 0cch, 0cch, 0ceh, 000h, 000h, 000h, 000h, 010h, 038h, 06ch, 000h,
07ch
 52403 0001324D 000010386C007C      <1>
 52404 00013254 C6C6C6C67C00000000- <1>     db  0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h, 000h, 0c6h, 0c6h, 000h, 07ch, 0c6h,
0c6h
 52405 0001325D 00C6C6007CC6C6      <1>
 52406 00013264 C6C67C000000006030- <1>     db  0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h, 060h, 030h, 018h, 000h, 07ch, 0c6h, 0c6h, 0c6h,
0c6h
 52407 0001326D 18007CC6C6C6C6      <1>
 52408 00013274 7C000000003078CC00- <1>     db  07ch, 000h, 000h, 000h, 000h, 030h, 078h, 0cch, 000h, 0cch, 0cch, 0cch, 0cch, 0cch, 076h,
000h
 52409 0001327D CCCCCCCCCC7600      <1>
 52410 00013284 00000060301800CCCC- <1>     db  000h, 000h, 000h, 060h, 030h, 018h, 000h, 0cch, 0cch, 0cch, 0cch, 0cch, 076h, 000h, 000h,
000h
 52411 0001328D CCCCCC76000000      <1>
 52412 00013294 0000C6C600C6C6C6C6- <1>     db  000h, 000h, 0c6h, 0c6h, 000h, 0c6h, 0c6h, 0c6h, 0c6h, 07eh, 006h, 00ch, 078h, 000h, 000h,
0c6h
 52413 0001329D 7E060C780000C6      <1>
 52414 000132A4 C6386CC6C6C6C66C38- <1>     db  0c6h, 038h, 06ch, 0c6h, 0c6h, 0c6h, 0c6h, 06ch, 038h, 000h, 000h, 000h, 000h, 0c6h, 0c6h,
000h
 52415 000132AD 00000000C6C600      <1>
 52416 000132B4 C6C6C6C6C6C67C0000- <1>     db  0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h, 018h, 018h, 03ch, 066h,
060h
 52417 000132BD 000018183C6660      <1>
 52418 000132C4 60663C181800000000- <1>     db  060h, 066h, 03ch, 018h, 018h, 000h, 000h, 000h, 000h, 038h, 06ch, 064h, 060h, 0f0h, 060h,
060h
 52419 000132CD 386C6460F06060      <1>
 52420 000132D4 60E6FC000000000066- <1>     db  060h, 0e6h, 0fch, 000h, 000h, 000h, 000h, 000h, 066h, 066h, 03ch, 018h, 07eh, 018h, 07eh,
018h
 52421 000132DD 663C187E187E18      <1>
 52422 000132E4 1800000000F8CCCCF8- <1>     db  018h, 000h, 000h, 000h, 000h, 0f8h, 0cch, 0cch, 0f8h, 0c4h, 0cch, 0deh, 0cch, 0cch, 0c6h,
000h
 52423 000132ED C4CCDECCCCC600      <1>
 52424 000132F4 0000000E1B1818187E- <1>     db  000h, 000h, 000h, 00eh, 01bh, 018h, 018h, 018h, 07eh, 018h, 018h, 018h, 018h, 0d8h, 070h,
000h
 52425 000132FD 18181818D87000      <1>
 52426 00013304 0018306000780C7CCC- <1>     db  000h, 018h, 030h, 060h, 000h, 078h, 00ch, 07ch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h,
00ch
 52427 0001330D CC76000000000C      <1>
 52428 00013314 183000381818183C- <1>     db  018h, 030h, 000h, 038h, 018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h, 000h, 018h, 030h,
060h
 52429 0001331D 00000000183060      <1>
 52430 00013324 007CC6C6C6C67C0000- <1>     db  000h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h, 018h, 030h, 060h, 000h,
0cch
 52431 0001332D 000018306000CC      <1>
 52432 00013334 CCCCCCCC7600000000- <1>     db  0cch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h, 000h, 076h, 0dch, 000h, 0dch, 066h,
066h
 52433 0001333D 0076DC00DC6666      <1>
 52434 00013344 66666600000076DC00- <1>     db  066h, 066h, 066h, 000h, 000h, 000h, 076h, 0dch, 000h, 0c6h, 0e6h, 0f6h, 0feh, 0deh, 0ceh,
0c6h
 52435 0001334D C6E6F6FEDECEC6      <1>
 52436 00013354 C6000000003C6C6C3E- <1>     db  0c6h, 000h, 000h, 000h, 000h, 03ch, 06ch, 06ch, 03eh, 000h, 07eh, 000h, 000h, 000h, 000h,
000h
 52437 0001335D 007E0000000000      <1>
 52438 00013364 000000386C6C38007C- <1>     db  000h, 000h, 000h, 038h, 06ch, 06ch, 038h, 000h, 07ch, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 52439 0001336D 00000000000000      <1>
 52440 00013374 0000303000303060C6- <1>     db  000h, 000h, 030h, 030h, 000h, 030h, 030h, 060h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h,
000h
 52441 0001337D C67C0000000000      <1>
 52442 00013384 00000000FEC0C0C000- <1>     db  000h, 000h, 000h, 000h, 0feh, 0c0h, 0c0h, 0c0h, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 52443 0001338D 00000000000000      <1>
 52444 00013394 0000FE060606000000- <1>     db  000h, 000h, 0feh, 006h, 006h, 006h, 000h, 000h, 000h, 000h, 000h, 0c0h, 0c0h, 0c6h, 0cch,
0d8h
 52445 0001339D 0000C0C0C6CCD8      <1>
 52446 000133A4 3060DC860C183E0000- <1>     db  030h, 060h, 0dch, 086h, 00ch, 018h, 03eh, 000h, 000h, 0c0h, 0c0h, 0c6h, 0cch, 0d8h, 030h,
066h
 52447 000133AD C0C0C6CCD83066      <1>
 52448 000133B4 CE9E3E060600000018- <1>     db  0ceh, 09eh, 03eh, 006h, 006h, 000h, 000h, 000h, 018h, 018h, 000h, 018h, 018h, 03ch, 03ch,
03ch
 52449 000133BD 180018183C3C3C      <1>
 52450 000133C4 1800000000000000036- <1>     db  018h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 036h, 06ch, 0d8h, 06ch, 036h, 000h, 000h,
000h
 52451 000133CD 6CD86C36000000      <1>
 52452 000133D4 0000000000000D86C36- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 0d8h, 06ch, 036h, 06ch, 0d8h, 000h, 000h, 000h, 000h,
000h
 52453 000133DD 6CD80000000000      <1>
 52454 000133E4 114411441144114411- <1>     db  011h, 044h, 011h, 044h, 011h, 044h, 011h, 044h, 011h, 044h, 011h, 044h, 011h, 044h, 055h,
0aah
```

```
 52455 000133ED 441144114455AA      <1>
 52456 000133F4 55AA55AA55AA55AA55- <1>     db  055h, 0aah, 055h, 0aah, 055h, 0aah, 055h, 0aah, 055h, 0aah, 055h, 0aah, 0ddh, 077h, 0ddh,
077h
 52457 000133FD AA55AADD77DD77      <1>
 52458 00013404 DD77DD77DD77DD77DD- <1>     db  0ddh, 077h, 0ddh, 077h, 0ddh, 077h, 0ddh, 077h, 0ddh, 077h, 018h, 018h, 018h, 018h, 018h,
018h
 52459 0001340D 77181818181818      <1>
 52460 00013414 181818181818181818- <1>     db  018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h,
0f8h
 52461 0001341D 181818181818F8      <1>
 52462 00013424 181818181818181818- <1>     db  018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 0f8h, 018h, 0f8h,
018h, 018h
 52463 0001342D 1818F818F81818      <1>
 52464 00013434 181818183636363636- <1>     db  018h, 018h, 018h, 018h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 0f6h, 036h, 036h, 036h,
036h
 52465 0001343D 3636F636363636      <1>
 52466 00013444 363600000000000000- <1>     db  036h, 036h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0feh, 036h, 036h, 036h, 036h, 036h,
036h
 52467 0001344D FE363636363636      <1>
 52468 00013454 0000000000F818F818- <1>     db  000h, 000h, 000h, 000h, 000h, 0f8h, 018h, 0f8h, 018h, 018h, 018h, 018h, 018h, 018h, 036h,
036h
 52469 0001345D 18181818183636      <1>
 52470 00013464 363636F606F6363636- <1>     db  036h, 036h, 036h, 0f6h, 006h, 0f6h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h,
036h
 52471 0001346D 36363636363636      <1>
 52472 00013474 363636363636363636- <1>     db  036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 000h, 000h, 000h, 000h, 000h,
0feh
 52473 0001347D 360000000000FE      <1>
 52474 00013484 06F636363636363636- <1>     db  006h, 0f6h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 0f6h, 006h,
0feh
 52475 0001348D 36363636F606FE      <1>
 52476 00013494 000000000000363636- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 0feh, 000h,
000h
 52477 0001349D 36363636FE0000      <1>
 52478 000134A4 000000001818181818- <1>     db  000h, 000h, 000h, 000h, 018h, 018h, 018h, 018h, 018h, 0f8h, 018h, 0f8h, 000h, 000h, 000h,
000h
 52479 000134AD F818F800000000      <1>
 52480 000134B4 000000000000000000- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0f8h, 018h, 018h, 018h, 018h, 018h,
018h
 52481 000134BD F8181818181818      <1>
 52482 000134C4 181818181818181F00- <1>     db  018h, 018h, 018h, 018h, 018h, 018h, 018h, 01fh, 000h, 000h, 000h, 000h, 000h, 000h, 018h,
018h
 52483 000134CD 00000000001818      <1>
 52484 000134D4 1818181818FF000000- <1>     db  018h, 018h, 018h, 018h, 018h, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 52485 000134DD 00000000000000      <1>
 52486 000134E4 000000FF1818181818- <1>     db  000h, 000h, 000h, 0ffh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h,
018h
 52487 000134ED 18181818181818      <1>
 52488 000134F4 181F18181818181800- <1>     db  018h, 01fh, 018h, 018h, 018h, 018h, 018h, 018h, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
0ffh
 52489 000134FD 000000000000FF      <1>
 52490 00013504 000000000000181818- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 0ffh, 018h,
018h
 52491 0001350D 18181818FF1818      <1>
 52492 00013514 181818181818181818- <1>     db  018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 01fh, 018h, 01fh, 018h, 018h, 018h,
018h
 52493 0001351D 1F181F18181818      <1>
 52494 00013524 181836363636363636- <1>     db  018h, 018h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 037h, 036h, 036h, 036h, 036h, 036h,
036h
 52495 0001352D 37363636363636      <1>
 52496 00013534 363636363637303F00- <1>     db  036h, 036h, 036h, 036h, 036h, 037h, 030h, 03fh, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 52497 0001353D 00000000000000      <1>
 52498 00013544 0000003F3037363636- <1>     db  000h, 000h, 000h, 03fh, 030h, 037h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h,
036h
 52499 0001354D 36363636363636      <1>
 52500 00013554 36F700FF0000000000- <1>     db  036h, 0f7h, 000h, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
0ffh
 52501 0001355D 000000000000FF      <1>
 52502 00013564 00F736363636363636- <1>     db  000h, 0f7h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 037h, 030h,
037h
 52503 0001356D 36363636373037      <1>
 52504 00013574 363636363636000000- <1>     db  036h, 036h, 036h, 036h, 036h, 036h, 000h, 000h, 000h, 000h, 000h, 0ffh, 000h, 0ffh, 000h,
000h
 52505 0001357D 0000FF00FF0000      <1>
 52506 00013584 000000003636363636- <1>     db  000h, 000h, 000h, 000h, 036h, 036h, 036h, 036h, 036h, 0f7h, 000h, 0f7h, 036h, 036h, 036h,
036h
 52507 0001358D F700F736363636      <1>
 52508 00013594 36361818181818FF00- <1>     db  036h, 036h, 018h, 018h, 018h, 018h, 018h, 0ffh, 000h, 0ffh, 000h, 000h, 000h, 000h, 000h,
000h
 52509 0001359D FF000000000000      <1>
 52510 000135A4 363636363636FF00- <1>     db  036h, 036h, 036h, 036h, 036h, 036h, 036h, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 52511 000135AD 00000000000000      <1>
 52512 000135B4 000000FF00FF181818- <1>     db  000h, 000h, 000h, 0ffh, 000h, 0ffh, 018h, 018h, 018h, 018h, 018h, 018h, 000h, 000h, 000h,
000h
 52513 000135BD 18181800000000      <1>
 52514 000135C4 000000FF3636363636- <1>     db  000h, 000h, 000h, 0ffh, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h,
036h
 52515 000135CD 36363636363636      <1>
 52516 000135D4 363F00000000000018- <1>     db  036h, 03fh, 000h, 000h, 000h, 000h, 000h, 000h, 018h, 018h, 018h, 018h, 018h, 01fh, 018h,
01fh
 52517 000135DD 181818181F181F      <1>
 52518 000135E4 000000000000000000- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 01fh, 018h, 01fh, 018h,
018h
 52519 000135ED 00001F181F1818      <1>
 52520 000135F4 181818180000000000- <1>     db  018h, 018h, 018h, 018h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 03fh, 036h, 036h, 036h,
036h
 52521 000135FD 00003F36363636      <1>
 52522 00013604 363636363636363636- <1>     db  036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 0ffh, 036h, 036h, 036h, 036h, 036h,
036h
```

```
 52523 0001360D FF363636363636      <1>
 52524 00013614 1818181818FF18FF18- <1>     db  018h, 018h, 018h, 018h, 018h, 0ffh, 018h, 0ffh, 018h, 018h, 018h, 018h, 018h, 018h, 018h,
018h
 52525 0001361D 18181818181818      <1>
 52526 00013624 1818181818F8000000- <1>     db  018h, 018h, 018h, 018h, 018h, 0f8h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 52527 0001362D 00000000000000      <1>
 52528 00013634 0000001F1818181818- <1>     db  000h, 000h, 000h, 01fh, 018h, 018h, 018h, 018h, 018h, 018h, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh,
0ffh
 52529 0001363D 18FFFFFFFFFFFF      <1>
 52530 00013644 FFFFFFFFFFFFFFFF00- <1>     db  0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h,
000h, 0ffh
 52531 0001364D 000000000000FF      <1>
 52532 00013654 FFFFFFFFFFFF0F0F0- <1>     db  0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h,
0f0h
 52533 0001365D F0F0F0F0F0F0F0      <1>
 52534 00013664 F0F0F0F00F0F0F0F0F- <1>     db  0f0h, 0f0h, 0f0h, 0f0h, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh,
00fh
 52535 0001366D 0F0F0F0F0F0F0F      <1>
 52536 00013674 0F0FFFFFFFFFFFFFFF- <1>     db  00fh, 00fh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 52537 0001367D 00000000000000      <1>
 52538 00013684 000000000076DCD8D8- <1>     db  000h, 000h, 000h, 000h, 000h, 076h, 0dch, 0d8h, 0d8h, 0dch, 076h, 000h, 000h, 000h, 000h,
000h
 52539 0001368D DC760000000000      <1>
 52540 00013694 00007CC6FCC6C6FCC0- <1>     db  000h, 000h, 07ch, 0c6h, 0fch, 0c6h, 0c6h, 0fch, 0c0h, 0c0h, 040h, 000h, 000h, 000h, 0feh,
0c6h
 52541 0001369D C040000000FEC6      <1>
 52542 000136A4 C6C0C0C0C0C0C00000- <1>     db  0c6h, 0c0h, 0c0h, 0c0h, 0c0h, 0c0h, 0c0h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0feh,
06ch
 52543 000136AD 0000000000FE6C      <1>
 52544 000136B4 6C6C6C6C6C00000000- <1>     db  06ch, 06ch, 06ch, 06ch, 06ch, 000h, 000h, 000h, 000h, 000h, 0feh, 0c6h, 060h, 030h, 018h,
030h
 52545 000136BD 00FEC660301830      <1>
 52546 000136C4 60C6FE000000000000- <1>     db  060h, 0c6h, 0feh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 07eh, 0d8h, 0d8h, 0d8h,
0d8h
 52547 000136CD 00007ED8D8D8D8      <1>
 52548 000136D4 7000000000000000066- <1>     db  070h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 066h, 066h, 066h, 066h, 07ch, 060h, 060h,
0c0h
 52549 000136DD 6666667C6060C0      <1>
 52550 000136E4 00000000000076DC18- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 076h, 0dch, 018h, 018h, 018h, 018h, 018h, 000h, 000h,
000h
 52551 000136ED 18181818000000      <1>
 52552 000136F4 00007E183C6666663C- <1>     db  000h, 000h, 07eh, 018h, 03ch, 066h, 066h, 066h, 03ch, 018h, 07eh, 000h, 000h, 000h, 000h,
000h
 52553 000136FD 187E0000000000      <1>
 52554 00013704 386CC6C6FEC6C66C38- <1>     db  038h, 06ch, 0c6h, 0c6h, 0feh, 0c6h, 0c6h, 06ch, 038h, 000h, 000h, 000h, 000h, 000h, 038h,
06ch
 52555 0001370D 0000000000386C      <1>
 52556 00013714 C6C6C66C6C6CEE0000- <1>     db  0c6h, 0c6h, 0c6h, 06ch, 06ch, 06ch, 0eeh, 000h, 000h, 000h, 000h, 000h, 01eh, 030h, 018h,
00ch
 52557 0001371D 0000001E30180C      <1>
 52558 00013724 3E6666663C00000000- <1>     db  03eh, 066h, 066h, 066h, 03ch, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 07eh, 0dbh,
0dbh
 52559 0001372D 000000007EDBDB      <1>
 52560 00013734 7E0000000000000003- <1>     db  07eh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 003h, 006h, 07eh, 0dbh, 0dbh, 0f3h, 07eh,
060h
 52561 0001373D 067EDBDBF37E60      <1>
 52562 00013744 C000000000001C3060- <1>     db  0c0h, 000h, 000h, 000h, 000h, 000h, 01ch, 030h, 060h, 060h, 07ch, 060h, 060h, 030h, 01ch,
000h
 52563 0001374D 607C6060301C00      <1>
 52564 00013754 00000000007CC6C6C6- <1>     db  000h, 000h, 000h, 000h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 000h, 000h,
000h
 52565 0001375D C6C6C6C6000000      <1>
 52566 00013764 000000FE0000FE0000- <1>     db  000h, 000h, 000h, 0feh, 000h, 000h, 0feh, 000h, 000h, 0feh, 000h, 000h, 000h, 000h, 000h,
000h
 52567 0001376D FE000000000000      <1>
 52568 00013774 0018187E18180000FF- <1>     db  000h, 018h, 018h, 07eh, 018h, 018h, 000h, 000h, 0ffh, 000h, 000h, 000h, 000h, 000h, 030h,
018h
 52569 0001377D 00000000003018      <1>
 52570 00013784 0C060C1830007E0000- <1>     db  00ch, 006h, 00ch, 018h, 030h, 000h, 07eh, 000h, 000h, 000h, 000h, 000h, 00ch, 018h, 030h,
060h
 52571 0001378D 0000000C183060      <1>
 52572 00013794 30180C007E00000000- <1>     db  030h, 018h, 00ch, 000h, 07eh, 000h, 000h, 000h, 000h, 000h, 00eh, 01bh, 01bh, 018h, 018h,
018h
 52573 0001379D 000E1B1B181818      <1>
 52574 000137A4 1818181818181818- <1>     db  018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 0d8h,
0d8h
 52575 000137AD 1818181818D8D8      <1>
 52576 000137B4 7000000000000001818- <1>     db  070h, 000h, 000h, 000h, 000h, 000h, 000h, 018h, 018h, 000h, 07eh, 000h, 018h, 018h, 000h,
000h
 52577 000137BD 007E0018180000      <1>
 52578 000137C4 00000000000076DC00- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 076h, 0dch, 000h, 076h, 0dch, 000h, 000h, 000h, 000h,
000h
 52579 000137CD 76DC0000000000      <1>
 52580 000137D4 00386C6C3800000000- <1>     db  000h, 038h, 06ch, 06ch, 038h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 52581 000137DD 00000000000000      <1>
 52582 000137E4 0000000001818000000- <1>     db  000h, 000h, 000h, 000h, 018h, 018h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 52583 000137ED 00000000000000      <1>
 52584 000137F4 0000001800000000000- <1>     db  000h, 000h, 000h, 018h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 00fh, 00ch, 00ch, 00ch,
00ch
 52585 000137FD 00000F0C0C0C0C      <1>
 52586 00013804 0CEC6C3C1C00000000- <1>     db  00ch, 0ech, 06ch, 03ch, 01ch, 000h, 000h, 000h, 000h, 0d8h, 06ch, 06ch, 06ch, 06ch, 06ch,
000h
 52587 0001380D D86C6C6C6C6C00      <1>
 52588 00013814 0000000000000070D8- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 070h, 0d8h, 030h, 060h, 0c8h, 0f8h, 000h, 000h,
000h
 52589 0001381D 3060C8F8000000      <1>
 52590 00013824 000000000000000007C- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 07ch, 07ch, 07ch, 07ch, 07ch, 07ch, 000h,
000h
```

```
 52591 0001382D 7C7C7C7C7C0000      <1>
 52592 00013834 000000000000000000- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 52593 0001383D 00000000000000      <1>
 52594                              <1> vgafont16:
 52595 00013844 000000000000000000- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 52596 0001384D 00000000000000      <1>
 52597 00013854 00007E81A58181BD99- <1>     db  000h, 000h, 07eh, 081h, 0a5h, 081h, 081h, 0bdh, 099h, 081h, 081h, 07eh, 000h, 000h, 000h,
000h
 52598 0001385D 81817E00000000      <1>
 52599 00013864 00007EFFDBFFFFC3E7- <1>     db  000h, 000h, 07eh, 0ffh, 0dbh, 0ffh, 0ffh, 0c3h, 0e7h, 0ffh, 0ffh, 07eh, 000h, 000h,
000h, 000h
 52600 0001386D FFFF7E00000000      <1>
 52601 00013874 000000006CFEFEFEFE- <1>     db  000h, 000h, 000h, 000h, 06ch, 0feh, 0feh, 0feh, 0feh, 07ch, 038h, 010h, 000h, 000h, 000h,
000h
 52602 0001387D 7C381000000000      <1>
 52603 00013884 0000000010387CFE7C- <1>     db  000h, 000h, 000h, 000h, 010h, 038h, 07ch, 0feh, 07ch, 038h, 010h, 000h, 000h, 000h, 000h,
000h
 52604 0001388D 38100000000000      <1>
 52605 00013894 000000183C3CE7E7E7- <1>     db  000h, 000h, 000h, 018h, 03ch, 03ch, 0e7h, 0e7h, 0e7h, 018h, 018h, 03ch, 000h, 000h, 000h,
000h
 52606 0001389D 18183C00000000      <1>
 52607 000138A4 000000183C7EFFFF7E- <1>     db  000h, 000h, 000h, 018h, 03ch, 07eh, 0ffh, 0ffh, 07eh, 018h, 018h, 03ch, 000h, 000h, 000h,
000h
 52608 000138AD 18183C00000000      <1>
 52609 000138B4 000000000000183C3C- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 018h, 03ch, 03ch, 018h, 000h, 000h, 000h, 000h, 000h,
000h
 52610 000138BD 18000000000000      <1>
 52611 000138C4 FFFFFFFFFFFFE7C3C3- <1>     db  0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0e7h, 0c3h, 0c3h, 0e7h, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh,
0ffh
 52612 000138CD E7FFFFFFFFFFFF      <1>
 52613 000138D4 00000000003C664242- <1>     db  000h, 000h, 000h, 000h, 000h, 03ch, 066h, 042h, 042h, 066h, 03ch, 000h, 000h, 000h, 000h,
000h
 52614 000138DD 663C0000000000      <1>
 52615 000138E4 FFFFFFFFFFC399BDBD- <1>     db  0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0c3h, 099h, 0bdh, 0bdh, 099h, 0c3h, 0ffh, 0ffh, 0ffh, 0ffh,
0ffh
 52616 000138ED 99C3FFFFFFFFFF      <1>
 52617 000138F4 00001E0E1A3278CCCC- <1>     db  000h, 000h, 01eh, 00eh, 01ah, 032h, 078h, 0cch, 0cch, 0cch, 0cch, 078h, 000h, 000h, 000h,
000h
 52618 000138FD CCCC7800000000      <1>
 52619 00013904 00003C666666663C18- <1>     db  000h, 000h, 03ch, 066h, 066h, 066h, 066h, 03ch, 018h, 07eh, 018h, 018h, 000h, 000h, 000h,
000h
 52620 0001390D 7E181800000000      <1>
 52621 00013914 00003F333F30303030- <1>     db  000h, 000h, 03fh, 033h, 03fh, 030h, 030h, 030h, 030h, 070h, 0f0h, 0e0h, 000h, 000h, 000h,
000h
 52622 0001391D 70F0E000000000      <1>
 52623 00013924 00007F637F63636363- <1>     db  000h, 000h, 07fh, 063h, 07fh, 063h, 063h, 063h, 063h, 067h, 0e7h, 0e6h, 0c0h, 000h, 000h,
000h
 52624 0001392D 67E7E6C0000000      <1>
 52625 00013934 0000001818DB3CE73C- <1>     db  000h, 000h, 000h, 018h, 018h, 0dbh, 03ch, 0e7h, 03ch, 0dbh, 018h, 018h, 000h, 000h, 000h,
000h
 52626 0001393D DB181800000000      <1>
 52627 00013944 0080C0E0F0F8FEF8F0- <1>     db  000h, 080h, 0c0h, 0e0h, 0f0h, 0f8h, 0feh, 0f8h, 0f0h, 0e0h, 0c0h, 080h, 000h, 000h, 000h,
000h
 52628 0001394D E0C08000000000      <1>
 52629 00013954 0002060E1E3EFE3E1E- <1>     db  000h, 002h, 006h, 00eh, 01eh, 03eh, 0feh, 03eh, 01eh, 00eh, 006h, 002h, 000h, 000h, 000h,
000h
 52630 0001395D 0E060200000000      <1>
 52631 00013964 0000183C7E1818187E- <1>     db  000h, 000h, 018h, 03ch, 07eh, 018h, 018h, 018h, 07eh, 03ch, 018h, 000h, 000h, 000h, 000h,
000h
 52632 0001396D 3C180000000000      <1>
 52633 00013974 000066666666666666- <1>     db  000h, 000h, 066h, 066h, 066h, 066h, 066h, 066h, 066h, 000h, 066h, 066h, 000h, 000h, 000h,
000h
 52634 0001397D 00666600000000      <1>
 52635 00013984 00007FDBDBDB7B1B1B- <1>     db  000h, 000h, 07fh, 0dbh, 0dbh, 0dbh, 07bh, 01bh, 01bh, 01bh, 01bh, 01bh, 000h, 000h, 000h,
000h
 52636 0001398D 1B1B1B00000000      <1>
 52637 00013994 007CC660386CC6C66C- <1>     db  000h, 07ch, 0c6h, 060h, 038h, 06ch, 0c6h, 0c6h, 06ch, 038h, 00ch, 0c6h, 07ch, 000h, 000h,
000h
 52638 0001399D 380CC67C000000      <1>
 52639 000139A4 0000000000000000FE- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0feh, 0feh, 0feh, 0feh, 000h, 000h, 000h,
000h
 52640 000139AD FEFEFE00000000      <1>
 52641 000139B4 0000183C7E1818187E- <1>     db  000h, 000h, 018h, 03ch, 07eh, 018h, 018h, 018h, 07eh, 03ch, 018h, 07eh, 000h, 000h, 000h,
000h
 52642 000139BD 3C187E00000000      <1>
 52643 000139C4 0000183C7E18181818- <1>     db  000h, 000h, 018h, 03ch, 07eh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 000h, 000h, 000h,
000h
 52644 000139CD 18181800000000      <1>
 52645 000139D4 000018181818181818- <1>     db  000h, 000h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 07eh, 03ch, 018h, 000h, 000h, 000h,
000h
 52646 000139DD 7E3C1800000000      <1>
 52647 000139E4 0000000000180CFE0C- <1>     db  000h, 000h, 000h, 000h, 000h, 018h, 00ch, 0feh, 00ch, 018h, 000h, 000h, 000h, 000h, 000h,
000h
 52648 000139ED 18000000000000      <1>
 52649 000139F4 00000000003060FE60- <1>     db  000h, 000h, 000h, 000h, 000h, 030h, 060h, 0feh, 060h, 030h, 000h, 000h, 000h, 000h, 000h,
000h
 52650 000139FD 30000000000000      <1>
 52651 00013A04 000000000000C0C0C0- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 0c0h, 0c0h, 0c0h, 0feh, 000h, 000h, 000h, 000h, 000h,
000h
 52652 00013A0D FE0000000000000     <1>
 52653 00013A14 00000000002466FF66- <1>     db  000h, 000h, 000h, 000h, 000h, 024h, 066h, 0ffh, 066h, 024h, 000h, 000h, 000h, 000h, 000h,
000h
 52654 00013A1D 24000000000000      <1>
 52655 00013A24 000000001038387C7C- <1>     db  000h, 000h, 000h, 000h, 010h, 038h, 038h, 07ch, 07ch, 0feh, 0feh, 000h, 000h, 000h, 000h,
000h
 52656 00013A2D FEFE0000000000      <1>
 52657 00013A34 00000000FEFE7C7C38- <1>     db  000h, 000h, 000h, 000h, 0feh, 0feh, 07ch, 07ch, 038h, 038h, 010h, 000h, 000h, 000h, 000h,
000h
 52658 00013A3D 38100000000000      <1>
```

```
 52659 00013A44 00000000000000000- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 52660 00013A4D 00000000000000      <1>
 52661 00013A54 0000183C3C3C181818- <1>     db  000h, 000h, 018h, 03ch, 03ch, 03ch, 018h, 018h, 018h, 000h, 018h, 018h, 000h, 000h, 000h,
000h
 52662 00013A5D 00181800000000      <1>
 52663 00013A64 006666662400000000- <1>     db  000h, 066h, 066h, 066h, 024h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 52664 00013A6D 00000000000000      <1>
 52665 00013A74 0000006C6CFE6C6C6C- <1>     db  000h, 000h, 000h, 06ch, 06ch, 0feh, 06ch, 06ch, 06ch, 0feh, 06ch, 06ch, 000h, 000h, 000h,
000h
 52666 00013A7D FE6C6C00000000      <1>
 52667 00013A84 18187CC6C2C07C0606- <1>     db  018h, 018h, 07ch, 0c6h, 0c2h, 0c0h, 07ch, 006h, 006h, 086h, 0c6h, 07ch, 018h, 018h,
000h, 000h
 52668 00013A8D 86C67C18180000      <1>
 52669 00013A94 00000000C2C60C1830- <1>     db  000h, 000h, 000h, 000h, 0c2h, 0c6h, 00ch, 018h, 030h, 060h, 0c6h, 086h, 000h, 000h, 000h,
000h
 52670 00013A9D 60C68600000000      <1>
 52671 00013AA4 0000386C6C3876DCCC- <1>     db  000h, 000h, 038h, 06ch, 06ch, 038h, 076h, 0dch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h,
000h
 52672 00013AAD CCCC7600000000      <1>
 52673 00013AB4 0030303060000000000- <1>    db  000h, 030h, 030h, 030h, 060h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 52674 00013ABD 00000000000000      <1>
 52675 00013AC4 00000C183030303030- <1>     db  000h, 000h, 00ch, 018h, 030h, 030h, 030h, 030h, 030h, 030h, 018h, 00ch, 000h, 000h, 000h,
000h
 52676 00013ACD 30180C00000000      <1>
 52677 00013AD4 000030180C0C0C0C0C- <1>     db  000h, 000h, 030h, 018h, 00ch, 00ch, 00ch, 00ch, 00ch, 00ch, 018h, 030h, 000h, 000h, 000h,
000h
 52678 00013ADD 0C183000000000      <1>
 52679 00013AE4 0000000000663CFF3C- <1>     db  000h, 000h, 000h, 000h, 000h, 066h, 03ch, 0ffh, 03ch, 066h, 000h, 000h, 000h, 000h, 000h,
000h
 52680 00013AED 66000000000000      <1>
 52681 00013AF4 000000000018187E18- <1>     db  000h, 000h, 000h, 000h, 000h, 018h, 018h, 07eh, 018h, 018h, 000h, 000h, 000h, 000h, 000h,
000h
 52682 00013AFD 18000000000000      <1>
 52683 00013B04 0000000000000000000- <1>    db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 018h, 018h, 018h, 030h, 000h, 000h,
000h
 52684 00013B0D 18181830000000      <1>
 52685 00013B14 00000000000000FE00- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 0feh, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 52686 00013B1D 00000000000000      <1>
 52687 00013B24 0000000000000000000- <1>    db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 018h, 018h, 000h, 000h, 000h,
000h
 52688 00013B2D 00181800000000      <1>
 52689 00013B34 0000000002060C1830- <1>     db  000h, 000h, 000h, 000h, 002h, 006h, 00ch, 018h, 030h, 060h, 0c0h, 080h, 000h, 000h, 000h,
000h
 52690 00013B3D 60C08000000000      <1>
 52691 00013B44 00003C66C3C3DBDBC3- <1>     db  000h, 000h, 03ch, 066h, 0c3h, 0c3h, 0dbh, 0dbh, 0c3h, 0c3h, 066h, 03ch, 000h, 000h, 000h,
000h
 52692 00013B4D C3663C00000000      <1>
 52693 00013B54 000018387818181818- <1>     db  000h, 000h, 018h, 038h, 078h, 018h, 018h, 018h, 018h, 018h, 018h, 07eh, 000h, 000h, 000h,
000h
 52694 00013B5D 18187E00000000      <1>
 52695 00013B64 00007CC6060C183060- <1>     db  000h, 000h, 07ch, 0c6h, 006h, 00ch, 018h, 030h, 060h, 0c0h, 0c6h, 0feh, 000h, 000h, 000h,
000h
 52696 00013B6D C0C6FE00000000      <1>
 52697 00013B74 00007CC606063C0606- <1>     db  000h, 000h, 07ch, 0c6h, 006h, 006h, 03ch, 006h, 006h, 006h, 0c6h, 07ch, 000h, 000h, 000h,
000h
 52698 00013B7D 06C67C00000000      <1>
 52699 00013B84 00000C1C3C6CCCFE0C- <1>     db  000h, 000h, 00ch, 01ch, 03ch, 06ch, 0cch, 0feh, 00ch, 00ch, 00ch, 01eh, 000h, 000h, 000h,
000h
 52700 00013B8D 0C0C1E00000000      <1>
 52701 00013B94 0000FEC0C0C0FC0606- <1>     db  000h, 000h, 0feh, 0c0h, 0c0h, 0c0h, 0fch, 006h, 006h, 006h, 0c6h, 07ch, 000h, 000h, 000h,
000h
 52702 00013B9D 06C67C00000000      <1>
 52703 00013BA4 00003860C0C0FCC6C6- <1>     db  000h, 000h, 038h, 060h, 0c0h, 0c0h, 0fch, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h,
000h
 52704 00013BAD C6C67C00000000      <1>
 52705 00013BB4 0000FEC606060C1830- <1>     db  000h, 000h, 0feh, 0c6h, 006h, 006h, 00ch, 018h, 030h, 030h, 030h, 030h, 000h, 000h, 000h,
000h
 52706 00013BBD 30303000000000      <1>
 52707 00013BC4 00007CC6C6C67CC6C6- <1>     db  000h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h,
000h
 52708 00013BCD C6C67C00000000      <1>
 52709 00013BD4 00007CC6C6C67E0606- <1>     db  000h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 07eh, 006h, 006h, 006h, 00ch, 078h, 000h, 000h, 000h,
000h
 52710 00013BDD 060C7800000000      <1>
 52711 00013BE4 000000001818000000- <1>     db  000h, 000h, 000h, 000h, 018h, 018h, 000h, 000h, 000h, 018h, 018h, 000h, 000h, 000h, 000h,
000h
 52712 00013BED 18180000000000      <1>
 52713 00013BF4 000000001818000000- <1>     db  000h, 000h, 000h, 000h, 018h, 018h, 000h, 000h, 000h, 018h, 018h, 030h, 000h, 000h, 000h,
000h
 52714 00013BFD 18183000000000      <1>
 52715 00013C04 000000060C18306030- <1>     db  000h, 000h, 000h, 006h, 00ch, 018h, 030h, 060h, 030h, 018h, 00ch, 006h, 000h, 000h, 000h,
000h
 52716 00013C0D 180C0600000000      <1>
 52717 00013C14 00000000007E00007E- <1>     db  000h, 000h, 000h, 000h, 000h, 07eh, 000h, 000h, 07eh, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 52718 00013C1D 00000000000000      <1>
 52719 00013C24 0000006030180C060C- <1>     db  000h, 000h, 000h, 060h, 030h, 018h, 00ch, 006h, 00ch, 018h, 030h, 060h, 000h, 000h, 000h,
000h
 52720 00013C2D 18306000000000      <1>
 52721 00013C34 00007CC6C60C181818- <1>     db  000h, 000h, 07ch, 0c6h, 0c6h, 00ch, 018h, 018h, 018h, 000h, 018h, 018h, 000h, 000h, 000h,
000h
 52722 00013C3D 00181800000000      <1>
 52723 00013C44 0000007CC6C6DEDEDE- <1>     db  000h, 000h, 000h, 07ch, 0c6h, 0c6h, 0deh, 0deh, 0deh, 0dch, 0c0h, 07ch, 000h, 000h, 000h,
000h
 52724 00013C4D DCC07C00000000      <1>
 52725 00013C54 000010386CC6C6FEC6- <1>     db  000h, 000h, 010h, 038h, 06ch, 0c6h, 0c6h, 0feh, 0c6h, 0c6h, 0c6h, 0c6h, 000h, 000h, 000h,
000h
 52726 00013C5D C6C6C600000000      <1>
```

```
 52727 00013C64 0000FC6666667C6666- <1>     db  000h, 000h, 0fch, 066h, 066h, 066h, 07ch, 066h, 066h, 066h, 066h, 0fch, 000h, 000h, 000h,
000h
 52728 00013C6D 6666FC00000000      <1>
 52729 00013C74 00003C66C2C0C0C0C0- <1>     db  000h, 000h, 03ch, 066h, 0c2h, 0c0h, 0c0h, 0c0h, 0c0h, 0c2h, 066h, 03ch, 000h, 000h, 000h,
000h
 52730 00013C7D C2663C00000000      <1>
 52731 00013C84 0000F86C6666666666- <1>     db  000h, 000h, 0f8h, 06ch, 066h, 066h, 066h, 066h, 066h, 066h, 06ch, 0f8h, 000h, 000h, 000h,
000h
 52732 00013C8D 666CF800000000      <1>
 52733 00013C94 0000FE666268786860- <1>     db  000h, 000h, 0feh, 066h, 062h, 068h, 078h, 068h, 060h, 062h, 066h, 0feh, 000h, 000h, 000h,
000h
 52734 00013C9D 6266FE00000000      <1>
 52735 00013CA4 0000FE666268786860- <1>     db  000h, 000h, 0feh, 066h, 062h, 068h, 078h, 068h, 060h, 060h, 060h, 0f0h, 000h, 000h,
000h, 000h
 52736 00013CAD 6060F000000000      <1>
 52737 00013CB4 00003C66C2C0C0DEC6- <1>     db  000h, 000h, 03ch, 066h, 0c2h, 0c0h, 0c0h, 0deh, 0c6h, 0c6h, 066h, 03ah, 000h, 000h, 000h,
000h
 52738 00013CBD C6663A00000000      <1>
 52739 00013CC4 0000C6C6C6C6FEC6C6- <1>     db  000h, 000h, 0c6h, 0c6h, 0c6h, 0c6h, 0feh, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 000h, 000h, 000h,
000h
 52740 00013CCD C6C6C600000000      <1>
 52741 00013CD4 00003C181818181818- <1>     db  000h, 000h, 03ch, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h,
000h
 52742 00013CDD 18183C00000000      <1>
 52743 00013CE4 00001E0C0C0C0C0CCC- <1>     db  000h, 000h, 01eh, 00ch, 00ch, 00ch, 00ch, 00ch, 0cch, 0cch, 0cch, 078h, 000h, 000h, 000h,
000h
 52744 00013CED CCCC7800000000      <1>
 52745 00013CF4 0000E666666C78786C- <1>     db  000h, 000h, 0e6h, 066h, 066h, 06ch, 078h, 078h, 06ch, 066h, 066h, 0e6h, 000h, 000h, 000h,
000h
 52746 00013CFD 6666E600000000      <1>
 52747 00013D04 0000F0606060606060- <1>     db  000h, 000h, 0f0h, 060h, 060h, 060h, 060h, 060h, 060h, 062h, 066h, 0feh, 000h, 000h, 000h,
000h
 52748 00013D0D 6266FE00000000      <1>
 52749 00013D14 0000C3E7FFFFDBC3C3- <1>     db  000h, 000h, 0c3h, 0e7h, 0ffh, 0ffh, 0dbh, 0c3h, 0c3h, 0c3h, 0c3h, 0c3h, 000h, 000h, 000h,
000h
 52750 00013D1D C3C3C300000000      <1>
 52751 00013D24 0000C6E6F6FEDECEC6- <1>     db  000h, 000h, 0c6h, 0e6h, 0f6h, 0feh, 0deh, 0ceh, 0c6h, 0c6h, 0c6h, 0c6h, 000h, 000h, 000h,
000h
 52752 00013D2D C6C6C600000000      <1>
 52753 00013D34 00007CC6C6C6C6C6C6- <1>     db  000h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h,
000h
 52754 00013D3D C6C67C00000000      <1>
 52755 00013D44 0000FC6666667C6060- <1>     db  000h, 000h, 0fch, 066h, 066h, 066h, 07ch, 060h, 060h, 060h, 060h, 0f0h, 000h, 000h, 000h,
000h
 52756 00013D4D 6060F000000000      <1>
 52757 00013D54 00007CC6C6C6C6C6C6- <1>     db  000h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0d6h, 0deh, 07ch, 00ch, 00eh, 000h,
000h
 52758 00013D5D D6DE7C0C0E0000      <1>
 52759 00013D64 0000FC6666667C6C66- <1>     db  000h, 000h, 0fch, 066h, 066h, 066h, 07ch, 06ch, 066h, 066h, 066h, 0e6h, 000h, 000h, 000h,
000h
 52760 00013D6D 6666E600000000      <1>
 52761 00013D74 00007CC6C660380C06- <1>     db  000h, 000h, 07ch, 0c6h, 0c6h, 060h, 038h, 00ch, 006h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h,
000h
 52762 00013D7D C6C67C00000000      <1>
 52763 00013D84 0000FFDB9918181818- <1>     db  000h, 000h, 0ffh, 0dbh, 099h, 018h, 018h, 018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h,
000h
 52764 00013D8D 18183C00000000      <1>
 52765 00013D94 0000C6C6C6C6C6C6C6- <1>     db  000h, 000h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h,
000h
 52766 00013D9D C6C67C00000000      <1>
 52767 00013DA4 0000C3C3C3C3C3C3C3- <1>     db  000h, 000h, 0c3h, 0c3h, 0c3h, 0c3h, 0c3h, 0c3h, 0c3h, 066h, 03ch, 018h, 000h, 000h, 000h,
000h
 52768 00013DAD 663C1800000000      <1>
 52769 00013DB4 0000C3C3C3C3C3DBDB- <1>     db  000h, 000h, 0c3h, 0c3h, 0c3h, 0c3h, 0c3h, 0dbh, 0dbh, 0ffh, 066h, 066h, 000h, 000h, 000h,
000h
 52770 00013DBD FF666600000000      <1>
 52771 00013DC4 0000C3C3663C18183C- <1>     db  000h, 000h, 0c3h, 0c3h, 066h, 03ch, 018h, 018h, 03ch, 066h, 0c3h, 0c3h, 000h, 000h, 000h,
000h
 52772 00013DCD 66C3C300000000      <1>
 52773 00013DD4 0000C3C3C3663C1818- <1>     db  000h, 000h, 0c3h, 0c3h, 0c3h, 066h, 03ch, 018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h,
000h
 52774 00013DDD 18183C00000000      <1>
 52775 00013DE4 0000FFC3860C183060- <1>     db  000h, 000h, 0ffh, 0c3h, 086h, 00ch, 018h, 030h, 060h, 0c1h, 0c3h, 0ffh, 000h, 000h, 000h,
000h
 52776 00013DED C1C3FF00000000      <1>
 52777 00013DF4 00003C303030303030- <1>     db  000h, 000h, 03ch, 030h, 030h, 030h, 030h, 030h, 030h, 030h, 030h, 03ch, 000h, 000h, 000h,
000h
 52778 00013DFD 30303C00000000      <1>
 52779 00013E04 00000080C0E070381C- <1>     db  000h, 000h, 000h, 080h, 0c0h, 0e0h, 070h, 038h, 01ch, 00eh, 006h, 002h, 000h, 000h, 000h,
000h
 52780 00013E0D 0E060200000000      <1>
 52781 00013E14 00003C0C0C0C0C0C0C- <1>     db  000h, 000h, 03ch, 00ch, 00ch, 00ch, 00ch, 00ch, 00ch, 00ch, 00ch, 03ch, 000h, 000h, 000h,
000h
 52782 00013E1D 0C0C3C00000000      <1>
 52783 00013E24 10386CC60000000000- <1>     db  010h, 038h, 06ch, 0c6h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 52784 00013E2D 00000000000000      <1>
 52785 00013E34 0000000000000000- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0ffh, 000h,
000h
 52786 00013E3D 00000000FF0000      <1>
 52787 00013E44 3030180000000000- <1>     db  030h, 030h, 018h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 52788 00013E4D 00000000000000      <1>
 52789 00013E54 0000000000780C7CCC- <1>     db  000h, 000h, 000h, 000h, 000h, 078h, 00ch, 07ch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h,
000h
 52790 00013E5D CCCC7600000000      <1>
 52791 00013E64 0000E06060786C6666- <1>     db  000h, 000h, 0e0h, 060h, 060h, 078h, 06ch, 066h, 066h, 066h, 066h, 07ch, 000h, 000h, 000h,
000h
 52792 00013E6D 66667C00000000      <1>
 52793 00013E74 00000000007CC6C0C0- <1>     db  000h, 000h, 000h, 000h, 000h, 07ch, 0c6h, 0c0h, 0c0h, 0c0h, 0c6h, 07ch, 000h, 000h, 000h,
000h
 52794 00013E7D C0C67C00000000      <1>
```

```
 52795 00013E84 00001C0C0C3C6CCCCC- <1>     db  000h, 000h, 01ch, 00ch, 00ch, 03ch, 06ch, 0cch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h,
000h
 52796 00013E8D CCCC7600000000      <1>
 52797 00013E94 00000000007CC6FEC0- <1>     db  000h, 000h, 000h, 000h, 000h, 07ch, 0c6h, 0feh, 0c0h, 0c0h, 0c6h, 07ch, 000h, 000h, 000h,
000h
 52798 00013E9D C0C67C00000000      <1>
 52799 00013EA4 0000386C6460F06060- <1>     db  000h, 000h, 038h, 06ch, 064h, 060h, 0f0h, 060h, 060h, 060h, 060h, 0f0h, 000h, 000h, 000h,
000h
 52800 00013EAD 6060F000000000      <1>
 52801 00013EB4 000000000076CCCCCC- <1>     db  000h, 000h, 000h, 000h, 000h, 076h, 0cch, 0cch, 0cch, 0cch, 0cch, 07ch, 00ch, 0cch, 078h,
000h
 52802 00013EBD CCCC7C0CCC7800      <1>
 52803 00013EC4 0000E060606C766666- <1>     db  000h, 000h, 0e0h, 060h, 060h, 06ch, 076h, 066h, 066h, 066h, 066h, 0e6h, 000h, 000h,
000h, 000h
 52804 00013ECD 6666E600000000      <1>
 52805 00013ED4 000018180038181818- <1>     db  000h, 000h, 018h, 018h, 000h, 038h, 018h, 018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h,
000h
 52806 00013EDD 18183C00000000      <1>
 52807 00013EE4 00000606000E060606- <1>     db  000h, 000h, 006h, 006h, 000h, 00eh, 006h, 006h, 006h, 006h, 006h, 006h, 066h, 066h, 03ch,
000h
 52808 00013EED 06060666663C00      <1>
 52809 00013EF4 0000E06060666C7878- <1>     db  000h, 000h, 0e0h, 060h, 060h, 066h, 06ch, 078h, 078h, 06ch, 066h, 0e6h, 000h, 000h, 000h,
000h
 52810 00013EFD 6C66E600000000      <1>
 52811 00013F04 000038181818181818- <1>     db  000h, 000h, 038h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h,
000h
 52812 00013F0D 18183C00000000      <1>
 52813 00013F14 0000000000E6FFDBDB- <1>     db  000h, 000h, 000h, 000h, 000h, 0e6h, 0ffh, 0dbh, 0dbh, 0dbh, 0dbh, 0dbh, 000h, 000h, 000h,
000h
 52814 00013F1D DBDBDB00000000      <1>
 52815 00013F24 0000000000DC666666- <1>     db  000h, 000h, 000h, 000h, 000h, 0dch, 066h, 066h, 066h, 066h, 066h, 066h, 000h, 000h, 000h,
000h
 52816 00013F2D 66666600000000      <1>
 52817 00013F34 00000000007CC6C6C6- <1>     db  000h, 000h, 000h, 000h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h,
000h
 52818 00013F3D C6C67C00000000      <1>
 52819 00013F44 0000000000DC666666- <1>     db  000h, 000h, 000h, 000h, 000h, 0dch, 066h, 066h, 066h, 066h, 066h, 07ch, 060h, 060h, 0f0h,
000h
 52820 00013F4D 66667C6060F000      <1>
 52821 00013F54 000000000076CCCCCC- <1>     db  000h, 000h, 000h, 000h, 000h, 076h, 0cch, 0cch, 0cch, 0cch, 0cch, 07ch, 00ch, 00ch, 01eh,
000h
 52822 00013F5D CCCC7C0C0C1E00      <1>
 52823 00013F64 0000000000DC766660- <1>     db  000h, 000h, 000h, 000h, 000h, 0dch, 076h, 066h, 060h, 060h, 060h, 0f0h, 000h, 000h, 000h,
000h
 52824 00013F6D 6060F000000000      <1>
 52825 00013F74 00000000007CC66038- <1>     db  000h, 000h, 000h, 000h, 000h, 07ch, 0c6h, 060h, 038h, 00ch, 0c6h, 07ch, 000h, 000h, 000h,
000h
 52826 00013F7D 0CC67C00000000      <1>
 52827 00013F84 0000103030FC303030- <1>     db  000h, 000h, 010h, 030h, 030h, 0fch, 030h, 030h, 030h, 030h, 036h, 01ch, 000h, 000h, 000h,
000h
 52828 00013F8D 30361C00000000      <1>
 52829 00013F94 0000000000CCCCCCCC- <1>     db  000h, 000h, 000h, 000h, 000h, 0cch, 0cch, 0cch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h,
000h
 52830 00013F9D CCCC7600000000      <1>
 52831 00013FA4 0000000000C3C3C3C3- <1>     db  000h, 000h, 000h, 000h, 000h, 0c3h, 0c3h, 0c3h, 0c3h, 066h, 03ch, 018h, 000h, 000h, 000h,
000h
 52832 00013FAD 663C1800000000      <1>
 52833 00013FB4 0000000000C3C3C3DB- <1>     db  000h, 000h, 000h, 000h, 000h, 0c3h, 0c3h, 0c3h, 0dbh, 0dbh, 0ffh, 066h, 000h, 000h, 000h,
000h
 52834 00013FBD DBFF6600000000      <1>
 52835 00013FC4 0000000000C3663C18- <1>     db  000h, 000h, 000h, 000h, 000h, 0c3h, 066h, 03ch, 018h, 03ch, 066h, 0c3h, 000h, 000h, 000h,
000h
 52836 00013FCD 3C66C300000000      <1>
 52837 00013FD4 0000000000C6C6C6C6- <1>     db  000h, 000h, 000h, 000h, 000h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 07eh, 006h, 00ch, 0f8h,
000h
 52838 00013FDD C6C67E060CF800      <1>
 52839 00013FE4 0000000000FECC1830- <1>     db  000h, 000h, 000h, 000h, 000h, 0feh, 0cch, 018h, 030h, 060h, 0c6h, 0feh, 000h, 000h, 000h,
000h
 52840 00013FED 60C6FE00000000      <1>
 52841 00013FF4 00000E181818701818- <1>     db  000h, 000h, 00eh, 018h, 018h, 018h, 070h, 018h, 018h, 018h, 018h, 00eh, 000h, 000h, 000h,
000h
 52842 00013FFD 18180E00000000      <1>
 52843 00014004 000018181818001818- <1>     db  000h, 000h, 018h, 018h, 018h, 018h, 000h, 018h, 018h, 018h, 018h, 018h, 000h, 000h, 000h,
000h
 52844 0001400D 18181800000000      <1>
 52845 00014014 0000701818180E1818- <1>     db  000h, 000h, 070h, 018h, 018h, 018h, 00eh, 018h, 018h, 018h, 018h, 070h, 000h, 000h, 000h,
000h
 52846 0001401D 18187000000000      <1>
 52847 00014024 000076DC0000000000- <1>     db  000h, 000h, 076h, 0dch, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 52848 0001402D 00000000000000      <1>
 52849 00014034 0000000010386CC6C6- <1>     db  000h, 000h, 000h, 000h, 010h, 038h, 06ch, 0c6h, 0c6h, 0c6h, 0feh, 000h, 000h, 000h, 000h,
000h
 52850 0001403D C6FE0000000000      <1>
 52851 00014044 00003C66C2C0C0C0C2- <1>     db  000h, 000h, 03ch, 066h, 0c2h, 0c0h, 0c0h, 0c0h, 0c2h, 066h, 03ch, 00ch, 006h, 07ch, 000h,
000h
 52852 0001404D 663C0C067C0000      <1>
 52853 00014054 0000CC0000CCCCCCCC- <1>     db  000h, 000h, 0cch, 000h, 000h, 0cch, 0cch, 0cch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h,
000h
 52854 0001405D CCCC7600000000      <1>
 52855 00014064 000C1830007CC6FEC0- <1>     db  000h, 00ch, 018h, 030h, 000h, 07ch, 0c6h, 0feh, 0c0h, 0c0h, 0c6h, 07ch, 000h, 000h, 000h,
000h
 52856 0001406D C0C67C00000000      <1>
 52857 00014074 0010386C00780C7CCC- <1>     db  000h, 010h, 038h, 06ch, 000h, 078h, 00ch, 07ch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h,
000h
 52858 0001407D CCCC7600000000      <1>
 52859 00014084 0000CC0000780C7CCC- <1>     db  000h, 000h, 0cch, 000h, 000h, 078h, 00ch, 07ch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h,
000h
 52860 0001408D CCCC7600000000      <1>
 52861 00014094 0060301800780C7CCC- <1>     db  000h, 060h, 030h, 018h, 000h, 078h, 00ch, 07ch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h,
000h
 52862 0001409D CCCC7600000000      <1>
```

```
 52863 000140A4 00386C3800780C7CCC- <1>     db  000h, 038h, 06ch, 038h, 000h, 078h, 00ch, 07ch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h,
000h
 52864 000140AD CCCC7600000000      <1>
 52865 000140B4 000000003C66606066- <1>     db  000h, 000h, 000h, 000h, 03ch, 066h, 060h, 060h, 066h, 03ch, 00ch, 006h, 03ch, 000h, 000h,
000h
 52866 000140BD 3C0C063C000000      <1>
 52867 000140C4 0010386C007CC6FEC0- <1>     db  000h, 010h, 038h, 06ch, 000h, 07ch, 0c6h, 0feh, 0c0h, 0c0h, 0c6h, 07ch, 000h, 000h, 000h,
000h
 52868 000140CD C0C67C00000000      <1>
 52869 000140D4 0000C600007CC6FEC0- <1>     db  000h, 000h, 0c6h, 000h, 000h, 07ch, 0c6h, 0feh, 0c0h, 0c0h, 0c6h, 07ch, 000h, 000h, 000h,
000h
 52870 000140DD C0C67C00000000      <1>
 52871 000140E4 00603018007CC6FEC0- <1>     db  000h, 060h, 030h, 018h, 000h, 07ch, 0c6h, 0feh, 0c0h, 0c0h, 0c6h, 07ch, 000h, 000h,
000h, 000h
 52872 000140ED C0C67C00000000      <1>
 52873 000140F4 000066000038181818- <1>     db  000h, 000h, 066h, 000h, 000h, 038h, 018h, 018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h,
000h
 52874 000140FD 18183C00000000      <1>
 52875 00014104 00183C660038181818- <1>     db  000h, 018h, 03ch, 066h, 000h, 038h, 018h, 018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h,
000h
 52876 0001410D 18183C00000000      <1>
 52877 00014114 006030180038181818- <1>     db  000h, 060h, 030h, 018h, 000h, 038h, 018h, 018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h,
000h
 52878 0001411D 18183C00000000      <1>
 52879 00014124 00C60010386CC6C6FE- <1>     db  000h, 0c6h, 000h, 010h, 038h, 06ch, 0c6h, 0c6h, 0feh, 0c6h, 0c6h, 0c6h, 000h, 000h, 000h,
000h
 52880 0001412D C6C6C600000000      <1>
 52881 00014134 386C3800386CC6C6FE- <1>     db  038h, 06ch, 038h, 000h, 038h, 06ch, 0c6h, 0c6h, 0feh, 0c6h, 0c6h, 0c6h, 000h, 000h, 000h,
000h
 52882 0001413D C6C6C600000000      <1>
 52883 00014144 18306000FE66607C60- <1>     db  018h, 030h, 060h, 000h, 0feh, 066h, 060h, 07ch, 060h, 060h, 066h, 0feh, 000h, 000h, 000h,
000h
 52884 0001414D 6066FE00000000      <1>
 52885 00014154 00000000006E3B1B7E- <1>     db  000h, 000h, 000h, 000h, 000h, 06eh, 03bh, 01bh, 07eh, 0d8h, 0dch, 077h, 000h, 000h, 000h,
000h
 52886 0001415D D8DC7700000000      <1>
 52887 00014164 00003E6CCCCCFECCCC- <1>     db  000h, 000h, 03eh, 06ch, 0cch, 0cch, 0feh, 0cch, 0cch, 0cch, 0cch, 0ceh, 000h, 000h, 000h,
000h
 52888 0001416D CCCCCE00000000      <1>
 52889 00014174 0010386C007CC6C6C6- <1>     db  000h, 010h, 038h, 06ch, 000h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h,
000h
 52890 0001417D C6C67C00000000      <1>
 52891 00014184 0000C600007CC6C6C6- <1>     db  000h, 000h, 0c6h, 000h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h,
000h
 52892 0001418D C6C67C00000000      <1>
 52893 00014194 00603018007CC6C6C6- <1>     db  000h, 060h, 030h, 018h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h,
000h
 52894 0001419D C6C67C00000000      <1>
 52895 000141A4 003078CC00CCCCCCCC- <1>     db  000h, 030h, 078h, 0cch, 000h, 0cch, 0cch, 0cch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h,
000h
 52896 000141AD CCCC7600000000      <1>
 52897 000141B4 0060301800CCCCCCCC- <1>     db  000h, 060h, 030h, 018h, 000h, 0cch, 0cch, 0cch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h,
000h
 52898 000141BD CCCC7600000000      <1>
 52899 000141C4 0000C60000C6C6C6C6- <1>     db  000h, 000h, 0c6h, 000h, 000h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 07eh, 006h, 00ch, 078h,
000h
 52900 000141CD C6C67E060C7800      <1>
 52901 000141D4 00C6007CC6C6C6C6C6- <1>     db  000h, 0c6h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h,
000h
 52902 000141DD C6C67C00000000      <1>
 52903 000141E4 00C600C6C6C6C6C6C6- <1>     db  000h, 0c6h, 000h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h,
000h
 52904 000141ED C6C67C00000000      <1>
 52905 000141F4 0018187EC3C0C0C0C3- <1>     db  000h, 018h, 018h, 07eh, 0c3h, 0c0h, 0c0h, 0c0h, 0c3h, 07eh, 018h, 018h, 000h, 000h, 000h,
000h
 52906 000141FD 7E181800000000      <1>
 52907 00014204 00386C6460F0606060- <1>     db  000h, 038h, 06ch, 064h, 060h, 0f0h, 060h, 060h, 060h, 060h, 0e6h, 0fch, 000h, 000h, 000h,
000h
 52908 0001420D 60E6FC00000000      <1>
 52909 00014214 0000C3663C18FF18FF- <1>     db  000h, 000h, 0c3h, 066h, 03ch, 018h, 0ffh, 018h, 0ffh, 018h, 018h, 018h, 000h, 000h, 000h,
000h
 52910 0001421D 18181800000000      <1>
 52911 00014224 00FC66667C62666F66- <1>     db  000h, 0fch, 066h, 066h, 07ch, 062h, 066h, 06fh, 066h, 066h, 066h, 0f3h, 000h, 000h, 000h,
000h
 52912 0001422D 6666F300000000      <1>
 52913 00014234 000E1B1818187E1818- <1>     db  000h, 00eh, 01bh, 018h, 018h, 018h, 07eh, 018h, 018h, 018h, 018h, 018h, 0d8h, 070h, 000h,
000h
 52914 0001423D 181818D8700000      <1>
 52915 00014244 0018306000780C7CCC- <1>     db  000h, 018h, 030h, 060h, 000h, 078h, 00ch, 07ch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h,
000h
 52916 0001424D CCCC7600000000      <1>
 52917 00014254 000C18300038181818- <1>     db  000h, 00ch, 018h, 030h, 000h, 038h, 018h, 018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h,
000h
 52918 0001425D 18183C00000000      <1>
 52919 00014264 00183060007CC6C6C6- <1>     db  000h, 018h, 030h, 060h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h,
000h
 52920 0001426D C6C67C00000000      <1>
 52921 00014274 0018306000CCCCCCCC- <1>     db  000h, 018h, 030h, 060h, 000h, 0cch, 0cch, 0cch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h,
000h
 52922 0001427D CCCC7600000000      <1>
 52923 00014284 000076DC00DC666666- <1>     db  000h, 000h, 076h, 0dch, 000h, 0dch, 066h, 066h, 066h, 066h, 066h, 066h, 000h, 000h, 000h,
000h
 52924 0001428D 66666600000000      <1>
 52925 00014294 76DC00C6E6F6FEDECE- <1>     db  076h, 0dch, 000h, 0c6h, 0e6h, 0f6h, 0feh, 0deh, 0ceh, 0c6h, 0c6h, 0c6h, 000h, 000h, 000h,
000h
 52926 0001429D C6C6C600000000      <1>
 52927 000142A4 003C6C6C3E007E0000- <1>     db  000h, 03ch, 06ch, 06ch, 03eh, 000h, 07eh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 52928 000142AD 00000000000000      <1>
 52929 000142B4 00386C6C38007C0000- <1>     db  000h, 038h, 06ch, 06ch, 038h, 000h, 07ch, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 52930 000142BD 00000000000000      <1>
```

```
 52931 000142C4 0000303000303060C0- <1>      db  000h, 000h, 030h, 030h, 000h, 030h, 030h, 060h, 0c0h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h,
000h
 52932 000142CD C6C67C00000000      <1>
 52933 000142D4 000000000000FEC0C0- <1>      db  000h, 000h, 000h, 000h, 000h, 000h, 0feh, 0c0h, 0c0h, 0c0h, 0c0h, 000h, 000h, 000h, 000h,
000h
 52934 000142DD C0C00000000000      <1>
 52935 000142E4 000000000000FE0606- <1>      db  000h, 000h, 000h, 000h, 000h, 000h, 0feh, 006h, 006h, 006h, 006h, 000h, 000h, 000h, 000h,
000h
 52936 000142ED 06060000000000      <1>
 52937 000142F4 00C0C0C2C6CC183060- <1>      db  000h, 0c0h, 0c0h, 0c2h, 0c6h, 0cch, 018h, 030h, 060h, 0ceh, 09bh, 006h, 00ch, 01fh, 000h,
000h
 52938 000142FD CE9B060C1F0000      <1>
 52939 00014304 00C0C0C2C6CC183066- <1>      db  000h, 0c0h, 0c0h, 0c2h, 0c6h, 0cch, 018h, 030h, 066h, 0ceh, 096h, 03eh, 006h, 006h,
000h, 000h
 52940 0001430D CE963E06060000      <1>
 52941 00014314 00001818001818183C- <1>      db  000h, 000h, 018h, 018h, 000h, 018h, 018h, 018h, 03ch, 03ch, 03ch, 018h, 000h, 000h, 000h,
000h
 52942 0001431D 3C3C1800000000      <1>
 52943 00014324 0000000000366CD86C- <1>      db  000h, 000h, 000h, 000h, 000h, 036h, 06ch, 0d8h, 06ch, 036h, 000h, 000h, 000h, 000h, 000h,
000h
 52944 0001432D 36000000000000      <1>
 52945 00014334 0000000000D86C366C- <1>      db  000h, 000h, 000h, 000h, 000h, 0d8h, 06ch, 036h, 06ch, 0d8h, 000h, 000h, 000h, 000h, 000h,
000h
 52946 0001433D D8000000000000      <1>
 52947 00014344 114411441144114411- <1>      db  011h, 044h, 011h, 044h, 011h, 044h, 011h, 044h, 011h, 044h, 011h, 044h, 011h, 044h, 011h,
044h
 52948 0001434D 44114411441144      <1>
 52949 00014354 55AA55AA55AA55AA55- <1>      db  055h, 0aah, 055h, 0aah, 055h, 0aah, 055h, 0aah, 055h, 0aah, 055h, 0aah, 055h, 0aah, 055h,
0aah
 52950 0001435D AA55AA55AA55AA      <1>
 52951 00014364 DD77DD77DD77DD77DD- <1>      db  0ddh, 077h, 0ddh, 077h, 0ddh, 077h, 0ddh, 077h, 0ddh, 077h, 0ddh, 077h, 0ddh, 077h, 0ddh,
077h
 52952 0001436D 77DD77DD77DD77      <1>
 52953 00014374 181818181818181818- <1>      db  018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h,
018h
 52954 0001437D 18181818181818      <1>
 52955 00014384 18181818181818F818- <1>      db  018h, 018h, 018h, 018h, 018h, 018h, 018h, 0f8h, 018h, 018h, 018h, 018h, 018h, 018h, 018h,
018h
 52956 0001438D 18181818181818      <1>
 52957 00014394 1818181818F818F818- <1>      db  018h, 018h, 018h, 018h, 018h, 0f8h, 018h, 0f8h, 018h, 018h, 018h, 018h, 018h, 018h, 018h,
018h
 52958 0001439D 18181818181818      <1>
 52959 000143A4 36363636363636F636- <1>      db  036h, 036h, 036h, 036h, 036h, 036h, 036h, 0f6h, 036h, 036h, 036h, 036h, 036h, 036h, 036h,
036h
 52960 000143AD 36363636363636      <1>
 52961 000143B4 000000000000000FE36- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 0feh, 036h, 036h, 036h, 036h, 036h, 036h, 036h,
036h
 52962 000143BD 36363636363636      <1>
 52963 000143C4 0000000000F818F818- <1>      db  000h, 000h, 000h, 000h, 000h, 0f8h, 018h, 0f8h, 018h, 018h, 018h, 018h, 018h, 018h, 018h,
018h
 52964 000143CD 18181818181818      <1>
 52965 000143D4 3636363636F606F636- <1>      db  036h, 036h, 036h, 036h, 036h, 0f6h, 006h, 0f6h, 036h, 036h, 036h, 036h, 036h, 036h, 036h,
036h
 52966 000143DD 36363636363636      <1>
 52967 000143E4 3636363636363636- <1>       db  036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h,
036h
 52968 000143ED 36363636363636      <1>
 52969 000143F4 0000000000FE06F636- <1>      db  000h, 000h, 000h, 000h, 000h, 0feh, 006h, 0f6h, 036h, 036h, 036h, 036h, 036h, 036h, 036h,
036h
 52970 000143FD 36363636363636      <1>
 52971 00014404 3636363636F606FE00- <1>      db  036h, 036h, 036h, 036h, 036h, 0f6h, 006h, 0feh, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 52972 0001440D 00000000000000      <1>
 52973 00014414 36363636363636FE00- <1>      db  036h, 036h, 036h, 036h, 036h, 036h, 036h, 0feh, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 52974 0001441D 00000000000000      <1>
 52975 00014424 1818181818F818F800- <1>      db  018h, 018h, 018h, 018h, 018h, 0f8h, 018h, 0f8h, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 52976 0001442D 00000000000000      <1>
 52977 00014434 000000000000000F818- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 0f8h, 018h, 018h, 018h, 018h, 018h, 018h, 018h,
018h
 52978 0001443D 18181818181818      <1>
 52979 00014444 181818181818181F00- <1>      db  018h, 018h, 018h, 018h, 018h, 018h, 018h, 01fh, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 52980 0001444D 00000000000000      <1>
 52981 00014454 18181818181818FF00- <1>      db  018h, 018h, 018h, 018h, 018h, 018h, 018h, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 52982 0001445D 00000000000000      <1>
 52983 00014464 000000000000000FF18- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 0ffh, 018h, 018h, 018h, 018h, 018h, 018h, 018h,
018h
 52984 0001446D 18181818181818      <1>
 52985 00014474 181818181818181F18- <1>      db  018h, 018h, 018h, 018h, 018h, 018h, 018h, 01fh, 018h, 018h, 018h, 018h, 018h, 018h, 018h,
018h
 52986 0001447D 18181818181818      <1>
 52987 00014484 000000000000000FF00- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 52988 0001448D 00000000000000      <1>
 52989 00014494 18181818181818FF18- <1>      db  018h, 018h, 018h, 018h, 018h, 018h, 018h, 0ffh, 018h, 018h, 018h, 018h, 018h, 018h, 018h,
018h
 52990 0001449D 18181818181818      <1>
 52991 000144A4 18181818181F181F18- <1>      db  018h, 018h, 018h, 018h, 018h, 01fh, 018h, 01fh, 018h, 018h, 018h, 018h, 018h, 018h, 018h,
018h
 52992 000144AD 18181818181818      <1>
 52993 000144B4 3636363636363736- <1>       db  036h, 036h, 036h, 036h, 036h, 036h, 036h, 037h, 036h, 036h, 036h, 036h, 036h, 036h, 036h,
036h
 52994 000144BD 36363636363636      <1>
 52995 000144C4 3636363637303F00- <1>       db  036h, 036h, 036h, 036h, 036h, 037h, 030h, 03fh, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 52996 000144CD 00000000000000      <1>
 52997 000144D4 00000000003F303736- <1>      db  000h, 000h, 000h, 000h, 000h, 03fh, 030h, 037h, 036h, 036h, 036h, 036h, 036h, 036h, 036h,
036h
 52998 000144DD 36363636363636      <1>
```

```
 52999 000144E4 3636363636F700FF00- <1>     db  036h, 036h, 036h, 036h, 036h, 0f7h, 000h, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 53000 000144ED 00000000000000      <1>
 53001 000144F4 0000000000FF00F736- <1>     db  000h, 000h, 000h, 000h, 000h, 0ffh, 000h, 0f7h, 036h, 036h, 036h, 036h, 036h, 036h, 036h,
036h
 53002 000144FD 36363636363636      <1>
 53003 00014504 363636363637303736- <1>     db  036h, 036h, 036h, 036h, 036h, 037h, 030h, 037h, 036h, 036h, 036h, 036h, 036h, 036h, 036h,
036h
 53004 0001450D 36363636363636      <1>
 53005 00014514 0000000000FF00FF00- <1>     db  000h, 000h, 000h, 000h, 000h, 0ffh, 000h, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 53006 0001451D 00000000000000      <1>
 53007 00014524 3636363636F700F736- <1>     db  036h, 036h, 036h, 036h, 036h, 0f7h, 000h, 0f7h, 036h, 036h, 036h, 036h, 036h, 036h,
036h, 036h
 53008 0001452D 36363636363636      <1>
 53009 00014534 1818181818FF00FF00- <1>     db  018h, 018h, 018h, 018h, 018h, 0ffh, 000h, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 53010 0001453D 00000000000000      <1>
 53011 00014544 36363636363636FF00- <1>     db  036h, 036h, 036h, 036h, 036h, 036h, 036h, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 53012 0001454D 00000000000000      <1>
 53013 00014554 0000000000FF00FF18- <1>     db  000h, 000h, 000h, 000h, 000h, 0ffh, 000h, 0ffh, 018h, 018h, 018h, 018h, 018h, 018h, 018h,
018h
 53014 0001455D 18181818181818      <1>
 53015 00014564 000000000000000FF36- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 0ffh, 036h, 036h, 036h, 036h, 036h, 036h, 036h,
036h
 53016 0001456D 36363636363636      <1>
 53017 00014574 3636363636363F00- <1>     db  036h, 036h, 036h, 036h, 036h, 036h, 036h, 03fh, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 53018 0001457D 00000000000000      <1>
 53019 00014584 18181818181F181F00- <1>     db  018h, 018h, 018h, 018h, 018h, 01fh, 018h, 01fh, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 53020 0001458D 00000000000000      <1>
 53021 00014594 00000000001F181F18- <1>     db  000h, 000h, 000h, 000h, 000h, 01fh, 018h, 01fh, 018h, 018h, 018h, 018h, 018h, 018h, 018h,
018h
 53022 0001459D 18181818181818      <1>
 53023 000145A4 000000000000003F36- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 03fh, 036h, 036h, 036h, 036h, 036h, 036h, 036h,
036h
 53024 000145AD 36363636363636      <1>
 53025 000145B4 3636363636363FF36- <1>     db  036h, 036h, 036h, 036h, 036h, 036h, 036h, 0ffh, 036h, 036h, 036h, 036h, 036h, 036h, 036h,
036h
 53026 000145BD 36363636363636      <1>
 53027 000145C4 1818181818FF18FF18- <1>     db  018h, 018h, 018h, 018h, 018h, 0ffh, 018h, 0ffh, 018h, 018h, 018h, 018h, 018h, 018h, 018h,
018h
 53028 000145CD 18181818181818      <1>
 53029 000145D4 18181818181818F800- <1>     db  018h, 018h, 018h, 018h, 018h, 018h, 018h, 0f8h, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 53030 000145DD 00000000000000      <1>
 53031 000145E4 000000000000001F18- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 01fh, 018h, 018h, 018h, 018h, 018h, 018h, 018h,
018h
 53032 000145ED 18181818181818      <1>
 53033 000145F4 FFFFFFFFFFFFFFFFFF- <1>     db  0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh,
0ffh
 53034 000145FD FFFFFFFFFFFFFF      <1>
 53035 00014604 000000000000FFFF- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh,
0ffh
 53036 0001460D FFFFFFFFFFFFFF      <1>
 53037 00014614 F0F0F0F0F0F0F0F0F0- <1>     db  0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h,
0f0h
 53038 0001461D F0F0F0F0F0F0F0      <1>
 53039 00014624 0F0F0F0F0F0F0F0F0F- <1>     db  00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh,
00fh
 53040 0001462D 0F0F0F0F0F0F0F      <1>
 53041 00014634 FFFFFFFFFFFFF0000- <1>     db  0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 53042 0001463D 00000000000000      <1>
 53043 00014644 000000000076DCD8D8- <1>     db  000h, 000h, 000h, 000h, 000h, 076h, 0dch, 0d8h, 0d8h, 0d8h, 0dch, 076h, 000h, 000h, 000h,
000h
 53044 0001464D D8DC7600000000      <1>
 53045 00014654 000078CCCCCCD8CCC6- <1>     db  000h, 000h, 078h, 0cch, 0cch, 0cch, 0d8h, 0cch, 0c6h, 0c6h, 0c6h, 0cch, 000h, 000h, 000h,
000h
 53046 0001465D C6C6CC00000000      <1>
 53047 00014664 0000FEC6C6C0C0C0C0- <1>     db  000h, 000h, 0feh, 0c6h, 0c6h, 0c0h, 0c0h, 0c0h, 0c0h, 0c0h, 0c0h, 0c0h, 000h, 000h, 000h,
000h
 53048 0001466D C0C0C000000000      <1>
 53049 00014674 00000000FE6C6C6C6C- <1>     db  000h, 000h, 000h, 000h, 0feh, 06ch, 06ch, 06ch, 06ch, 06ch, 06ch, 06ch, 000h, 000h, 000h,
000h
 53050 0001467D 6C6C6C00000000      <1>
 53051 00014684 000000FEC660301830- <1>     db  000h, 000h, 000h, 0feh, 0c6h, 060h, 030h, 018h, 030h, 060h, 0c6h, 0feh, 000h, 000h, 000h,
000h
 53052 0001468D 60C6FE00000000      <1>
 53053 00014694 00000000007ED8D8D8- <1>     db  000h, 000h, 000h, 000h, 000h, 07eh, 0d8h, 0d8h, 0d8h, 0d8h, 0d8h, 070h, 000h, 000h, 000h,
000h
 53054 0001469D D8D87000000000      <1>
 53055 000146A4 0000000066666666666- <1>     db  000h, 000h, 000h, 000h, 066h, 066h, 066h, 066h, 066h, 07ch, 060h, 060h, 0c0h, 000h, 000h,
000h
 53056 000146AD 7C6060C0000000      <1>
 53057 000146B4 0000000076DC181818- <1>     db  000h, 000h, 000h, 000h, 076h, 0dch, 018h, 018h, 018h, 018h, 018h, 018h, 000h, 000h, 000h,
000h
 53058 000146BD 18181800000000      <1>
 53059 000146C4 0000007E183C666666- <1>     db  000h, 000h, 000h, 07eh, 018h, 03ch, 066h, 066h, 066h, 03ch, 018h, 07eh, 000h, 000h, 000h,
000h
 53060 000146CD 3C187E00000000      <1>
 53061 000146D4 000000386CC6C6FEC6- <1>     db  000h, 000h, 000h, 038h, 06ch, 0c6h, 0c6h, 0feh, 0c6h, 0c6h, 06ch, 038h, 000h, 000h, 000h,
000h
 53062 000146DD C66C3800000000      <1>
 53063 000146E4 0000386CC6C6C66C6C- <1>     db  000h, 000h, 038h, 06ch, 0c6h, 0c6h, 0c6h, 06ch, 06ch, 06ch, 06ch, 0eeh, 000h, 000h, 000h,
000h
 53064 000146ED 6C6CEE00000000      <1>
 53065 000146F4 00001E30180C3E6666- <1>     db  000h, 000h, 01eh, 030h, 018h, 00ch, 03eh, 066h, 066h, 066h, 066h, 03ch, 000h, 000h, 000h,
000h
 53066 000146FD 66663C00000000      <1>
```

```
 53067 00014704 00000000007EDBDBDB- <1>     db  000h, 000h, 000h, 000h, 000h, 07eh, 0dbh, 0dbh, 0dbh, 07eh, 000h, 000h, 000h, 000h, 000h,
000h
 53068 0001470D 7E000000000000      <1>
 53069 00014714 00000003067EDBDBF3- <1>     db  000h, 000h, 000h, 003h, 006h, 07eh, 0dbh, 0dbh, 0f3h, 07eh, 060h, 0c0h, 000h, 000h, 000h,
000h
 53070 0001471D 7E60C000000000      <1>
 53071 00014724 00001C3060607C6060- <1>     db  000h, 000h, 01ch, 030h, 060h, 060h, 07ch, 060h, 060h, 060h, 030h, 01ch, 000h, 000h, 000h,
000h
 53072 0001472D 60301C00000000      <1>
 53073 00014734 0000007CC6C6C6C6C6- <1>     db  000h, 000h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 000h, 000h, 000h,
000h
 53074 0001473D C6C6C600000000      <1>
 53075 00014744 00000000FE0000FE00- <1>     db  000h, 000h, 000h, 000h, 0feh, 000h, 000h, 0feh, 000h, 000h, 0feh, 000h, 000h, 000h,
000h, 000h
 53076 0001474D 00FE0000000000      <1>
 53077 00014754 0000000018187E1818- <1>     db  000h, 000h, 000h, 000h, 018h, 018h, 07eh, 018h, 018h, 000h, 000h, 0ffh, 000h, 000h, 000h,
000h
 53078 0001475D 0000FF00000000      <1>
 53079 00014764 00000030180C060C18- <1>     db  000h, 000h, 000h, 030h, 018h, 00ch, 006h, 00ch, 018h, 030h, 000h, 07eh, 000h, 000h, 000h,
000h
 53080 0001476D 30007E00000000      <1>
 53081 00014774 0000000C1830603018- <1>     db  000h, 000h, 000h, 00ch, 018h, 030h, 060h, 030h, 018h, 00ch, 000h, 07eh, 000h, 000h, 000h,
000h
 53082 0001477D 0C007E00000000      <1>
 53083 00014784 00000E1B1B18181818- <1>     db  000h, 000h, 00eh, 01bh, 01bh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h,
018h
 53084 0001478D 18181818181818      <1>
 53085 00014794 1818181818181818D8- <1>     db  018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 0d8h, 0d8h, 0d8h, 070h, 000h, 000h, 000h,
000h
 53086 0001479D D8D87000000000      <1>
 53087 000147A4 000000001818007E00- <1>     db  000h, 000h, 000h, 000h, 018h, 018h, 000h, 07eh, 000h, 018h, 018h, 000h, 000h, 000h, 000h,
000h
 53088 000147AD 18180000000000      <1>
 53089 000147B4 000000000076DC0076- <1>     db  000h, 000h, 000h, 000h, 000h, 076h, 0dch, 000h, 076h, 0dch, 000h, 000h, 000h, 000h, 000h,
000h
 53090 000147BD DC000000000000      <1>
 53091 000147C4 00386C6C3800000000- <1>     db  000h, 038h, 06ch, 06ch, 038h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 53092 000147CD 00000000000000      <1>
 53093 000147D4 0000000000001818- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 018h, 018h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 53094 000147DD 00000000000000      <1>
 53095 000147E4 0000000000000018- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 018h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 53096 000147ED 00000000000000      <1>
 53097 000147F4 000F0C0C0C0C0CEC6C- <1>     db  000h, 00fh, 00ch, 00ch, 00ch, 00ch, 00ch, 0ech, 06ch, 06ch, 03ch, 01ch, 000h, 000h, 000h,
000h
 53098 000147FD 6C3C1C00000000      <1>
 53099 00014804 00D86C6C6C6C6C0000- <1>     db  000h, 0d8h, 06ch, 06ch, 06ch, 06ch, 06ch, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 53100 0001480D 00000000000000      <1>
 53101 00014814 0070D83060C8F80000- <1>     db  000h, 070h, 0d8h, 030h, 060h, 0c8h, 0f8h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 53102 0001481D 00000000000000      <1>
 53103 00014824 000000007C7C7C7C7C- <1>     db  000h, 000h, 000h, 000h, 07ch, 07ch, 07ch, 07ch, 07ch, 07ch, 07ch, 000h, 000h, 000h, 000h,
000h
 53104 0001482D 7C7C0000000000      <1>
 53105 00014834 0000000000000000- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 53106 0001483D 00000000000000      <1>
 53107                              <1> vgafont14alt:
 53108 00014844 1D000000002466FF66- <1>     db  01dh, 000h, 000h, 000h, 000h, 024h, 066h, 0ffh, 066h, 024h, 000h, 000h, 000h, 000h, 000h,
022h
 53109 0001484D 24000000000022      <1>
 53110 00014854 006363632200000000- <1>     db  000h, 063h, 063h, 063h, 022h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 02bh,
000h
 53111 0001485D 00000000002B00      <1>
 53112 00014864 0000181818FF181818- <1>     db  000h, 000h, 018h, 018h, 018h, 0ffh, 018h, 018h, 018h, 000h, 000h, 000h, 000h, 02dh, 000h,
000h
 53113 0001486D 000000002D0000      <1>
 53114 00014874 00000000FF00000000- <1>     db  000h, 000h, 000h, 000h, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 04dh, 000h, 000h,
0c3h
 53115 0001487D 0000004D0000C3      <1>
 53116 00014884 E7FFDBC3C3C3C3C300- <1>     db  0e7h, 0ffh, 0dbh, 0c3h, 0c3h, 0c3h, 0c3h, 0c3h, 000h, 000h, 000h, 054h, 000h, 000h, 0ffh,
0dbh
 53117 0001488D 0000540000FFDB      <1>
 53118 00014894 9918181818183C0000- <1>     db  099h, 018h, 018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h, 056h, 000h, 000h, 0c3h, 0c3h,
0c3h
 53119 0001489D 00560000C3C3C3      <1>
 53120 000148A4 C3C3C3663C18000000- <1>     db  0c3h, 0c3h, 0c3h, 066h, 03ch, 018h, 000h, 000h, 000h, 057h, 000h, 000h, 0c3h, 0c3h, 0c3h,
0c3h
 53121 000148AD 570000C3C3C3C3      <1>
 53122 000148B4 DBDBFF666600000058- <1>     db  0dbh, 0dbh, 0ffh, 066h, 066h, 000h, 000h, 000h, 058h, 000h, 000h, 0c3h, 0c3h, 066h, 03ch,
018h
 53123 000148BD 0000C3C3663C18      <1>
 53124 000148C4 3C66C3C30000005900- <1>     db  03ch, 066h, 0c3h, 0c3h, 000h, 000h, 000h, 059h, 000h, 000h, 0c3h, 0c3h, 0c3h, 066h, 03ch,
018h
 53125 000148CD 00C3C3C3663C18      <1>
 53126 000148D4 18183C0000005A0000- <1>     db  018h, 018h, 03ch, 000h, 000h, 000h, 05ah, 000h, 000h, 0ffh, 0c3h, 086h, 00ch, 018h, 030h,
061h
 53127 000148DD FFC3860C183061      <1>
 53128 000148E4 C3FF0000006D000000- <1>     db  0c3h, 0ffh, 000h, 000h, 000h, 06dh, 000h, 000h, 000h, 000h, 000h, 0e6h, 0ffh, 0dbh, 0dbh,
0dbh
 53129 000148ED 0000E6FFDBDBDB      <1>
 53130 000148F4 DB0000007600000000- <1>     db  0dbh, 000h, 000h, 000h, 076h, 000h, 000h, 000h, 000h, 000h, 0c3h, 0c3h, 0c3h, 066h, 03ch,
018h
 53131 000148FD 00C3C3C3663C18      <1>
 53132 00014904 0000007700000000- <1>     db  000h, 000h, 000h, 077h, 000h, 000h, 000h, 000h, 000h, 0c3h, 0c3h, 0dbh, 0dbh, 0ffh, 066h,
000h
 53133 0001490D C3C3DBDBFF6600      <1>
 53134 00014914 000091000000006E3B- <1>     db  000h, 000h, 091h, 000h, 000h, 000h, 000h, 06eh, 03bh, 01bh, 07eh, 0d8h, 0dch, 077h, 000h,
000h
```

```
 53135 0001491D 1B7ED8DC770000      <1>
 53136 00014924 009B0018187EC3C0C0- <1>     db  000h, 09bh, 000h, 018h, 018h, 07eh, 0c3h, 0c0h, 0c0h, 0c3h, 07eh, 018h, 018h, 000h, 000h,
000h
 53137 0001492D C37E1818000000      <1>
 53138 00014934 9D0000C3663C18FF18- <1>     db  09dh, 000h, 000h, 0c3h, 066h, 03ch, 018h, 0ffh, 018h, 0ffh, 018h, 018h, 000h, 000h, 000h,
09eh
 53139 0001493D FF18180000009E      <1>
 53140 00014944 00FC66667C62666F66- <1>     db  000h, 0fch, 066h, 066h, 07ch, 062h, 066h, 06fh, 066h, 066h, 0f3h, 000h, 000h, 000h, 0f1h,
000h
 53141 0001494D 66F3000000F100      <1>
 53142 00014954 00181818FF18181800- <1>     db  000h, 018h, 018h, 018h, 0ffh, 018h, 018h, 018h, 000h, 0ffh, 000h, 000h, 000h, 0f6h,
000h, 000h
 53143 0001495D FF000000F60000      <1>
 53144 00014964 18180000FF00001818- <1>     db  018h, 018h, 000h, 000h, 0ffh, 000h, 000h, 018h, 018h, 000h, 000h, 000h, 000h
 53145 0001496D 00000000            <1>
 53146                              <1> vgafont16alt:
 53147 00014971 1D00000000002466FF- <1>     db  01dh, 000h, 000h, 000h, 000h, 000h, 024h, 066h, 0ffh, 066h, 024h, 000h, 000h, 000h, 000h,
000h
 53148 0001497A 66240000000000      <1>
 53149 00014981 003000003C66C3C3DB- <1>     db  000h, 030h, 000h, 000h, 03ch, 066h, 0c3h, 0c3h, 0dbh, 0dbh, 0c3h, 0c3h, 066h, 03ch, 000h,
000h
 53150 0001498A DBC3C3663C0000      <1>
 53151 00014991 00004D0000C3E7FFFF- <1>     db  000h, 000h, 04dh, 000h, 000h, 0c3h, 0e7h, 0ffh, 0ffh, 0dbh, 0c3h, 0c3h, 0c3h, 0c3h, 0c3h,
000h
 53152 0001499A DBC3C3C3C3C300      <1>
 53153 000149A1 000000540000FFDB99- <1>     db  000h, 000h, 000h, 054h, 000h, 000h, 0ffh, 0dbh, 099h, 018h, 018h, 018h, 018h, 018h, 018h,
03ch
 53154 000149AA 1818181818183C      <1>
 53155 000149B1 00000000560000C3C3- <1>     db  000h, 000h, 000h, 000h, 056h, 000h, 000h, 0c3h, 0c3h, 0c3h, 0c3h, 0c3h, 0c3h, 0c3h, 066h,
03ch
 53156 000149BA C3C3C3C3C3663C      <1>
 53157 000149C1 1800000000570000C3- <1>     db  018h, 000h, 000h, 000h, 000h, 057h, 000h, 000h, 0c3h, 0c3h, 0c3h, 0c3h, 0c3h, 0dbh, 0dbh,
0ffh
 53158 000149CA C3C3C3C3DBDBFF      <1>
 53159 000149D1 666600000000580000- <1>     db  066h, 066h, 000h, 000h, 000h, 000h, 058h, 000h, 000h, 0c3h, 0c3h, 066h, 03ch, 018h, 018h,
03ch
 53160 000149DA C3C3663C18183C      <1>
 53161 000149E1 66C3C3000000005900- <1>     db  066h, 0c3h, 0c3h, 000h, 000h, 000h, 000h, 059h, 000h, 000h, 0c3h, 0c3h, 0c3h, 066h, 03ch,
018h
 53162 000149EA 00C3C3C3663C18      <1>
 53163 000149F1 1818183C000000005A- <1>     db  018h, 018h, 018h, 03ch, 000h, 000h, 000h, 000h, 05ah, 000h, 000h, 0ffh, 0c3h, 086h, 00ch,
018h
 53164 000149FA 0000FFC3860C18      <1>
 53165 00014A01 3060C1C3FF00000000- <1>     db  030h, 060h, 0c1h, 0c3h, 0ffh, 000h, 000h, 000h, 000h, 06dh, 000h, 000h, 000h, 000h, 000h,
0e6h
 53166 00014A0A 6D0000000000E6      <1>
 53167 00014A11 FFDBDBDBDBDB000000- <1>     db  0ffh, 0dbh, 0dbh, 0dbh, 0dbh, 0dbh, 000h, 000h, 000h, 000h, 076h, 000h, 000h, 000h, 000h,
000h
 53168 00014A1A 00760000000000      <1>
 53169 00014A21 C3C3C3C3663C180000- <1>     db  0c3h, 0c3h, 0c3h, 0c3h, 066h, 03ch, 018h, 000h, 000h, 000h, 000h, 077h, 000h, 000h, 000h,
000h
 53170 00014A2A 00007700000000      <1>
 53171 00014A31 00C3C3C3DBDBFF6600- <1>     db  000h, 0c3h, 0c3h, 0c3h, 0dbh, 0dbh, 0ffh, 066h, 000h, 000h, 000h, 000h, 078h, 000h, 000h,
000h
 53172 00014A3A 00000078000000      <1>
 53173 00014A41 0000C3663C183C66C3- <1>     db  000h, 000h, 0c3h, 066h, 03ch, 018h, 03ch, 066h, 0c3h, 000h, 000h, 000h, 000h, 091h, 000h,
000h
 53174 00014A4A 00000000910000      <1>
 53175 00014A51 0000006E3B1B7ED8DC- <1>     db  000h, 000h, 000h, 06eh, 03bh, 01bh, 07eh, 0d8h, 0dch, 077h, 000h, 000h, 000h, 000h, 09bh,
000h
 53176 00014A5A 77000000009B00      <1>
 53177 00014A61 18187EC3C0C0C0C37E- <1>     db  018h, 018h, 07eh, 0c3h, 0c0h, 0c0h, 0c0h, 0c3h, 07eh, 018h, 018h, 000h, 000h, 000h, 000h,
09dh
 53178 00014A6A 1818000000009D      <1>
 53179 00014A71 0000C3663C18FF18FF- <1>     db  000h, 000h, 0c3h, 066h, 03ch, 018h, 0ffh, 018h, 0ffh, 018h, 018h, 018h, 000h, 000h, 000h,
000h
 53180 00014A7A 18181800000000      <1>
 53181 00014A81 9E00FC66667C62666F- <1>     db  09eh, 000h, 0fch, 066h, 066h, 07ch, 062h, 066h, 06fh, 066h, 066h, 066h, 0f3h, 000h, 000h,
000h
 53182 00014A8A 666666F3000000      <1>
 53183 00014A91 00AB00C0C0C2C6CC18- <1>     db  000h, 0abh, 000h, 0c0h, 0c0h, 0c2h, 0c6h, 0cch, 018h, 030h, 060h, 0ceh, 09bh, 006h, 00ch,
01fh
 53184 00014A9A 3060CE9B060C1F      <1>
 53185 00014AA1 0000AC00C0C0C2C6CC- <1>     db  000h, 000h, 0ach, 000h, 0c0h, 0c0h, 0c2h, 0c6h, 0cch, 018h, 030h, 066h, 0ceh, 096h, 03eh,
006h
 53186 00014AAA 183066CE963E06      <1>
 53187 00014AB1 06000000            <1>     db  006h, 000h, 000h, 000h
 53188
 53189 00014AB5 90                      align 2
 53190
 53191                                  ; EPOCH Variables
 53192                                  ; 13/04/2015 - Retro UNIX 386 v1 Beginning
 53193                                  ; 09/04/2013 epoch variables
 53194                                  ; Retro UNIX 8086 v1 Prototype: UNIXCOPY.ASM, 10/03/2013
 53195                                  ;
 53196 00014AB6 B207                    year:       dw 1970
 53197 00014AB8 0100                    month:      dw 1
 53198 00014ABA 0100                    day:  dw 1
 53199 00014ABC 0000                    hour:       dw 0
 53200 00014ABE 0000                    minute: dw 0
 53201 00014AC0 0000                    second: dw 0
 53202
 53203                                  DMonth:
 53204 00014AC2 0000                          dw 0
 53205 00014AC4 1F00                          dw 31
 53206 00014AC6 3B00                          dw 59
 53207 00014AC8 5A00                          dw 90
 53208 00014ACA 7800                          dw 120
 53209 00014ACC 9700                          dw 151
 53210 00014ACE B500                          dw 181
 53211 00014AD0 D400                          dw 212
 53212 00014AD2 F300                          dw 243
```

```
53213 00014AD4 1101                     dw 273
53214 00014AD6 3001                     dw 304
53215 00014AD8 4E01                     dw 334
53216
53217                                 ; 20/02/2017
53218                                 KERNELFSIZE  equ $  ; 04/07/2016
53219
53220                                 bss_start:
53221
53222                                 ABSOLUTE bss_start
53223
53224 00014ADA <res 00000006>         alignb 8 ; 25/12/2016
53225
53226                                       ; 15/04/2016
53227                                       ; TRDOS 386 (TRDOS v2.0)
53228                                       ;     80 interrupts
53229                                       ; 11/03/2015
53230                                       ; Interrupt Descriptor Table (20/08/2014)
53231                                 idt:
53232                                       ;resb  64*8 ; INT 0 to INT 3Fh
53233                                       ; 15/04/2016
53234 00014AE0 <res 00000280>               resb   80*8 ; INT 0 to INT 4Fh
53235
53236                                 idt_end:
53237
53238                                 ;alignb 4
53239
53240                                 task_state_segment:
53241                                       ; 24/03/2015
53242 00014D60 <res 00000002>         tss.link:   resw 1
53243 00014D62 <res 00000002>                     resw 1
53244                                 ; tss offset 4
53245 00014D64 <res 00000004>         tss.esp0:   resd 1
53246 00014D68 <res 00000002>         tss.ss0:    resw 1
53247 00014D6A <res 00000002>                     resw 1
53248 00014D6C <res 00000004>         tss.esp1:   resd 1
53249 00014D70 <res 00000002>         tss.ss1:    resw 1
53250 00014D72 <res 00000002>                     resw 1
53251 00014D74 <res 00000004>         tss.esp2:   resd 1
53252 00014D78 <res 00000002>         tss.ss2:    resw 1
53253 00014D7A <res 00000002>                     resw 1
53254                                 ; tss offset 28
53255 00014D7C <res 00000004>         tss.CR3:    resd 1
53256 00014D80 <res 00000004>         tss.eip:    resd 1
53257 00014D84 <res 00000004>         tss.eflags: resd 1
53258                                 ; tss offset 40
53259 00014D88 <res 00000004>         tss.eax:    resd 1
53260 00014D8C <res 00000004>         tss.ecx:    resd 1
53261 00014D90 <res 00000004>         tss.edx:    resd 1
53262 00014D94 <res 00000004>         tss.ebx:    resd 1
53263 00014D98 <res 00000004>         tss.esp:    resd 1
53264 00014D9C <res 00000004>         tss.ebp:    resd 1
53265 00014DA0 <res 00000004>         tss.esi:    resd 1
53266 00014DA4 <res 00000004>         tss.edi:    resd 1
53267                                 ; tss offset 72
53268 00014DA8 <res 00000002>         tss.ES:     resw 1
53269 00014DAA <res 00000002>                     resw 1
53270 00014DAC <res 00000002>         tss.CS:           resw 1
53271 00014DAE <res 00000002>                     resw 1
53272 00014DB0 <res 00000002>         tss.SS:           resw 1
53273 00014DB2 <res 00000002>                     resw 1
53274 00014DB4 <res 00000002>         tss.DS:           resw 1
53275 00014DB6 <res 00000002>                     resw 1
53276 00014DB8 <res 00000002>         tss.FS:           resw 1
53277 00014DBA <res 00000002>                     resw 1
53278 00014DBC <res 00000002>         tss.GS:           resw 1
53279 00014DBE <res 00000002>                     resw 1
53280 00014DC0 <res 00000002>         tss.LDTR:   resw 1
53281 00014DC2 <res 00000002>                     resw 1
53282                                 ; tss offset 100
53283 00014DC4 <res 00000002>                     resw 1
53284 00014DC6 <res 00000002>         tss.IOPB:   resw 1
53285                                 ; tss offset 104
53286                                 tss_end:
53287
53288 00014DC8 <res 00000004>         k_page_dir:  resd 1 ; Kernel's (System) Page Directory address
53289                                                   ; (Physical address = Virtual address)
53290 00014DCC <res 00000004>         memory_size: resd 1 ; memory size in pages
53291 00014DD0 <res 00000004>         free_pages:  resd 1 ; number of free pages
53292 00014DD4 <res 00000004>         next_page:   resd 1 ; offset value in M.A.T. for
53293                                                   ; first free page search
53294 00014DD8 <res 00000004>         last_page:   resd 1 ; offset value in M.A.T. which
53295                                                   ; next free page search will be
53296                                                   ; stopped after it. (end of M.A.T.)
53297 00014DDC <res 00000004>         first_page:  resd 1 ; offset value in M.A.T. which
53298                                                   ; first free page search
53299                                                   ; will be started on it. (for user)
53300 00014DE0 <res 00000004>         mat_size:    resd 1 ; Memory Allocation Table size in pages
53301
53302                                 ; 02/09/2014 (Retro UNIX 386 v1)
53303                                 ; 04/12/2013 (Retro UNIX 8086 v1)
53304 00014DE4 <res 00000002>         CRT_START:   resw 1        ; starting address in regen buffer
53305                                                ; NOTE: active page only
53306 00014DE6 <res 00000010>         CURSOR_POSN: resw 8 ; cursor positions for video pages
53307                                 ACTIVE_PAGE:
53308 00014DF6 <res 00000001>         ptty:            resb 1 ; current tty
53309                                 ; 01/07/2015 - 29/01/2016
53310 00014DF7 <res 00000001>         ccolor:          resb 1 ; current color attribute
53311                                 ; 26/10/2015
53312                                 ; 07/09/2014
53313 00014DF8 <res 00000014>         ttychr:      resw ntty+2 ; Character buffer (multiscreen)
53314
```

```
                                        ; 18/05/2015 (03/06/2013 - Retro UNIX 8086 v1 feature only!)
00014E0C <res 00000004>                 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) ]
00014E10 <res 00000014>                 ttyl:        resw ntty+2 ; opening locks for TTYs.

                                        ; 15/04/2015 (Retro UNIX 386 v1)
                                        ; 22/09/2013 (Retro UNIX 8086 v1)
00014E24 <res 0000000A>                 wlist:       resb ntty+2 ; wait channel list (0 to 9 for TTYs)
                                        ; 15/04/2015 (Retro UNIX 386 v1)
                                        ;; 12/07/2014 -> sp_init set comm. parameters as 0E3h
                                        ;; 0 means serial port is not available
                                        ;;comprm: ; 25/06/2014
00014E2E <res 00000001>                 com1p:       resb 1  ;;0E3h
00014E2F <res 00000001>                 com2p:       resb 1  ;;0E3h

                                        ; 17/11/2015
                                        ; request for response (from the terminal)
00014E30 <res 00000002>                 req_resp:    resw 1
                                        ; 07/11/2015
00014E32 <res 00000001>                 ccomport:    resb 1 ; current COM (serial) port
                                                        ; (0= COM1, 1= COM2)
                                        ; 09/11/2015
00014E33 <res 00000001>                 comqr:           resb 1 ; 'query or response' sign (u9.s, 'sndc')
                                        ; 07/11/2015
00014E34 <res 00000002>                 rchar:           resw 1 ; last received char for COM 1 and COM 2
00014E36 <res 00000002>                 schar:           resw 1 ; last sent char for COM 1 and COM 2

                                        ; 22/08/2014 (RTC)
                                        ; (Packed BCD)
00014E38 <res 00000001>                 time_seconds: resb 1
00014E39 <res 00000001>                 time_minutes: resb 1
00014E3A <res 00000001>                 time_hours:   resb 1
00014E3B <res 00000001>                 date_wday:    resb 1
00014E3C <res 00000001>                 date_day:     resb 1
00014E3D <res 00000001>                 date_month:   resb 1
00014E3E <res 00000001>                 date_year:    resb 1
00014E3F <res 00000001>                 date_century: resb 1

                                        ; 24/01/2016
00014E40 <res 00000004>                 RTC_LH:            resd 1
00014E44 <res 00000001>                 RTC_WAIT_FLAG: resb 1
00014E45 <res 00000001>                 USER_FLAG:     resb 1
                                        ; 19/05/2016
                                        ;RTC_second:
00014E46 <res 00000001>                 RTC_2Hz:       resb 1 ;  from 2Hz interrupt to 1Hz timer event function

                                    %include 'diskbss.s'     ; UNINITIALIZED DISK (BIOS) DATA
                                <1> ; ****************************************************************************
                                <1> ; TRDOS386.ASM (TRDOS 386 Kernel) - v2.0.0 - diskbss.s
                                <1> ; ----------------------------------------------------------------------------
                                <1> ; Last Update: 24/01/2016
                                <1> ; ----------------------------------------------------------------------------
                                <1> ; Beginning: 24/01/2016
                                <1> ; ----------------------------------------------------------------------------
                                <1> ; Assembler: NASM version 2.11 (trdos386.s)
                                <1> ; ----------------------------------------------------------------------------
                                <1> ; Turkish Rational DOS
                                <1> ; Operating System Project v2.0 by ERDOGAN TAN (Beginning: 04/01/2016)
                                <1> ;
                                <1> ; Derived from 'Retro UNIX 386 Kernel - v0.2.1.0' source code by Erdogan Tan
                                <1> ; diskbss.inc (10/07/2015)
                                <1> ;
                                <1> ; Derived from 'IBM PC-XT-286' BIOS source code (1986)
                                <1> ; ****************************************************************************
                                <1>
                                <1> ; Retro UNIX 386 v1 Kernel - DISKBSS.INC
                                <1> ; Last Modification: 10/07/2015
                                <1> ;     (Unnitialized Disk Parameters Data section for 'DISKIO.INC')
                                <1>
00014E47 <res 00000001>         <1> alignb 2
                                <1>
                                <1> ;----------------------------------------
                                <1> ;     TIMER DATA AREA           :
                                <1> ;----------------------------------------
                                <1>
                                <1> TIMER_LH:    ; 16/02/205
00014E48 <res 00000002>         <1> TIMER_LOW:      resw     1               ; LOW WORD OF TIMER COUNT
00014E4A <res 00000002>         <1> TIMER_HIGH:     resw     1               ; HIGH WORD OF TIMER COUNT
00014E4C <res 00000001>         <1> TIMER_OFL:      resb     1               ; TIMER HAS ROLLED OVER SINCE LAST READ
                                <1>
                                <1> ;----------------------------------------
                                <1> ;     DISKETTE DATA AREAS       :
                                <1> ;----------------------------------------
                                <1>
00014E4D <res 00000001>         <1> SEEK_STATUS: resb   1
00014E4E <res 00000001>         <1> MOTOR_STATUS:      resb   1
00014E4F <res 00000001>         <1> MOTOR_COUNT: resb   1
00014E50 <res 00000001>         <1> DSKETTE_STATUS:    resb   1
00014E51 <res 00000007>         <1> NEC_STATUS:  resb   7
                                <1>
                                <1> ;----------------------------------------
                                <1> ;     ADDITIONAL MEDIA DATA           :
                                <1> ;----------------------------------------
                                <1>
00014E58 <res 00000001>         <1> LASTRATE:    resb   1
00014E59 <res 00000001>         <1> HF_STATUS:   resb   1
00014E5A <res 00000001>         <1> HF_ERROR:    resb   1
00014E5B <res 00000001>         <1> HF_INT_FLAG: resb   1
00014E5C <res 00000001>         <1> HF_CNTRL:    resb   1
```

```
53417 00014E5D <res 00000004>      <1> DSK_STATE:   resb   4
53418 00014E61 <res 00000002>      <1> DSK_TRK:     resb   2
53419                              <1>
53420                              <1> ;----------------------------------------
53421                              <1> ;     FIXED DISK DATA AREAS            :
53422                              <1> ;----------------------------------------
53423                              <1>
53424 00014E63 <res 00000001>      <1> DISK_STATUS1:      resb   1            ; FIXED DISK STATUS
53425 00014E64 <res 00000001>      <1> HF_NUM:            resb   1            ; COUNT OF FIXED DISK DRIVES
53426 00014E65 <res 00000001>      <1> CONTROL_BYTE:      resb   1            ; HEAD CONTROL BYTE
53427                              <1> ;@PORT_OFF   resb   1            ; RESERVED (PORT OFFSET)
53428                              <1> ;port1_off   resb   1            ; Hard disk controller 1 - port offset
53429                              <1> ;port2_off   resb   1            ; Hard idsk controller 2 - port offset
53430                              <1>
53431 00014E66 <res 00000002>      <1> alignb 4
53432                              <1>
53433                              <1> ;HF_TBL_VEC: resd   1            ; Primary master disk param. tbl. pointer
53434                              <1> ;HF1_TBL_VEC:      resd   1            ; Primary slave disk param. tbl. pointer
53435                              <1> HF_TBL_VEC: ; 22/12/2014
53436 00014E68 <res 00000004>      <1> HDPM_TBL_VEC:      resd   1            ; Primary master disk param. tbl. pointer
53437 00014E6C <res 00000004>      <1> HDPS_TBL_VEC:      resd   1            ; Primary slave disk param. tbl. pointer
53438 00014E70 <res 00000004>      <1> HDSM_TBL_VEC:      resd   1            ; Secondary master disk param. tbl. pointer
53439 00014E74 <res 00000004>      <1> HDSS_TBL_VEC:      resd   1            ; Secondary slave disk param. tbl. pointer
53440                              <1>
53441                              <1> ; 03/01/2015
53442 00014E78 <res 00000001>      <1> LBAMode:           resb   1
53443                              <1>
53444                              <1> ; ****************************************************************************
53445
53446                                  ;;; Real Mode Data (10/07/2015 - BSS)
53447
53448                                  ;alignb 2
53449
53450                                  ; 10/01/2016
53451                                  %include 'trdoskx.s'     ; UNINITIALIZED KERNEL (Logical Drive & FS) DATA
53452                              <1> ; ****************************************************************************
53453                              <1> ; TRDOS386.ASM (TRDOS 386 Kernel - v2.0.0) - UNINITIALIZED DATA : trdoskx.s
53454                              <1> ; ----------------------------------------------------------------------------
53455                              <1> ; Last Update: 28/05/2017
53456                              <1> ; ----------------------------------------------------------------------------
53457                              <1> ; Beginning: 04/01/2016
53458                              <1> ; ----------------------------------------------------------------------------
53459                              <1> ; Assembler: NASM version 2.11 (trdos386.s)
53460                              <1> ; ----------------------------------------------------------------------------
53461                              <1> ; Derived from TRDOS Operating System v1.0 (8086) source code by Erdogan Tan
53462                              <1> ; TRDOS2.ASM (09/11/2011)
53463                              <1> ; ****************************************************************************
53464                              <1> ; DRV_INIT.ASM [26/09/2009] Last Update: 07/08/2011
53465                              <1> ; MAINPROG.ASM [17/01/2004] Last Update: 09/11/2011
53466                              <1> ; DIR.ASM      [17/01/2004] Last Update: 09/10/2011
53467                              <1> ; CMD_INTR.ASM [29/01/2005] Last update: 09/11/2011
53468                              <1> ; DRV_FAT.ASM  [07/07/2009] Last update: 21/08/2011
53469                              <1>
53470 00014E79 <res 00000003>      <1> alignb 4
53471                              <1>
53472                              <1> ; MAINPROG.ASM
53473 00014E7C <res 00000004>      <1> MainProgCfg_FileSize:   resd 1 ; 14/04/2016
53474 00014E80 <res 00000004>      <1> MainProgCfg_LineOffset: resd 1 ; 14/04/2016
53475                              <1>
53476 00014E84 <res 00000004>      <1> Current_VolSerial: resd 1
53477                              <1>
53478 00014E88 <res 00000004>      <1> Current_Dir_FCluster: resd 1
53479                              <1>
53480 00014E8C <res 00000001>      <1> Current_Dir_Level: resb 1
53481 00014E8D <res 00000001>      <1> Current_FATType: resb 1
53482 00014E8E <res 00000001>      <1> Current_Drv: resb 1
53483 00014E8F <res 00000001>      <1> Current_Dir_Drv:   resb 1 ; '?'
53484 00014E90 <res 00000001>      <1>                    resb 1 ; ':'
53485 00014E91 <res 00000001>      <1> Current_Dir_Root:  resb 1 ; '/'
53486 00014E92 <res 0000005A>      <1> Current_Directory: resb 90
53487 00014EEC <res 00000001>      <1> End_Of_Current_Dir_Str: resb 1
53488 00014EED <res 00000001>      <1> Current_Dir_StrLen: resb 1
53489                              <1>
53490 00014EEE <res 00000001>      <1> CursorColumn:      resb 1
53491 00014EEF <res 00000001>      <1> CmdArgStart:    resb 1
53492                              <1>
53493                              <1> ; 03/02/2016
53494 00014EF0 <res 0000004E>      <1> Remark:            resb 78
53495                              <1>
53496 00014F3E <res 00000050>      <1> CommandBuffer:     resb 80
53497                              <1>
53498 00014F8E <res 00000100>      <1> TextBuffer:  resb 256
53499                              <1>
53500                              <1> MasterBootBuff:
53501 0001508E <res 000001BE>      <1> MasterBootCode: resb 1BEh
53502 0001524C <res 00000040>      <1> PartitionTable: resb 64
53503 0001528C <res 00000002>      <1> MBIDCode: resw 1
53504                              <1>
53505                              <1> PTable_Buffer:
53506 0001528E <res 00000040>      <1> PTable_hd0: resb 64
53507 000152CE <res 00000040>      <1> PTable_hd1: resb 64
53508 0001530E <res 00000040>      <1> PTable_hd2: resb 64
53509 0001534E <res 00000040>      <1> PTable_hd3: resb 64
53510 0001538E <res 00000040>      <1> PTable_ep0: resb 64
53511 000153CE <res 00000040>      <1> PTable_ep1: resb 64
53512 0001540E <res 00000040>      <1> PTable_ep2: resb 64
53513 0001544E <res 00000040>      <1> PTable_ep3: resb 64
53514                              <1>
53515 0001548E <res 00000001>      <1> scount:     resb 1 ; 16/05/2016 (diskio.s, 'int33h:')
53516 0001548F <res 00000001>      <1> HD_LBA_yes: resb 1
53517 00015490 <res 00000001>      <1> PP_Counter: resb 1
53518 00015491 <res 00000001>      <1> EP_Counter: resb 1
```

```
                                <1>
00015492 <res 00000004>         <1> EP_StartSector: resd 1
00015496 <res 00000004>         <1>                 resd 1
0001549A <res 00000004>         <1>                 resd 1
0001549E <res 00000004>         <1>                 resd 1
                                <1>
000154A2 <res 00000200>         <1> DOSBootSectorBuff: resb 512
                                <1>
                                <1> FAT_BuffDescriptor:
000156A2 <res 00000004>         <1> FAT_CurrentCluster: resd 1
000156A6 <res 00000001>         <1> FAT_BuffValidData: resb 1
000156A7 <res 00000001>         <1> FAT_BuffDrvName: resb 1
000156A8 <res 00000002>         <1> FAT_BuffOffset: resw 1
000156AA <res 00000004>         <1> FAT_BuffSector: resd 1
                                <1>
000156AE <res 00000004>         <1> FAT_ClusterCounter: resd 1
000156B2 <res 00000004>         <1> LastCluster: resd 1
                                <1>
                                <1> ; 16/05/2016
                                <1> ;; 18/03/2016 (TRDOS v2.0)
                                <1> ;ClusterBuffer_Valid: resb 1
                                <1>
                                <1> Dir_BuffDescriptor:
000156B6 <res 00000001>         <1> DirBuff_DRV: resb 1
000156B7 <res 00000001>         <1> DirBuff_FATType: resb 1
000156B8 <res 00000001>         <1> DirBuff_ValidData: resb 1
000156B9 <res 00000002>         <1> DirBuff_CurrentEntry: resw 1
000156BB <res 00000002>         <1> DirBuff_LastEntry: resw 1
000156BD <res 00000004>         <1> DirBuff_Cluster: resd 1
000156C1 <res 00000002>         <1> DirBuffer_Size: resw 1
                                <1> ;DirBuff_EntryCounter: resw 1
                                <1>
                                <1> ; 01/02/2016
                                <1> ; these are on (real mode) segment 8000h and later
                                <1> ; FAT_Buffer:      resb 1536 ; 3 sectors
                                <1> ; Dir_Buffer:      resb 512*32
                                <1> ; Logical_DOSDisks:  resb 6656 ; 26 * 256 bytes
                                <1>
                                <1> ; 18/01/2016
                                <1>
000156C3 <res 00000004>         <1> FreeClusterCount: resd 1
                                <1>
000156C7 <res 00000004>         <1> VolSize_Unit1:   resd 1
000156CB <res 00000004>         <1> VolSize_Unit2:   resd 1
                                <1>
000156CF <res 00000004>         <1> Vol_Tot_Sec_Str_Start:       resd 1
000156D3 <res 0000000A>         <1> Vol_Tot_Sec_Str:       resb 10
000156DD <res 00000001>         <1> Vol_Tot_Sec_Str_End:         resb 1
000156DE <res 00000001>         <1> resb 1
000156DF <res 00000004>         <1> Vol_Free_Sectors_Str_Start: resd 1
000156E3 <res 0000000A>         <1> Vol_Free_Sectors_Str:        resb 10
000156ED <res 00000001>         <1> Vol_Free_Sectors_Str_End:   resb 1
                                <1>
                                <1> ; 10/02/2016
000156EE <res 00000001>         <1> RUN_CDRV: resb 1 ; CMD_INTR.ASM  ; 09/11/2011
                                <1>
                                <1> ; 24/01/2016
000156EF <res 00000080>         <1> PATH_Array:     resb 128 ; DIR.ASM ; 09/10/2011
                                <1> ; 06/02/2016
0001576F <res 00000004>         <1> CCD_DriveDT: resd 1 ; DIR.ASM ; (word)
00015773 <res 00000001>         <1> CCD_Level:   resb 1 ; DIR.ASM
00015774 <res 00000001>         <1> Last_Dir_Level:    resb 1 ; DIR.ASM
                                <1> ;
00015775 <res 00000002>         <1> CDLF_AttributesMask: resw 1 ; DIR.ASM
00015777 <res 00000004>         <1> CDLF_FNAddress:    resd 1 ; DIR.ASM (word)
0001577B <res 00000002>         <1> CDLF_DEType: resw 1 ; DIR.ASM
                                <1> ;
0001577D <res 00000001>         <1> CD_COMMAND:  resb 1 ; DIR.ASM
                                <1>
0001577E <res 00000002>         <1> alignb 4
                                <1>
                                <1> ; 29/01/2016
00015780 <res 00000001>         <1> Program_Exit:      resb 1 ; CMD_INTR.ASM  ; 09/11/2011
                                <1>
                                <1> ;alignb 4
                                <1> ; 23/02/2016
00015781 <res 00000001>         <1> disk_rw_op:  resb 1 ;  0 = disk read, 1 = disk write
                                <1> ;disk_rw_spt:      resb 1 ; sectors per track (<= 63) /// (<256)
                                <1> ; 31/01/2016
00015782 <res 00000001>         <1> retry_count:       resb 1 ; DISK_IO.ASM ; 20/07/2011 (CHS_RetryCount)
00015783 <res 00000001>         <1> disk_rw_err:       resb 1 ; DISK_IO.ASM ; (Disk_IO_err_code)
00015784 <res 00000004>         <1> sector_count:      resd 1 ; DISK_IO.ASM ; (Disk_RW_SectorCount)
                                <1>
                                <1> ; 06/02/2016 (long name)
00015788 <res 00000002>         <1> FDE_AttrMask:         resw 1 ; DIR.ASM
0001578A <res 00000002>         <1> AmbiguousFileName: resw 1 ; DIR.ASM
0001578C <res 00000001>         <1> PreviousAttr:         resb 1 ; DIR.ASM
                                <1> ;
0001578D <res 00000001>         <1> LongNameFound:   resb 1     ; DIR.ASM
0001578E <res 00000001>         <1> LFN_EntryLength: resb 1   ; DIR.ASM
0001578F <res 00000001>         <1> LFN_CheckSum:    resb 1   ; DIR.ASM
00015790 <res 00000084>         <1> LongFileName:    resb 132 ; DIR.ASM
                                <1>
                                <1> ;PATH_Array_Ptr: resw 1 ; DIR.ASM
00015814 <res 00000001>         <1> PATH_CDLevel:       resb 1 ; DIR.ASM
00015815 <res 00000001>         <1> PATH_Level:   resb 1 ; DIR.ASM
                                <1>
                                <1> ; 07/02/2016
00015816 <res 0000000D>         <1> Dir_File_Name:     resb 13 ; DIR.ASM ; 09/10/2011
                                <1>
                                <1> ; 10/02/2016
00015823 <res 0000000D>         <1> Dir_Entry_Name:    resb 13 ; DIR.ASM
```

```
53621                                  <1>
53622                                  <1> alignb 2
53623                                  <1>
53624 00015830 <res 00000002>          <1> AttributesMask: resw 1 ; CMD_INTR.ASM ; 09/11/2011
53625                                  <1>
53626                                  <1> ; 10/02/2016 (128 bytes -> 126 bytes)
53627                                  <1> ; 08/02/2016
53628                                  <1> ;FFF Structure (128 bytes) ; DIR.ASM ; 09/10/2011
53629 00015832 <res 00000001>          <1> FindFile_Drv:              resb 1
53630 00015833 <res 00000041>          <1> FindFile_Directory:        resb 65
53631 00015874 <res 0000000D>          <1> FindFile_Name:             resb 13
53632                                  <1> FindFile_LongNameEntryLength:
53633 00015881 <res 00000001>          <1> FindFile_LongNameYes:      resb 1 ; Sign for longname procedures
53634                                  <1> ;Above 80 bytes form
53635                                  <1> ;TR-DOS Source/Destination File FullName Format/Structure
53636 00015882 <res 00000002>          <1> FindFile_AttributesMask:  resw 1
53637 00015884 <res 00000020>          <1> FindFile_DirEntry:   resb 32
53638 000158A4 <res 00000004>          <1> FindFile_DirFirstCluster: resd 1
53639 000158A8 <res 00000004>          <1> FindFile_DirCluster:       resd 1
53640 000158AC <res 00000002>          <1> FindFile_DirEntryNumber:  resw 1
53641 000158AE <res 00000002>          <1> FindFile_MatchCounter:     resw 1
53642 000158B0 <res 00000002>          <1> FindFile_Reserved:   resw 1 ; 06/03/2016
53643                                  <1>
53644 000158B2 <res 00000004>          <1> First_Path_Pos: resd 1    ; DIR.ASM ; 09/10/2011
53645 000158B6 <res 00000004>          <1> Last_Slash_Pos: resd 1    ; DIR.ASM
53646                                  <1>
53647                                  <1> ; 10/02/2016
53648 000158BA <res 00000002>          <1> File_Count:     resw 1    ; DIR.ASM ; 09/10/2011
53649 000158BC <res 00000002>          <1> Dir_Count:      resw 1
53650 000158BE <res 00000004>          <1> Total_FSize:    resd 1
53651 000158C2 <res 00000004>          <1> TFS_Dec_Begin:  resd 1
53652 000158C6 <res 0000000A>          <1>                 resb 10
53653 000158D0 <res 00000001>          <1> TFS_Dec_End:    resb 1
53654                                  <1>
53655 000158D1 <res 00000001>          <1> PrintDir_RowCounter: resb 1
53656                                  <1>
53657 000158D2 <res 00000002>          <1> alignb 4
53658                                  <1> ; 15/02/2015 ('show' command variables)
53659 000158D4 <res 00000004>          <1> Show_FDT:    resd 1
53660 000158D8 <res 00000004>          <1> Show_LDDDT:  resd 1
53661 000158DC <res 00000004>          <1> Show_Cluster:      resd 1
53662 000158E0 <res 00000004>          <1> Show_FileSize:     resd 1
53663 000158E4 <res 00000004>          <1> Show_FilePointer: resd 1
53664 000158E8 <res 00000002>          <1> Show_ClusterPointer: resw 1
53665 000158EA <res 00000002>          <1> Show_ClusterSize: resw 1
53666 000158EC <res 00000001>          <1> Show_RowCount:     resb 1
53667                                  <1>
53668 000158ED <res 00000003>          <1> alignb 4
53669                                  <1> ; 21/02/2016
53670 000158F0 <res 00000004>          <1> DelFile_FNPointer: resd 1 ; ; CMD_INTR.ASM (word) ; 09/11/2011
53671                                  <1> ; 27/02/2016
53672                                  <1> ; DIR.ASM (09/10/2011)
53673 000158F4 <res 00000004>          <1> DelFile_FCluster:  resd 1
53674 000158F8 <res 00000002>          <1> DelFile_EntryCounter:    resw 1
53675 000158FA <res 00000001>          <1> DelFile_LNEL:            resb 1
53676 000158FB <res 00000001>          <1> resb 1
53677                                  <1>
53678                                  <1> ; DIR.ASM
53679 000158FC <res 00000004>          <1> mkdir_DirName_Offset:    resd 1
53680 00015900 <res 00000004>          <1> mkdir_FFCluster:   resd 1
53681 00015904 <res 00000004>          <1> mkdir_LastDirCluster:    resd 1
53682 00015908 <res 00000004>          <1> mkdir_FreeSectors: resd 1
53683 0001590C <res 00000002>          <1> mkdir_attrib:            resw 1
53684 0001590E <res 00000001>          <1> mkdir_SecPerClust: resb 1
53685 0001590F <res 00000001>          <1> mkdir_add_new_cluster:   resb 1
53686 00015910 <res 0000000D>          <1> mkdir_Name:        resb 13
53687 0001591D <res 00000002>          <1> resw 1 ; 01/03/2016
53688                                  <1> ; 27/02/2016
53689 0001591F <res 00000001>          <1> RmDir_MultiClusters:     resb 1
53690 00015920 <res 00000004>          <1> RmDir_DirEntryOffset:    resd 1 ; 01/03/2016 (word -> dword)
53691 00015924 <res 00000004>          <1> RmDir_ParentDirCluster: resd 1
53692 00015928 <res 00000004>          <1> RmDir_DirLastCluster:   resd 1
53693 0001592C <res 00000004>          <1> RmDir_PreviousCluster:  resd 1
53694                                  <1> ; 22/02/2016
53695 00015930 <res 00000001>          <1> UPDLMDT_CDirLevel: resb 1
53696 00015931 <res 00000004>          <1> UPDLMDT_CDirFCluster:    resd 1
53697                                  <1>
53698 00015935 <res 00000003>          <1> alignb 4
53699                                  <1> ; DRV_FAT.ASM ; 21/08/2011
53700 00015938 <res 00000004>          <1> gffc_next_free_cluster:  resd 1
53701 0001593C <res 00000004>          <1> gffc_first_free_cluster: resd 1
53702 00015940 <res 00000004>          <1> gffc_last_free_cluster:  resd 1
53703                                  <1>
53704                                  <1> ;29/04/2016
53705                                  <1> Cluster_Index: ; resd 1
53706                                  <1> ; 22/02/2016
53707 00015944 <res 00000004>          <1> ClusterValue:      resd 1
53708                                  <1> ; 04/03/2016
53709 00015948 <res 00000001>          <1> Attributes:  resb 1
53710                                  <1> ;;CFS_error:  resb 1 ;; 01/03/2016
53711 00015949 <res 00000001>          <1> resb 1
53712 0001594A <res 00000001>          <1> CFS_OPType: resb 1
53713 0001594B <res 00000001>          <1> CFS_Drv:    resb 1
53714 0001594C <res 00000004>          <1> CFS_CC:         resd 1
53715 00015950 <res 00000004>          <1> CFS_FAT32FSINFOSEC: resd 1
53716 00015954 <res 00000004>          <1> CFS_FAT32FC: resd 1
53717                                  <1>
53718                                  <1> ; 27/02/2016
53719                                  <1> ;alignb 4
53720 00015958 <res 00000004>          <1> glc_prevcluster: resd 1 ; DRV_FAT.ASM (21/08/2011)
53721                                  <1> ; 22/10/2016
53722 0001595C <res 00000004>          <1> glc_index:   resd 1 ;  Last Cluster Index (22/10/2016)
```

```
53723                                   <1>
53724                                   <1> ; DIR.ASM
53725 00015960 <res 00000002>           <1> DLN_EntryNumber: resw 1
53726 00015962 <res 00000001>           <1> DLN_40h:     resb 1
53727                                   <1> ; 28/02/2016
53728 00015963 <res 00000001>           <1> TCC_FATErr:  resb 1 ; DRV_FAT.ASM
53729                                   <1>
53730                                   <1> alignb 4
53731                                   <1> ; DIR.ASM (09/10/2011)
53732 00015964 <res 00000002>           <1> LCDE_EntryIndex: resw 1 ; LCDE_EntryOffset
53733 00015966 <res 00000002>           <1> LCDE_ClusterSN:  resw 1
53734 00015968 <res 00000004>           <1> LCDE_Cluster:       resd 1
53735 0001596C <res 00000004>           <1> LCDE_ByteOffset: resd 1
53736                                   <1>
53737                                   <1> ;alignb4
53738                                   <1> ; 06/03/2016 (word -> dword)
53739                                   <1> ; CMD_INTR.ASM (01/08/2010)
53740 00015970 <res 00000004>           <1> SourceFilePath:         resd 1
53741 00015974 <res 00000004>           <1> DestinationFilePath: resd 1
53742                                   <1>
53743                                   <1> ;alignb 4
53744                                   <1> ; 06/03/2016
53745                                   <1> ; FILE.ASM (09/10/2011)
53746                                   <1> ;Source File Structure (same with 'Find File' Structure)
53747 00015978 <res 00000001>           <1> SourceFile_Drv:                 resb 1
53748 00015979 <res 00000041>           <1> SourceFile_Directory:           resb 65
53749 000159BA <res 0000000D>           <1> SourceFile_Name:         resb 13
53750                                   <1> SourceFile_LongNameEntryLength:
53751 000159C7 <res 00000001>           <1> SourceFile_LongNameYes:         resb 1 ; Sign for longname procedures
53752                                   <1> ;Above 80 bytes
53753                                   <1> ;is TR-DOS Source File FullName Format/Structure
53754 000159C8 <res 00000002>           <1> SourceFile_AttributesMask:      resw 1
53755 000159CA <res 00000020>           <1> SourceFile_DirEntry:            resb 32
53756 000159EA <res 00000004>           <1> SourceFile_DirFirstCluster:     resd 1
53757 000159EE <res 00000004>           <1> SourceFile_DirCluster:          resd 1
53758 000159F2 <res 00000002>           <1> SourceFile_DirEntryNumber:      resw 1
53759 000159F4 <res 00000002>           <1> SourceFile_MatchCounter:  resw 1
53760                                   <1> ; 16/03/2016
53761 000159F6 <res 00000001>           <1> SourceFile_SecPerClust:         resb 1
53762 000159F7 <res 00000001>           <1> SourceFile_Reserved:            resb 1
53763                                   <1> ; Above is 128 bytes
53764                                   <1>
53765                                   <1> ;Destination File Structure (same with 'Find File' Structure)
53766 000159F8 <res 00000001>           <1> DestinationFile_Drv:            resb 1
53767 000159F9 <res 00000041>           <1> DestinationFile_Directory:      resb 65
53768 00015A3A <res 0000000D>           <1> DestinationFile_Name:           resb 13
53769                                   <1> DestinationFile_LongNameEntryLength:
53770 00015A47 <res 00000001>           <1> DestinationFile_LongNameYes:    resb 1 ; Sign for longname procedures
53771                                   <1> ;Above 80 bytes
53772                                   <1> ;is TR-DOS Destination File FullName Format/Structure
53773 00015A48 <res 00000002>           <1> DestinationFile_AttributesMask: resw 1
53774 00015A4A <res 00000020>           <1> DestinationFile_DirEntry: resb 32
53775 00015A6A <res 00000004>           <1> DestinationFile_DirFirstCluster: resd 1
53776 00015A6E <res 00000004>           <1> DestinationFile_DirCluster:     resd 1
53777 00015A72 <res 00000002>           <1> DestinationFile_DirEntryNumber: resw 1
53778 00015A74 <res 00000002>           <1> DestinationFile_MatchCounter:   resw 1
53779                                   <1> ; 16/03/2016
53780 00015A76 <res 00000001>           <1> DestinationFile_SecPerClust:    resb 1
53781 00015A77 <res 00000001>           <1> DestinationFile_Reserved: resb 1
53782                                   <1> ; Above is 128 bytes
53783                                   <1>
53784                                   <1> ; 24/04/2016
53785 00015A78 <res 00000002>           <1> resw 1
53786                                   <1>
53787                                   <1> ; 10/03/2016
53788                                   <1> ; FILE.ASM
53789 00015A7A <res 00000001>           <1> move_cmd_phase:       resb 1
53790 00015A7B <res 00000001>           <1> msftdf_sf_df_drv:  resb 1
53791 00015A7C <res 00000004>           <1> msftdf_drv_offset: resd 1
53792                                   <1>
53793                                   <1> ; 11/03/2016
53794                                   <1> ; DRV_FAT.ASM (21/08/2011)
53795 00015A80 <res 00000004>           <1> FAT_anc_LCluster:  resd 1
53796 00015A84 <res 00000004>           <1> FAT_anc_FFCluster: resd 1
53797                                   <1>
53798                                   <1> ;alignb 4
53799                                   <1>
53800                                   <1> ; 14/03/2016
53801                                   <1> ; TRDOS 386 = TRDOS v2.0 feature only !
53802                                   <1> ; 'allocate_memory_block' in 'memory.s'
53803 00015A88 <res 00000004>           <1> mem_ipg_count:     resd 1 ; page count (for contiguous allocation)
53804 00015A8C <res 00000004>           <1> mem_pg_count:      resd 1 ; page count (for count down)
53805 00015A90 <res 00000004>           <1> mem_aperture:      resd 1 ; contiguous free pages (current)
53806 00015A94 <res 00000004>           <1> mem_max_aperture: resd 1 ; maximum value of contiguous free pages
53807 00015A98 <res 00000004>           <1> mem_pg_pos:  resd 1 ; mem. position (page #) of current aperture
53808 00015A9C <res 00000004>           <1> mem_max_pg_pos: resd 1 ; mem. position (page #) of max. aperture
53809                                   <1>
53810                                   <1> ; 15/03/2016
53811                                   <1> ; FILE.ASM ('copy_source_file_to_destination_file')
53812 00015AA0 <res 00000001>           <1> copy_cmd_phase:       resb 1
53813 00015AA1 <res 00000001>           <1> csftdf_rw_err:           resb 1
53814 00015AA2 <res 00000001>           <1> DestinationFileFound: resb 1
53815 00015AA3 <res 00000001>           <1> csftdf_cdrv:             resb 1
53816 00015AA4 <res 00000004>           <1> csftdf_filesize:      resd 1
53817                                   <1> ; TRDOS386 (TRDOS v2.0)
53818 00015AA8 <res 00000004>           <1> csftdf_sf_mem_addr:   resd 1
53819 00015AAC <res 00000004>           <1> csftdf_sf_mem_bsize:  resd 1
53820                                   <1> ;
53821                                   <1>
53822 00015AB0 <res 00000004>           <1> csftdf_sf_cluster:    resd 1 ; 16/03/2016
53823 00015AB4 <res 00000004>           <1> csftdf_df_cluster:    resd 1
53824                                   <1> ; 16/03/2016
```

```
53825 00015AB8 <res 00000004>      <1> csftdf_r_size:        resd 1
53826 00015ABC <res 00000004>      <1> csftdf_w_size:        resd 1
53827 00015AC0 <res 00000004>      <1> csftdf_sf_rbytes:     resd 1
53828 00015AC4 <res 00000004>      <1> csftdf_df_wbytes:     resd 1
53829 00015AC8 <res 00000001>      <1> csftdf_percentage:    resb 1
53830                              <1> ; 17/03/2016
53831 00015AC9 <res 00000001>      <1> csftdf_videopage:     resb 1
53832 00015ACA <res 00000002>      <1> csftdf_cursorpos:     resw 1
53833 00015ACC <res 00000004>      <1> csftdf_sf_drv_dt:     resd 1
53834 00015AD0 <res 00000004>      <1> csftdf_df_drv_dt:     resd 1
53835                              <1>
53836                              <1> ; 21/03/2016
53837                              <1> ; 20/03/2016
53838                              <1> ; FILE.ASM
53839 00015AD4 <res 00000004>      <1> createfile_Name_Offset:  resd 1
53840 00015AD8 <res 00000004>      <1> createfile_FreeSectors:  resd 1
53841 00015ADC <res 00000004>      <1> createfile_size:         resd 1
53842 00015AE0 <res 00000004>      <1> createfile_FFCluster:    resd 1 ; 11/03/2016
53843 00015AE4 <res 00000004>      <1> createfile_LastDirCluster: resd 1
53844 00015AE8 <res 00000004>      <1> createfile_Cluster:      resd 1
53845 00015AEC <res 00000004>      <1> createfile_PCluster:     resd 1
53846 00015AF0 <res 00000001>      <1> createfile_attrib:  resb 1
53847 00015AF1 <res 00000001>      <1> createfile_SecPerClust:  resb 1
53848 00015AF2 <res 00000002>      <1> createfile_DirIndex:     resw 1
53849 00015AF4 <res 00000004>      <1> createfile_CCount:  resd 1
53850 00015AF8 <res 00000002>      <1> createfile_BytesPerSec:   resw 1 ; 23/03/2016
53851 00015AFA <res 00000001>      <1> createfile_wfc:             resb 1
53852 00015AFB <res 00000001>      <1> createfile_UpdatePDir:    resb 1 ; 31/03/2016
53853                              <1>
53854                              <1> ;alignb 4
53855                              <1>
53856                              <1> ; 11/04/2016
53857 00015AFC <res 00000002>      <1> env_var_length:       resw 1
53858                              <1>
53859 00015AFE <res 00000002>      <1> alignb 4
53860                              <1>
53861                              <1> ; 25/04/2016
53862 00015B00 <res 00000001>      <1> readi.valid: resb 1 ; valid data (>0 = valid for readi)
53863 00015B01 <res 00000001>      <1> readi.drv:   resb 1 ; drive number (0, 1,2,3,4..)
53864 00015B02 <res 00000001>      <1> readi.spc:   resb 1 ; sectors per cluster for 'readi' drive
53865 00015B03 <res 00000001>      <1> readi.s_index:  resb 1 ; sector index in current cluster (buffer)
53866 00015B04 <res 00000004>      <1> readi.sector:      resd 1 ; current disk sector
53867 00015B08 <res 00000002>      <1> readi.bpc:   resw 1 ; bytes per cluster - 1
53868 00015B0A <res 00000002>      <1> readi.offset:      resw 1 ; byte offset in cluster buffer
53869 00015B0C <res 00000004>      <1> readi.cluster:  resd 1 ; current cluster number
53870 00015B10 <res 00000004>      <1> readi.c_index:     resd 1 ; cluster index of the current cluster (0,1,2,3..)
53871 00015B14 <res 00000004>      <1> readi.fclust:      resd 1 ; first cluster of the current cluster
53872 00015B18 <res 00000004>      <1> readi.fs_index: resd 1 ; sector index in disk/file section (for Singlix FS)
53873                              <1> ;readi.buffer:     resd 1 ; readi sector buffer address
53874                              <1>
53875                              <1> ;alignb 4
53876                              <1>
53877 00015B1C <res 00000001>      <1> writei.valid:      resb 1 ; valid data (>0 = valid for writei)
53878 00015B1D <res 00000001>      <1> writei.drv:  resb 1 ; drive number (0, 1,2,3,4..)
53879 00015B1E <res 00000001>      <1> writei.spc:  resb 1 ; sectors per cluster for 'writei' drive
53880 00015B1F <res 00000001>      <1> writei.s_index: resb 1 ; sector index in current cluster (buffer)
53881 00015B20 <res 00000004>      <1> writei.sector:     resd 1 ; current disk sector
53882 00015B24 <res 00000002>      <1> writei.bpc:  resw 1 ; bytes per cluster - 1
53883 00015B26 <res 00000002>      <1> writei.offset:     resw 1 ; byte offset in cluster buffer
53884 00015B28 <res 00000004>      <1> writei.cluster: resd 1 ; current cluster number
53885 00015B2C <res 00000004>      <1> writei.c_index:    resd 1 ; cluster index of the current cluster (0,1,2,3..)
53886 00015B30 <res 00000004>      <1> writei.fclust:  resd 1 ; first cluster of the current cluster
53887 00015B34 <res 00000004>      <1> writei.fs_index: resd 1 ; sector index in disk/file section (for Singlix FS)
53888                              <1> ;writei.buffer:    resd 1 ; writei sector buffer address
53889 00015B38 <res 00000004>      <1> writei.lclust:     resd 1 ; writei last cluster (mget_w) ; 23/10/2016
53890 00015B3C <res 00000004>      <1> writei.l_index:    resd 1 ; writei last cluster index (mget_w) ; 23/10/2016
53891 00015B40 <res 00000001>      <1> writei.ofn:  resb 1 ; open file number (to be written) ; 23/10/2016
53892                              <1>
53893 00015B41 <res 00000003>      <1> alignb 4
53894                              <1>
53895                              <1> ; 29/04/2016
53896 00015B44 <res 00000004>      <1> Run_CDirFC:  resd 1
53897 00015B48 <res 00000001>      <1> Run_Auto_Path:     resb 1
53898 00015B49 <res 00000001>      <1> Run_Manual_Path: resb 1 ; 0 -> auto path sequence needed
53899 00015B4A <res 00000001>      <1> EXE_ID:            resb 1
53900 00015B4B <res 00000001>      <1> EXE_dot:    resb 1
53901                              <1>
53902                              <1> ; 06/05/2016
53903 00015B4C <res 00000004>      <1> mainprog_return_addr: resd 1
53904 00015B50 <res 00000004>      <1> last_error:  resd 1  ; this will be used to return error code to MainProg
53905                              <1>                      ; 'lasterror' keyword will be used later to get the
53906                              <1>                      ; last error code/number/status.
53907                              <1> ; 12/05/2016
53908 00015B54 <res 00000004>      <1> video_eax:   resd 1  ; eax return value of video function
53909                              <1>
53910                              <1> ; 01/06/2016
53911 00015B58 <res 00000004>      <1> user_buffer: resd 1  ; 'diskio.s' (INT 33h, Function 08h, floppy disk type)
53912                              <1>
53913                              <1> ; 21/05/2016 - TRDOS 386 ('swap/switch', 'rswap', [u.pri])
53914 00015B5C <res 00000001>      <1> priority:    resb 1  ; running priority level of process (0,1,2)
53915                              <1>                      ; (run queue which is process comes from)
53916                              <1> ; 22/05/2016 - TRDOS 386 ('set_run_sequence', 'rtc_int', 'u_timer')
53917 00015B5D <res 00000001>      <1> p_change:    resb 1  ; process change status (for timer events)
53918                              <1> ; 23/05/2016 - TRDOS 386 ('clock')
53919 00015B5E <res 00000001>      <1> multi_tasking:     resb 1   ; Multi Tasking status (0 = disabled, >0 = enabled)
53920                              <1>                      ; (EBX will return with user buffer addr or disk type)
53921                              <1> ; 07/06/2016
53922 00015B5F <res 00000001>      <1> timer_events:      resb 1  ; number of (active) timer events, <= 16
53923                              <1>
53924                              <1> ; 24/06/2016
53925 00015B60 <res 00000001>      <1> w_str_cmd:   resb 1 ; WRITE_STRING command (0,1,2,3) ; video.s
53926 00015B61 <res 00000001>      <1> p_crt_mode:  resb 1  ; previous video mode (=3 or 0), backup mark/sign
```

```
                                <1> ; 26/06/2016
00015B62 <res 00000001>         <1> p_crt_page:  resb 1  ; previous active page (for 'set_mode')
                                <1> ; 04/07/2016
00015B63 <res 00000001>         <1> noclearmem:  resb 1  ; if set, 'SET MODE' (INT 31h) function (AH = 4)
                                <1>                      ; will not clear the video memory
                                <1>                      ; (usable for graphics modes only)
                                <1> alignb 2
00015B64 <res 00000002>         <1> CRT_LEN:     resw 1  ; length of regen buffer in bytes
00015B66 <res 00000010>         <1> cursor_pposn:      resw 8  ; cursor positions backup
                                <1>
                                <1> ; 10/07/2016 ('VGA_FONT_SETUP', INT 43H address for x86 real mode bios)
00015B76 <res 00000004>         <1> VGA_INT43H:  resd 1 ; 0 = default (not configured by user)
                                <1>                      ; 0FFFFFFFFh = user defined fonts
                                <1>                      ; address:
                                <1>                      ;     vgafont8
                                <1>                      ;     vgafont16
                                <1>                      ;     vgafont14
                                <1>
                                <1> ; 25/07/2016
00015B7A <res 00000001>         <1> VGA_MTYPE:   resb 1  ; 0=CTEXT,1=MTEXT,2=CGA,3=PLANAR1,4=PLANAR4,5=LINEAR
                                <1>
                                <1> ; 23/10/2016
00015B7B <res 00000001>         <1> setfmod            resb 1 ; update last modification date&time sign (if >0)
                                <1>                      ; (it is Open File Number + 1, if > 0)
                                <1> alignb 4
                                <1>
                                <1> ; 16/10/2016
00015B7C <res 00000004>         <1> FFF_UBuffer: resd 1  ; User's buffer address for FFF & FNF system calls
                                <1> ; 15/10/2016
00015B80 <res 00000001>         <1> FFF_Valid:   resb 1  ; Find First File Structure validation byte
                                <1>                      ; 0  = invalid (Find Next File can't use FFF struct)
                                <1>                      ; >0 = valid, return type for FFF and Find Next File
                                <1>                      ; 24 = basic parameters, 24 bytes
                                <1>                      ; 128 = entire FFF structure/table, 128 bytes
                                <1> ; 16/10/2016 (FFF_Attrib: resw 1)
00015B81 <res 00000001>         <1> FFF_Attrib:  resb 1 ; Find First File attributes for Find Next File (LB)
00015B82 <res 00000001>         <1> FFF_RType:   resb 1  ; FFF return type (0 = Basic, >0 = complete) (HB)
                                <1> ; 16/10/2016 - 05/10/2016 (Set Working Path)
00015B83 <res 00000001>         <1> SWP_inv_fname:     resb 1 ; Set Working Path - Invalid File Name
00015B84 <res 00000002>         <1> SWP_Mode:    resw 1 ; Set Working Path - Mode
00015B86 <res 00000001>         <1> SWP_DRV:     resb 1 ; Set Working Path - Drive
00015B87 <res 00000001>         <1> SWP_DRV_chg: resb 1 ; Set Working Path - Drive Change
                                <1>
                                <1> ; 27/02/2017
00015B88 <res 00000001>         <1> fpready:     resb 1 ; '80387 fpu is ready' flag
                                <1>
                                <1> ; 08/10/2016
00015B89 <res 00000009>         <1> device_name:    resb 9  ; capitalized (and zero padded) device canem
                                <1>                      ; (example: "TTY0",0,0,0,0,0")
                                <1>
00015B92 <res 00000002>         <1> alignb 4
                                <1>
                                <1> ; 08/10/2016
                                <1> ; 07/10/2016
                                <1> ; Table of kernel devices (which do not use installable device drivers)
                                <1> ; has been coded into KERNEL (trdosk9.s)
                                <1> ; 07/10/2016
                                <1> ; 8 installable device drivers available to install (NUMIDEV)
00015B94 <res 00000020>         <1> IDEV_PGDIR: resd NUMIDEV
                                <1>                      ; Page directories of installable device drivers
                                <1>                      ;
                                <1>                      ; Note: Virtual start address is always 400000h
                                <1>                      ; (end of the 1st 4MB). [org 400000h]
                                <1>                      ; Segments: KCODE, KDATA
                                <1>                      ; Method: call 400000h (after changing page dir)
                                <1>                      ; Query code located at the start (400000h).
                                <1>                      ; Query code returns with
                                <1>                      ;   eax = device type and driver version
                                <1>                      ;         AL = Device Type minor
                                <1>                      ;         AH = Device Type major
                                <1>                      ;         Byte 16-23 : Version minor
                                <1>                      ;         Byte 24-31 : Version major - 1
                                <1>                      ;                      (0:0 -> 1.0)
                                <1>                      ;   ebx = initialization code address
                                <1>                      ;   ecx = configuration table address
                                <1>                      ;   edx = description table address
                                <1>                      ;   esi = device (default) name address (ASCIIZ)
                                <1>                      ;       (name has "/DEV/" prefix)
                                <1>                      ;   edi = dispatch table address
                                <1>                      ;         (for calling kernel-device functions)
                                <1>                      ;   ebp = address table address
                                <1>                      ; Initialization code returns with
                                <1>                      ;   eax = open code address
                                <1>                      ;   ecx = close code address
                                <1>                      ;   ebx = read code address
                                <1>                      ;   edx = write code address
                                <1>                      ;   esi = IOCTL code address
                                <1>                      ;   edi = dispatch table address
                                <1>                      ;   ebp = address table address
                                <1>                      ; Address Table:
                                <1>                      ;    Offset 0  : open code address
                                <1>                      ;    Offset 4  : read code address
                                <1>                      ;    Offset 8  : write code address
                                <1>                      ;    Offset 12 : close code address
                                <1>                      ;    Offset 16 : IOCTL code address
                                <1>                      ;    Offset 20 : initialization code address
                                <1>                      ;    Offset 24 : description table address
                                <1>                      ;    Offset 28 : configuration table address
                                <1>                      ;    Offset 32 : device name address
                                <1>                      ;    Offset 36 : dispatch table address
                                <1>                      ;          (for calling kernel-device functions)
                                <1>
```

```
54029 00015BB4 <res 00000040>      <1> IDEV_NAME:  resb 8*NUMIDEV
54030                              <1>                   ; 8 byte names of installable device drivers
54031                              <1>
54032 00015BF4 <res 00000008>      <1> IDEV_TYPE:  resb NUMIDEV ; Driver type of installable device drivers
54033 00015BFC <res 00000008>      <1> IDEV_FLAGS: resb NUMIDEV ; Device access parameters for installable
54034                              <1>                          ; device drivers (These values are set while
54035                              <1>                    ; the device driver is being loaded.)
54036 00015C04 <res 00000020>      <1> IDEV_OADDR: resd NUMIDEV ; open function addr for installable dev driver
54037 00015C24 <res 00000020>      <1> IDEV_CADDR: resd NUMIDEV ; close function addr for installable dev driver
54038 00015C44 <res 00000020>      <1> IDEV_RADDR: resd NUMIDEV ; read function addr for installable dev driver
54039 00015C64 <res 00000020>      <1> IDEV_WADDR: resd NUMIDEV ; write function addr for installable dev driver
54040                              <1>
54041                              <1> ; 08/10/2016
54042                              <1> ; 07/10/2016
54043                              <1> ; Device Open and Access parameters
54044 00015C84 <res 0000001E>      <1> DEV_ACCESS:   resb NUMOFDEVICES   ; bit 0 = accessable by normal users
54045                              <1>                                   ; bit 1 = read access permission
54046                              <1>                                   ; bit 2 = write access permission
54047                              <1>                                   ; bit 3 = IOCTL permission to users
54048                              <1>                                   ; bit 4 = block device if it is set
54049                              <1>                                   ; bit 5 = 16 bit or 1024 byte data
54050                              <1>                                   ; bit 6 = 32 bit or 2048 byte data
54051                              <1>                                   ; bit 7 = installable device driver
54052 00015CA2 <res 0000001E>      <1> DEV_R_OWNER:  resb NUMOFDEVICES   ; Reading owner no (u.uid) of devices
54053 00015CC0 <res 0000001E>      <1> DEV_R_OPENCOUNT: resb NUMOFDEVICES   ; Reading open count
54054 00015CDE <res 0000001E>      <1> DEV_W_OWNER:  resb NUMOFDEVICES   ; Writing owner no (u.uid) of devices
54055 00015CFC <res 0000001E>      <1> DEV_W_OPENCOUNT: resb NUMOFDEVICES   ; Writing open count
54056 00015D1A <res 0000001E>      <1> DEV_DRIVER:   resb NUMOFDEVICES   ; device driver number (1 to 7Fh)
54057                              <1>                                   ; *if bit 7 is set (80 to FFh)
54058                              <1>                                   ; *if it is installable device driver
54059                              <1>                                   ; *index (0 to 7Fh)
54060                              <1>                                   ; otherwise it is kernel device index
54061 00015D38 <res 0000001E>      <1> DEV_OPENMODE:       resb NUMOFDEVICES   ; 1 = read mode
54062                              <1>                                   ; 2 = write mode
54063                              <1>                                   ; 3 = read & write
54064                              <1>                                   ; 0 = not open (free)
54065 00015D56 <res 00000078>      <1> DEV_NAME_PTR:    resd NUMOFDEVICES   ; pointers to name addresses of drivers
54066                              <1>                                   ; Address base: KDEV_NAME+
54067                              <1>                                   ; or IDEV_NAME+
54068 00015DCE <res 00000078>      <1> DEV_R_POINTER:      resd NUMOFDEVICES   ; reading pointer, writing pointer
54069 00015E46 <res 00000078>      <1> DEV_W_POINTER:      resd NUMOFDEVICES   ; sector number if block device
54070                              <1>                                   ; character offset if char device
54071 00015EBE <res 00000002>      <1> alignb 4
54072                              <1>
54073                              <1> ; 06/10/2016
54074                              <1> ; Open File Parameters
54075 00015EC0 <res 00000028>      <1> OF_FCLUSTER: resd OPENFILES  ; First clusters of open files
54076 00015EE8 <res 0000000A>      <1> OF_DRIVE:    resb OPENFILES  ; Logical DOS drive numbers of open files
54077 00015EF2 <res 0000000A>      <1> OF_MODE:     resb OPENFILES  ; Open mode (1 = read, 2 = write, 3 = r&w)
54078 00015EFC <res 0000000A>      <1> OF_STATUS:   resb OPENFILES  ; (bit 0 = read, bit 1 = write)
54079 00015F06 <res 0000000A>      <1> OF_OPENCOUNT:      resb OPENFILES  ; Open counts of open files
54080 00015F10 <res 00000028>      <1> OF_POINTER:  resd OPENFILES      ; File seek/read/write pointer
54081 00015F38 <res 00000028>      <1> OF_SIZE:     resd OPENFILES      ; File sizes of open files (in bytes)
54082 00015F60 <res 00000028>      <1> OF_DIRFCLUSTER:    resd OPENFILES  ; Directory First Clusters of open files
54083 00015F88 <res 00000028>      <1> OF_DIRCLUSTER:     resd OPENFILES  ; Directory (Entry) Clusters of open files
54084 00015FB0 <res 00000028>      <1> OF_VOLUMEID: resd OPENFILES  ; Vol ID for removable drives of open files
54085 00015FD8 <res 00000028>      <1> OF_CCLUSTER: resd OPENFILES  ; Current clusters of open files
54086 00016000 <res 00000028>      <1> OF_CCINDEX:  resd OPENFILES  ; Cluster index numbers of current clusters
54087                              <1> ; 24/10/2016
54088 00016028 <res 00000014>      <1> OF_DIRENTRY: resw OPENFILES  ; Directory entry index no. in dir cluster
54089                              <1>                              ; Sector index = entry index / 16
54090                              <1> ;alignb 2
54091                              <1>
54092 0001603C <res 00000060>      <1> DTA:         resd 24            ; Find First File data transfer area
54093                              <1>
54094                              <1> ; 19/12/2016
54095 0001609C <res 00000001>      <1> tcallback:   resb 1       ; Timer callback method flag for 'systimer'
54096 0001609D <res 00000001>      <1> trtc:        resb 1       ; Timer interrupt type flag for 'systimer'
54097                              <1> ; 20/02/2017
54098 0001609E <res 00000001>      <1> no_page_swap:      resb 1       ; Swap lock for Signal Response Byte pages
54099                              <1> ;;15/01/2017
54100                              <1> ; 02/01/2017
54101                              <1> ;;intflg:    resb 1       ; software interrupt in progress signal
54102                              <1>                           ; (for timer interrupt)
54103                              <1>
54104 0001609F <res 00000001>      <1> alignb 4
54105                              <1> ; 13/04/2017
54106 000160A0 <res 0000001E>      <1> DEV_INTR:    resb NUMOFDEVICES ; Device Interrupt (IRQ) number + 1
54107                              <1>                           ; (0= not available, 1= IRQ 0, 16= IRQ 15)
54108 000160BE <res 00000040>      <1> DEV_INT_HNDLR:     resd 16            ; Device Interrupt Handler addr, if > 0
54109                              <1>
54110                              <1>
54111                              <1> ;alignb 4
54112                              <1>
54113                              <1> ; 26/02/2017 ; IRQ Callback parameters ('syscalbac')
54114                              <1> ;Index: ; 0 to 8
54115                              <1> ;     0 = IRQ3, 1 = IRQ4, 2 = IRQ5, 3 = IRQ7
54116                              <1> ;     4 = IRQ9, 5 = IRQ10, 6 = IRQ11, 7 = IRQ12, 8 = IRQ13
54117 000160FE <res 00000009>      <1> IRQ.owner:   resb 9       ; owner, 0 = free, >0 = [u.uno]
54118 00016107 <res 00000009>      <1> IRQ.dev:     resb 9       ; 0 = default/kernel, >0 = device number
54119 00016110 <res 00000009>      <1> IRQ.method:  resb 9             ; 0 = Signal Response Byte, 1 = Callback
54120 00016119 <res 00000009>      <1> IRQ.srb:     resb 9             ; Signal Response/Return Byte value
54121 00016122 <res 00000024>      <1> IRQ.addr:    resd 9       ; Rignal Response Byte address (physical)
54122                              <1>                           ; or Callback service address (virtual)
54123                              <1> ; 28/02/2017
54124 00016146 <res 00000004>      <1> IRQ_cr3:     resd 1       ; for saving cr3 register in IRQ handler
54125 0001614A <res 00000001>      <1> IRQnum:            resb 1       ; IRQ number for IRQ handler (trdosk8.s)
54126                              <1>
54127                              <1>
54128                              <1> ; 10/04/2017
54129                              <1> ; 03/04/2017
54130                              <1> ; UNINITIALIZED AUDIO DATA
```

```
54131 0001614B <res 00000001>      <1> alignb 4
54132 0001614C <res 00000001>      <1> audio_pci:   resb 1
54133 0001614D <res 00000001>      <1> audio_device:      resb 1
54134 0001614E <res 00000001>      <1> audio_mode:  resb 1
54135 0001614F <res 00000001>      <1> audio_intr:  resb 1
54136 00016150 <res 00000001>      <1> audio_busy:  resb 1  ; Busy flag for audio irq ; 21/04/2017
54137 00016151 <res 00000001>      <1> audio_reserved: resb 1
54138 00016152 <res 00000002>      <1> audio_io_base:     resw 1       ; Base I/O address of audio device
54139 00016154 <res 00000004>      <1> audio_dev_id:      resd 1 ; BUS/DEV/FN ; 00000000BBBBBBBBDDDDDFFF00000000
54140 00016158 <res 00000004>      <1> audio_vendor:      resd 1
54141 0001615C <res 00000004>      <1> audio_stats_cmd: resd 1
54142                              <1> ;
54143 00016160 <res 00000004>      <1> audio_buffer:      resd 1 ; virtual address of user's audio buffer
54144 00016164 <res 00000004>      <1> audio_p_buffer:    resd 1 ; Physical address of user's audio buffer
54145 00016168 <res 00000004>      <1> audio_buff_size: resd 1 ; user's audio buffer size (half buffer size)
54146 0001616C <res 00000004>      <1> audio_dma_buff: resd 1  ; dma buffer address
54147 00016170 <res 00000004>      <1> audio_dmabuff_size: resd 1 ; dma buffer size (2 * half buffer size)
54148 00016174 <res 00000001>      <1> audio_flag:  resb 1  ; dma buffer flag (1st half = 0, 2nd half = 1)
54149 00016175 <res 00000001>      <1> audio_user:  resb 1 ; user number of the owner
54150 00016176 <res 00000001>      <1> audio_cb_mode:     resb 1 ; 0 = signal response byte method
54151                              <1>                    ; 1 = callback method
54152                              <1>                    ; 2 = s.r.b. method with auto increment
54153 00016177 <res 00000001>      <1> audio_srb:   resb 1 ; signal response byte value
54154 00016178 <res 00000004>      <1> audio_cb_addr:     resd 1  ; callback service address or s.r.b. address
54155                              <1>                    ; (s.r.b. addr is physical, cbs addr is virtual)
54156                              <1>
54157 0001617C <res 00000001>      <1> audio_bps:   resb 1  ; selected mode: 8 bit, 16 bit
54158 0001617D <res 00000001>      <1> audio_stmo:  resb 1 ; selected mode: mono /stereo
54159 0001617E <res 00000002>      <1> audio_freq:  resw 1 ; sampling rate
54160                              <1>
54161                              <1>
54162                              <1> ; 21/04/2017
54163 00016180 <res 00000001>      <1> audio_play_cmd: resb 1  ; Play/Stop command (1 = play, 0 = stop)
54164                              <1> audio_civ: ; 28/05/2017 ; Current Buffer Index (AC'97)
54165 00016181 <res 00000001>      <1> audio_flag_eol:    resb 1  ; End of Link status (vt8233, EOL/FLAG)
54166                              <1>
54167                              <1> audio_master_volume:
54168 00016182 <res 00000001>      <1> audio_master_volume_l: resb 1 ; sound volume (lineout) left channel
54169 00016183 <res 00000001>      <1> audio_master_volume_r: resb 1 ; sound volume (lineout) right channel
54170                              <1>
54171                              <1> alignb 4
54172                              <1> ; 28/05/2017
54173                              <1> ; AC'97 Audio Controller Base Adress Registers
54174 00016184 <res 00000002>      <1> NAMBAR:            resw 1 ; Native Audio Mixer Base Address
54175 00016186 <res 00000002>      <1> NABMBAR:     resw 1 ; Native Audio Bus Mastering Base Address
54176                              <1>
54177                              <1> ;alignb 4
54178                              <1> ; 21/04/2017
54179 00016188 <res 00000400>      <1> audio_bdl_buff:    resd 32*8  ; VT8233 (AC97) BDL Buffer Size
54180                              <1> ; 12/05/2017
54181 00016588 <res 00000004>      <1> base_addr:   resd 1 ; 'direct_memory_access' (memory.s)
54182                              <1>
54183 0001658C <res 00009A74>      <1> alignb 65536
54184                              <1> ; 12/05/2017
54185 00020000 <res 00040000>      <1> sb16_dma_buffer: resd 65536 ; DMA buffer for sb16 audio playing.
54186                                  ; 24/01/2016
54187                                  %include 'ubss.s'  ; UNINITIALIZED KERNEL (USER) DATA
54188                              <1> ; ****************************************************************************
54189                              <1> ; TRDOS386.ASM (TRDOS 386 Kernel - v2.0.0) - UNINITIALIZED USER DATA : ubss.s
54190                              <1> ; ----------------------------------------------------------------------------
54191                              <1> ; Last Update: 28/02/2017
54192                              <1> ; ----------------------------------------------------------------------------
54193                              <1> ; Beginning: 24/01/2016
54194                              <1> ; ----------------------------------------------------------------------------
54195                              <1> ; Assembler: NASM version 2.11 (trdos386.s)
54196                              <1> ; ----------------------------------------------------------------------------
54197                              <1> ; Derived from 'Retro UNIX 386 Kernel - v0.2.1.0' source code by Erdogan Tan
54198                              <1> ; ux.s (04/12/2015)
54199                              <1> ; ****************************************************************************
54200                              <1>
54201                              <1> ; Retro UNIX 386 v1 Kernel - ux.s
54202                              <1> ; Last Modification: 04/12/2015
54203                              <1> ;
54204                              <1> ; ///////// RETRO UNIX 386 V1 SYSTEM DEFINITIONS //////////////
54205                              <1> ; (Modified from
54206                              <1> ;     Retro UNIX 8086 v1 system definitions in 'UNIX.ASM', 01/09/2014)
54207                              <1> ; ((UNIX.ASM (RETRO UNIX 8086 V1 Kernel), 11/03/2013 - 01/09/2014))
54208                              <1> ; ----------------------------------------------------------------------------
54209                              <1> ; Derived from UNIX Operating System (v1.0 for PDP-11)
54210                              <1> ; (Original) Source Code by Ken Thompson (1971-1972)
54211                              <1> ; <Bell Laboratories (17/3/1972)>
54212                              <1> ; <Preliminary Release of UNIX Implementation Document>
54213                              <1> ; (Section E10 (17/3/1972) - ux.s)
54214                              <1> ; ****************************************************************************
54215                              <1>
54216                              <1> alignb 2
54217                              <1>
54218                              <1> inode:
54219                              <1>       ; 11/03/2013.
54220                              <1>       ;Derived from UNIX v1 source code 'inode' structure (ux).
54221                              <1>       ;i.
54222                              <1>
54223 00060000 <res 00000002>      <1>       i.flgs:       resw 1
54224 00060002 <res 00000001>      <1>       i.nlks:       resb 1
54225 00060003 <res 00000001>      <1>       i.uid:  resb 1
54226                              <1>         ;i.size:  resw 1 ; size
54227 00060004 <res 00000002>      <1>       resw 1 ; 29/04/2016
54228 00060006 <res 00000010>      <1>       i.dskp:       resw 8 ; 16 bytes
54229 00060016 <res 00000004>      <1>       i.ctim:       resd 1
54230 0006001A <res 00000004>      <1>       i.mtim:       resd 1
54231 0006001E <res 00000002>      <1>       i.rsvd:  resw 1 ; Reserved (ZERO/Undefined word for UNIX v1.)
54232                              <1>
```

```
54233                                  <1> I_SIZE      equ $ - inode
54234                                  <1>
54235                                  <1> process:
54236                                  <1>       ; 19/12/2016
54237                                  <1>       ; 21/05/2016
54238                                  <1>       ; 19/05/2016 - TRDOS 386 (TRDOS v2.0)
54239                                  <1>       ; 06/05/2015 - Retro UNIX 386 v1
54240                                  <1>       ; 11/03/2013 - 05/02/2014 (Retro UNIX 8086 v1)
54241                                  <1>       ;Derived from UNIX v1 source code 'proc' structure (ux).
54242                                  <1>       ;p.
54243                                  <1>
54244 00060020 <res 00000020>          <1>         p.pid:   resw nproc
54245 00060040 <res 00000020>          <1>         p.ppid:  resw nproc
54246 00060060 <res 00000020>          <1>         p.break: resw nproc
54247 00060080 <res 00000010>          <1>         p.ttyc:  resb nproc ; console tty in Retro UNIX 8086 v1.
54248 00060090 <res 00000010>          <1>       p.waitc: resb nproc ; waiting channel in Retro UNIX 8086 v1.
54249 000600A0 <res 00000010>          <1>       p.link:       resb nproc
54250 000600B0 <res 00000010>          <1>       p.stat:       resb nproc
54251                                  <1>
54252                                  <1>       ; 06/05/2015 (Retro UNIX 386 v1 feature only !)
54253 000600C0 <res 00000040>          <1>       p.upage: resd nproc ; Physical address of the process's
54254                                  <1>                           ; 'user' structure
54255                                  <1>       ; 21/05/2016
54256                                  <1>       ; 19/05/2016 (TRDOS 386 feature only!)
54257 00060100 <res 00000010>          <1>       p.timer: resb nproc ; number of timer events of the processs
54258                                  <1>
54259                                  <1>       ; 19/12/2016
54260 00060110 <res 00000040>          <1>       p.tcb: resd nproc ; timer callback service address (if > 0)
54261                                  <1>
54262                                  <1> P_SIZE      equ $ - process
54263                                  <1>
54264                                  <1> ; fsp table (original UNIX v1)
54265                                  <1> ;
54266                                  <1> ;Entry
54267                                  <1> ;          15                                      0
54268                                  <1> ;  1     |---|--------------------------------------|
54269                                  <1> ;        |r/w|       i-number of open file          |
54270                                  <1> ;        |---|--------------------------------------|
54271                                  <1> ;        |               device number              |
54272                                  <1> ;        |------------------------------------------|
54273                                  <1> ;    (*) | offset pointer, i.e., r/w pointer to file |
54274                                  <1> ;        |------------------------------------------|
54275                                  <1> ;        |  flag that says    | number of processes  |
54276                                  <1> ;        |   file deleted     | that have file open  |
54277                                  <1> ;        |------------------------------------------|
54278                                  <1> ;  2     |                                          |
54279                                  <1> ;        |------------------------------------------|
54280                                  <1> ;        |                                          |
54281                                  <1> ;        |------------------------------------------|
54282                                  <1> ;        |                                          |
54283                                  <1> ;        |------------------------------------------|
54284                                  <1> ;        |                                          |
54285                                  <1> ;        |------------------------------------------|
54286                                  <1> ;  3     |                                          |
54287                                  <1> ;        |                                          |
54288                                  <1> ;
54289                                  <1> ; (*) Retro UNIX 386 v1 modification: 32 bit offset pointer
54290                                  <1>
54291                                  <1>
54292                                  <1> ; 15/04/2015
54293 00060150 <res 000001F4>          <1> fsp:   resb nfiles * 10 ; 11/05/2015 (8 -> 10)
54294 00060344 <res 00000002>          <1> idev:  resw 1 ; device number is 1 byte in Retro UNIX 8086 v1 !
54295 00060346 <res 00000002>          <1> cdev:    resw 1 ; device number is 1 byte in Retro UNIX 8086 v1 !
54296                                  <1> ; 18/05/2015
54297                                  <1> ; 26/04/2013 device/drive parameters (Retro UNIX 8086 v1 feature only!)
54298                                  <1> ; 'UNIX' device numbers (as in 'cdev' and 'u.cdrv')
54299                                  <1> ;     0 -> root device (which has Retro UNIX 8086 v1 file system)
54300                                  <1> ;     1 -> mounted device (which has Retro UNIX 8086 v1 file system)
54301                                  <1> ; 'Retro UNIX 8086 v1' device numbers: (for disk I/O procedures)
54302                                  <1> ;     0 -> fd0 (physical drive, floppy disk 1), physical drive number = 0
54303                                  <1> ;     1 -> fd1 (physical drive, floppy disk 2), physical drive number = 1
54304                                  <1> ;     2 -> hd0 (physical drive, hard disk 1), physical drive number = 80h
54305                                  <1> ;     3 -> hd1 (physical drive, hard disk 2), physical drive number = 81h
54306                                  <1> ;     4 -> hd2 (physical drive, hard disk 3), physical drive number = 82h
54307                                  <1> ;     5 -> hd3 (physical drive, hard disk 4), physical drive number = 83h
54308 00060348 <res 00000001>          <1> rdev:  resb 1 ; root device number ; Retro UNIX 8086 v1 feature only!
54309                                  <1>               ; as above, for physical drives numbers in following table
54310 00060349 <res 00000001>          <1> mdev:  resb 1 ; mounted device number ; Retro UNIX 8086 v1 feature only!
54311                                  <1> ; 15/04/2015
54312 0006034A <res 00000001>          <1> active:      resb 1
54313 0006034B <res 00000001>          <1>        resb 1 ; 09/06/2015
54314 0006034C <res 00000002>          <1> mnti:  resw 1
54315 0006034E <res 00000002>          <1> mpid:  resw 1
54316 00060350 <res 00000002>          <1> rootdir: resw 1
54317                                  <1>
54318                                  <1> ; 21/05/2016 - TRDOS 386 (TRDOS v2.0) - priority levels, 3 run queues
54319                                  <1> runq:
54320 00060352 <res 00000002>          <1> runq_event:   resw 1 ; high priority, 'run for event'            ; 2
54321 00060354 <res 00000002>          <1> runq_normal:  resw 1 ; normal/regular priority, 'run as requlat' ; 1
54322 00060356 <res 00000002>          <1> runq_background: resw 1 ; low priority, 'run on background'          ; 0
54323                                  <1> ;
54324 00060358 <res 00000001>          <1> imod:  resb 1
54325 00060359 <res 00000001>          <1> smod:  resb 1
54326 0006035A <res 00000001>          <1> mmod:  resb 1
54327 0006035B <res 00000001>          <1> sysflg:      resb 1
54328                                  <1>
54329                                  <1> alignb 4
54330                                  <1>
54331                                  <1> user:
54332                                  <1>       ; 13/01/2017
54333                                  <1>       ; 19/12/2016
54334                                  <1>       ; 21/05/2016 - TRDOS 386 (TRDOS v2.0)
```

```
54335                                  <1>       ;            [u.pri] usage method modification
54336                                  <1>       ; 04/12/2015
54337                                  <1>       ; 18/10/2015
54338                                  <1>       ; 12/10/2015
54339                                  <1>       ; 21/09/2015
54340                                  <1>       ; 24/07/2015
54341                                  <1>       ; 16/06/2015
54342                                  <1>       ; 09/06/2015
54343                                  <1>       ; 11/05/2015
54344                                  <1>       ; 16/04/2015 (Retro UNIX 386 v1 - 32 bit modifications)
54345                                  <1>       ; 10/10/2013
54346                                  <1>       ; 11/03/2013.
54347                                  <1>       ;Derived from UNIX v1 source code 'user' structure (ux).
54348                                  <1>       ;u.
54349                                  <1>
54350 0006035C <res 00000004>          <1>       u.sp:    resd 1 ; esp (kernel stack at the beginning of 'sysent')
54351 00060360 <res 00000004>          <1>       u.usp:   resd 1 ; esp (kernel stack points to user's registers)
54352 00060364 <res 00000004>          <1>       u.r0:    resd 1 ; eax
54353 00060368 <res 00000002>          <1>       u.cdir:         resw 1
54354 0006036A <res 0000000A>          <1>       u.fp:    resb 10
54355 00060374 <res 00000004>          <1>       u.fofp:         resd 1
54356 00060378 <res 00000004>          <1>       u.dirp:         resd 1
54357 0006037C <res 00000004>          <1>       u.namep:  resd 1
54358 00060380 <res 00000004>          <1>       u.off:   resd 1
54359 00060384 <res 00000004>          <1>       u.base:         resd 1
54360 00060388 <res 00000004>          <1>       u.count:  resd 1
54361 0006038C <res 00000004>          <1>       u.nread:  resd 1
54362 00060390 <res 00000004>          <1>       u.break:  resd 1 ; break
54363 00060394 <res 00000002>          <1>       u.ttyp:         resw 1
54364                                  <1>       ; 10/01/2017 (TRDOS 386, relocation and dword alignment)
54365                                  <1>       ; tty number (rtty, rcvt, wtty)
54366 00060396 <res 00000001>          <1>       u.ttyn:         resb 1 ; 28/07/2013 - Retro Unix 8086 v1 feature only !
54367 00060397 <res 00000001>          <1>       u.resb:   resb 1 ; 10/01/2017 (TRDOS 386, temporary)
54368 00060398 <res 00000010>          <1>       u.dirbuf: resb 16 ; 04/12/2015 (10 -> 16)
54369                                  <1>       ;u.pri:         resw 1 ; 14/02/2014
54370 000603A8 <res 00000001>          <1>       u.quant:  resb 1 ; Retro UNIX 8086 v1 Feature only ! (uquant)
54371 000603A9 <res 00000001>          <1>       u.pri:   resb 1 ; Modification: 21/05/2016 (priority levels: 0, 1, 2)
54372 000603AA <res 00000002>          <1>       u.intr:         resw 1
54373 000603AC <res 00000002>          <1>       u.quit:         resw 1
54374                                  <1>       ;u.emt:         resw 1 ; 10/10/2013
54375                                  <1>       ;u.ilgins: resw 1 ; 10/01/2017
54376 000603AE <res 00000002>          <1>       u.cdrv:         resw 1 ; cdev
54377 000603B0 <res 00000001>          <1>       u.uid:   resb 1 ; uid
54378 000603B1 <res 00000001>          <1>       u.ruid:         resb 1
54379 000603B2 <res 00000001>          <1>       u.bsys:         resb 1
54380 000603B3 <res 00000001>          <1>       u.uno:   resb 1
54381 000603B4 <res 00000004>          <1>         u.upage:  resd 1 ; 16/04/2015 - Retro Unix 386 v1 feature only !
54382 000603B8 <res 00000004>          <1>       u.pgdir:  resd 1 ; 09/03/2015 (page dir addr of process)
54383 000603BC <res 00000004>          <1>       u.ppgdir: resd 1 ; 06/05/2015 (page dir addr of the parent process)
54384 000603C0 <res 00000004>          <1>       u.pbase:  resd 1 ; 20/05/2015 (physical base/transfer address)
54385 000603C4 <res 00000002>          <1>       u.pcount: resw 1 ; 20/05/2015 (byte -transfer- count for page)
54386                                  <1>       ;u.pncount: resw 1
54387                                  <1>              ; 16/06/2015 (byte -transfer- count for page, 'namei', 'mkdir')
54388                                  <1>       ;u.pnbase:  resd 1
54389                                  <1>              ; 16/06/2015 (physical base/transfer address, 'namei', 'mkdir')
54390                                  <1>                     ; 09/06/2015
54391 000603C6 <res 00000001>          <1>       u.kcall:  resb 1 ; The caller is 'namei' (dskr) or 'mkdir' (dskw) sign
54392 000603C7 <res 00000001>          <1>       u.brwdev: resb 1 ; Block device number for direct I/O (bread & bwrite)
54393                                  <1>                     ; 24/07/2015 - 24/06/2015
54394                                  <1>       ;u.args:  resd 1 ; arguments list (line) offset from start of [u.upage]
54395                                  <1>                     ; (arg list/line is from offset [u.args] to 4096 in [u.upage])
54396                                  <1>                     ; ([u.args] points to argument count -argc- address offset)
54397                                  <1>                     ; 24/06/2015
54398                                  <1>       ;u.core:  resd 1 ; physical start address of user's memory space (for sys exec)
54399                                  <1>       ;u.ecore: resd 1 ; physical end address of user's memory space (for sys exec)
54400                                  <1>       ; last error number
54401 000603C8 <res 00000004>          <1>       u.error:  resd 1 ; 28/07/2013 - 09/03/2015
54402                                  <1>                      ; Retro UNIX 8086/386 v1 feature only!
54403                                  <1>                    ; 21/09/2015 (debugging - page fault analyze)
54404 000603CC <res 00000004>          <1>       u.pfcount: resd 1 ; page fault count for (this) process (for sys geterr)
54405                                  <1>              ; 19/12/2016 (TRDOS 386)
54406 000603D0 <res 00000004>          <1>       u.tcb:   resd 1 ; Timer callback address/flag which will be used by timer int
54407                                  <1>              ; 13/01/2017 (TRDOS 386)
54408 000603D4 <res 00000001>          <1>       u.t_lock: resb 1 ; Timer interrupt (callback) lock (unlocked by 'sysrele')
54409 000603D5 <res 00000001>          <1>       u.t_mode: resb 1 ; running mode during timer interrupt (0= system, 0FFh= user)
54410                                  <1>              ; 26/02/2017 (TRDOS 386)
54411 000603D6 <res 00000001>          <1>       u.irqc:        resb 1  ; Count of IRQ callback services (IRQs in use)
54412                                  <1>              ; 28/02/2017 (TRDOS 386)
54413 000603D7 <res 00000001>          <1>       u.irqwait: resb 1 ; IRQ waiting for callback service flag (IRQ number, If > 0)
54414 000603D8 <res 00000001>          <1>       u.r_lock:  resb 1 ; 'IRQ callback service is in progress' flag (IRQ lock)
54415 000603D9 <res 00000001>          <1>       u.r_mode:  resb 1 ; running mode during hadware interrupt
54416                                  <1>              ; 27/02/2017 (TRDOS 386)
54417 000603DA <res 00000001>          <1>       u.fpsave: resb 1  ; TRDOS 386, 'save/restore FPU registers' flag
54418 000603DB <res 00000001>          <1> alignb 4
54419 000603DC <res 0000005E>          <1>       u.fpregs: resb 94 ; 94 byte area for saving and restoring FPU registers
54420                                  <1>
54421 0006043A <res 00000002>          <1> alignb 4
54422                                  <1>
54423                                  <1> U_SIZE      equ $ - user
54424                                  <1>
54425                                  <1> ; 18/10/2015 - Retro UNIX 386 v1 (local variables for 'namei' and 'sysexec')
54426 0006043C <res 00000004>          <1> pcore:  resd 1 ; physical start address of user's memory space (for sys exec)
54427 00060440 <res 00000004>          <1> ecore:  resd 1 ; physical start address of user's memory space (for sys exec)
54428 00060444 <res 00000004>          <1> nbase:      resd 1 ; physical base address for 'namei' & 'sysexec'
54429 00060448 <res 00000002>          <1> ncount: resw 1     ; remain byte count in page for 'namei' & 'sysexec'
54430 0006044A <res 00000002>          <1> argc: resw 1 ; argument count for 'sysexec'
54431 0006044C <res 00000004>          <1> argv: resd 1 ; argument list (recent) address for 'sysexec'
54432                                  <1>
54433                                  <1> ; 03/06/2015 - Retro UNIX 386 v1 Beginning
54434                                  <1> ; 07/04/2013 - 31/07/2013 - Retro UNIX 8086 v1
54435 00060450 <res 00000001>          <1> rw:    resb 1 ;; Read/Write sign (iget)
54436                                  <1>
```

```
54437                                  <1> ;alignb 4
54438                                  <1>
54439                                  <1> ; 24/04/2016
54440 00060451 <res 00000004>          <1> ii:         resd 1 ; first cluster of the program file
54441 00060455 <res 00000004>          <1> i.size:           resd 1 ; size of the program file
54442
54443 00060459 <res 00000003>              alignb 4
54444
54445                                      ; 23/05/2016 (TRDOS 386)
54446                                      ; 14/10/2015 (Retro UNIX 386 v1, 'unix386.s')
54447 0006045C <res 00000004>              cr3reg:      resd 1  ; cr3 register content at the beginning of the timer
54448                                                           ; (or RTC) interrupt handler.
54449
54450                                      ; 10/12/2016 (callback)
54451                                      ; 10/06/2016
54452                                      ; 19/05/2016
54453                                      ; 18/05/2016 - TRDOS 386 feature only !
54454 00060460 <res 00000100>              timer_set: resd 16*4   ; 256 bytes memory space for 16 timer events
54455                                            ; Timer Event Structure: (max. 16 timer events, 16*16 bytes)
54456                                            ;       Owner:             resb 1 ; 0 = free
54457                                            ;                                 ;>0 = process number (u.uno)
54458                                            ;     Callback:    resb 1 ; 0 = response byte address (phy)
54459                                            ;                           1 = callback address (virtual)
54460                                            ;     Interrupt:      resb 1 ; 0 = Timer interrupt (or none)
54461                                            ;                             ; 1 = Real Time Clock interrupt
54462                                            ;     Response:       resb 1 ; 0 to 255, signal return value
54463                                            ;     Count Limit: resd 1 ; count of ticks (total/set)
54464                                            ;     Current Count:      resd 1 ; count of ticks (current)
54465                                            ;     Response Addr:  resd 1 ; response byte (pointer) address
54466                                            ;                             ; (or callback -user service- address)
54467
54468                                      ;; Memory (swap) Data (11/03/2015)
54469                                      ; 09/03/2015
54470 00060560 <res 00000002>              swpq_count: resw 1 ; count of pages on the swap queue
54471 00060562 <res 00000004>              swp_drv:    resd 1 ; logical drive description table address of the swap drive/disk
54472 00060566 <res 00000004>              swpd_size:  resd 1 ; size of swap drive/disk (volume) in sectors (512 bytes).

54473 0006056A <res 00000004>              swpd_free:  resd 1 ; free page blocks (4096 bytes) on swap disk/drive (logical)
54474 0006056E <res 00000004>              swpd_next:  resd 1 ; next free page block
54475 00060572 <res 00000004>              swpd_last:  resd 1 ; last swap page block
54476
54477 00060576 <res 00000002>              alignb 4
54478
54479                                      ; 10/07/2015
54480                                      ; 28/08/2014
54481 00060578 <res 00000004>              error_code: resd 1
54482                                      ; 29/08/2014
54483 0006057C <res 00000004>              FaultOffset:       resd 1
54484                                      ; 21/09/2015
54485 00060580 <res 00000004>              PF_Count:   resd 1 ; total page fault count
54486                                                         ; (for debugging - page fault analyze)
54487                                                         ; 'page_fault_handler' (memory.inc)
54488                                                         ; 'sysgeterr' (u9.s)
54489
54490                                      ; 29/04/2016 (TRDOS 386 = TRDOS v2.0)
54491                                      ; 22/08/2015 (Retro UNIX 386 v1)
54492                                      buffer:
54493 00060584 <res 00000008>                  resb   8
54494                                      readi_buffer:
54495 0006058C <res 00000200>                  resb   512
54496 0006078C <res 00000008>                  resb   8
54497                                      writei_buffer:
54498 00060794 <res 00000200>                  resb   512
54499                                      ; 24/10/2016
54500 00060994 <res 00000008>                  resb   8
54501                                      rw_buffer:
54502 0006099C <res 00000800>                  resb   2048  ; general purposed, r/w sector buffer
54503
54504                                      bss_end:
54505
54506                                      ; 27/12/2013
54507                                      _end:  ; end of kernel code
```