TRDOS386.ASM (TRDOS 386 Kernel) - v2.0.0

```
                                ; ****************************************************************************
                                ; TRDOS386.ASM (TRDOS 386 Kernel) - v2.0.0
                                ; ----------------------------------------------------------------------------
                                ; Last Update: 27/12/2017
                                ; ----------------------------------------------------------------------------
                                ; 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)
                                ; ****************************************************************************
                                ; nasm trdos386.s -l trdos386.txt -o TRDOS386.SYS


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

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



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

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

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

```
                                ; 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)
0000019B 8B4410                 <1>       mov    ax, [si+16]
```

```
411 0000019E 854412              <1>       test   ax, [si+18]
412 000001A1 7412                <1>       jz     short L10_A0h
413                              <1>                    ; 'CHS only' disks on EDD system
414                              <1>                    ;  are reported with ZERO disk size
415 000001A3 81EB[425D]          <1>       sub    bx, drv.status
416 000001A7 C1E302              <1>       shl    bx, 2
417 000001AA 81C3[265D]          <1>       add    bx, drv.size ; disk size (in sectors)
418 000001AE 8907                <1>       mov    [bx], ax
419 000001B0 8B4412              <1>       mov    ax, [si+18]
420 000001B3 8907                <1>       mov    [bx], ax
421                              <1>
422                              <1> L10_A0h: ; Jump here to fix a ZERO (LBA) disk size problem
423                              <1>        ; for CHS disks (28/02/2015)
424                              <1>       ; 30/12/2014
425 000001B5 BF0000              <1>       mov    di, HD0_DPT
426 000001B8 88D0                <1>       mov    al, dl
427 000001BA 83E003              <1>       and    ax, 3
428 000001BD C0E005              <1>       shl    al, 5 ; *32
429 000001C0 01C7                <1>       add    di, ax
430 000001C2 B80090              <1>       mov    ax, DPT_SEGM
431 000001C5 8EC0                <1>       mov    es, ax
432                              <1>       ;
433 000001C7 88E8                <1>       mov    al, ch ; max. cylinder number (bits 0-7)
434 000001C9 88CC                <1>       mov    ah, cl
435 000001CB C0EC06              <1>       shr    ah, 6  ; max. cylinder number (bits 8-9)
436 000001CE 40                  <1>       inc    ax     ; logical cylinders (limit 1024)
437 000001CF AB                  <1>       stosw
438 000001D0 88F0                <1>       mov    al, dh ; max. head number
439 000001D2 FEC0                <1>       inc    al
440 000001D4 AA                  <1>       stosb         ; logical heads (limits 256)
441 000001D5 B0A0                <1>       mov    al, 0A0h ; Indicates translated table
442 000001D7 AA                  <1>       stosb
443 000001D8 8A440C              <1>       mov    al, [si+12]
444 000001DB AA                  <1>       stosb         ; physical sectors per track
445 000001DC 31C0                <1>       xor    ax, ax
446                              <1>       ;dec   ax      ; 02/01/2015
447 000001DE AB                  <1>       stosw         ; precompensation (obsolete)
448                              <1>       ;xor   al, al  ; 02/01/2015
449 000001DF AA                  <1>       stosb         ; reserved
450 000001E0 B008                <1>       mov    al, 8   ; drive control byte
451                              <1>                      ; (do not disable retries,
452                              <1>                     ; more than 8 heads)
453 000001E2 AA                  <1>       stosb
454 000001E3 8B4404              <1>       mov    ax, [si+4]
455 000001E6 AB                  <1>       stosw         ; physical number of cylinders
456                              <1>       ;push  ax      ; 02/01/2015
457 000001E7 8A4408              <1>       mov    al, [si+8]
458 000001EA AA                  <1>       stosb         ; physical num. of heads (limit 16)
459 000001EB 29C0                <1>       sub    ax, ax
460                              <1>       ;pop   ax      ; 02/01/2015
461 000001ED AB                  <1>       stosw         ; landing zone (obsolete)
462 000001EE 88C8                <1>       mov    al, cl  ; logical sectors per track (limit 63)
463 000001F0 243F                <1>       and    al, 3Fh
464 000001F2 AA                  <1>       stosb
465                              <1>       ;sub   al, al  ; checksum
466                              <1>       ;stosb
467                              <1>       ;
468 000001F3 83C61A              <1>       add    si, 26   ; (BIOS) DPTE address pointer
469 000001F6 AD                  <1>       lodsw
470 000001F7 50                  <1>       push   ax      ; (BIOS) DPTE offset
471 000001F8 AD                  <1>       lodsw
472 000001F9 50                  <1>       push   ax      ; (BIOS) DPTE segment
473                              <1>       ;
474                              <1>       ; checksum calculation
475 000001FA 89FE                <1>       mov    si, di
476 000001FC 06                  <1>       push   es
477 000001FD 1F                  <1>       pop    ds
478                              <1>       ;mov   cx, 16
479 000001FE B90F00              <1>       mov    cx, 15
480 00000201 29CE                <1>       sub    si, cx
481 00000203 30E4                <1>       xor    ah, ah
482                              <1>       ;del   cl
483                              <1> L11:
484 00000205 AC                  <1>       lodsb
485 00000206 00C4                <1>       add    ah, al
486 00000208 E2FB                <1>       loop   L11
487                              <1>       ;
488 0000020A 88E0                <1>       mov    al, ah
489 0000020C F6D8                <1>       neg    al     ; -x+x = 0
490 0000020E AA                  <1>       stosb         ; put checksum in byte 15 of the tbl
491                              <1>       ;
492 0000020F 1F                  <1>       pop    ds     ; (BIOS) DPTE segment
493 00000210 5E                  <1>       pop    si     ; (BIOS) DPTE offset
494                              <1>       ;
495                              <1>       ; 23/02/2015
496 00000211 57                  <1>       push   di
497                              <1>       ; ES:DI points to DPTE (FDPTE) location
498                              <1>       ;mov   cx, 8
499 00000212 B108                <1>       mov    cl, 8
500 00000214 F3A5                <1>       rep    movsw
501                              <1>       ;
502                              <1>       ; 23/02/2015
503                              <1>       ; (P)ATA drive and LBA validation
504                              <1>       ; (invalidating SATA drives and setting
505                              <1>       ; CHS type I/O for old type fixed disks)
506 00000216 5B                  <1>       pop    bx
507 00000217 8CC8                <1>       mov    ax, cs
508 00000219 8ED8                <1>       mov    ds, ax
509 0000021B 268B07              <1>       mov    ax, [es:bx]
510 0000021E 3DF001              <1>       cmp    ax, 1F0h
511 00000221 7418                <1>       je     short L11a
512 00000223 3D7001              <1>       cmp    ax, 170h
513 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
000002CB B080                     mov    al, 80h
000002CD E670                     out    70h, al             ; set bit 7 to 1 for disabling NMI
                                  ;23/02/2015
```

```
                                        ;nop                ;
                                        ;in    al, 71h              ; read in 71h just after writing out to 70h
                                                            ; for preventing unknown state (!?)
                                        ;
                                        ; 20/08/2014
                                        ; Moving the kernel 64 KB back (to physical address 0)
                                        ; DS = CS = 1000h
                                        ; 05/11/2014
000002CF 31C0                           xor    ax, ax
000002D1 8EC0                           mov    es, ax ; ES = 0
                                        ;
                                        ; 04/07/2016 -  TRDOS 386 (64K - 128K kernel)
                                              xor    si, si
000002D3 31F6                           xor    di, di
000002D5 31FF                           mov    cx, 16384
000002D7 B90040                         rep    movsd
000002DA F366A5                         ;
                                        push   es ; 0
000002DD 06                             push   L17
000002DE 68[E202]                       retf
000002E1 CB                     L17:
                                              mov    cx, 1000h
000002E2 B90010                               mov    es, cx  ; 1000h
000002E5 8EC1                                 add    cx, cx
000002E7 01C9                                 mov    ds, cx  ; 2000h
000002E9 8ED9                                 sub    si, si
000002EB 29F6                                 sub    di, di
000002ED 29FF                                 mov    cx, 16384
000002EF B90040                               rep    movsd
000002F2 F366A5
                                        ; Turn off the floppy drive motor
000002F5 BAF203                           mov     dx, 3F2h
000002F8 EE                               out     dx, al ; 0 ; 31/12/2013

                                        ; Enable access to memory above one megabyte
                                L18:
000002F9 E464                                 in     al, 64h
000002FB A802                                 test   al, 2
000002FD 75FA                             jnz     short L18
000002FF B0D1                                 mov    al, 0D1h     ; Write output port
00000301 E664                                 out    64h, al
                                L19:
00000303 E464                                 in     al, 64h
00000305 A802                                 test   al, 2
00000307 75FA                             jnz     short L19
00000309 B0DF                                 mov    al, 0DFh     ; Enable A20 line
0000030B E660                                 out    60h, al
                                ;L20:
                                              ;
                                              ; Load global descriptor table register

                                                ;mov     ax, cs
                                                ;mov     ds, ax

0000030D 2E0F0116[605C]                         lgdt    [cs:gdtd]

00000313 0F20C0                                 mov     eax, cr0
                                              ; or   eax, 1
00000316 40                                   inc     ax
00000317 0F22C0                               mov     cr0, eax

                                              ; Jump to 32 bit code

0000031A 66                                   db 66h                    ; Prefix for 32-bit
0000031B EA                                   db 0EAh             ; Opcode for far jump
0000031C [22030000]                           dd StartPM          ; Offset to start, 32-bit
                                                                  ; (1000h:StartPM = StartPM + 10000h)
00000320 0800                                 dw KCODE            ; This is the selector for CODE32_DESCRIPTOR,
                                                                  ; assuming that StartPM resides in code32

                                ; 20/02/2017


                                [BITS 32]

                                StartPM:
                                      ; Kernel Base Address = 0 ; 30/12/2013
00000322 66B81000                     mov ax, KDATA           ; Save data segment identifier
00000326 8ED8                           mov ds, ax              ; Move a valid data segment into DS register
00000328 8EC0                                mov es, ax             ; Move data segment into ES register
0000032A 8EE0                                mov fs, ax             ; Move data segment into FS register
0000032C 8EE8                                mov gs, ax             ; Move data segment into GS register
0000032E 8ED0                           mov ss, ax              ; Move data segment into SS register
00000330 BC00000900                     mov esp, 90000h         ; Move the stack pointer to 090000h

                                clear_bss: ; Clear uninitialized data area
                                      ; 11/03/2015
00000335 31C0                         xor  eax, eax ; 0
00000337 B999700000                   mov  ecx, (bss_end - bss_start)/4
                                      ;shr  ecx, 2 ; bss section is already aligned for double words
0000033C BF[364F0100]                 mov  edi, bss_start
00000341 F3AB                         rep  stosd

                                memory_init:
                                      ; Initialize memory allocation table and page tables
                                      ; 16/11/2014
                                      ; 15/11/2014
                                      ; 07/11/2014
                                      ; 06/11/2014
                                      ; 05/11/2014
                                      ; 04/11/2014
                                      ; 31/10/2014 (Retro UNIX 386 v1 - Beginning)
                                      ;
```

```
                                ;       xor     eax, eax
                                ;       xor     ecx, ecx
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[28520100]                 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[28520100]                 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[24520100]                   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[38520100]                 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[20520100]                 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[30520100]                   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[38520100]               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[28520100]                 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[24520100]                 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[28520100]                 sub    [free_pages], ecx ; 07/11/2014
                                        ;
00000442 8B35[20520100]                 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[20520100]                 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[24520100]                 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[34520100]               mov    [first_page], eax
000004B1 A3[2C520100]               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[20520100]                 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[8D130100]               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[50100100]               mov    esi, ilist
00000513 8D3D[384F0100]             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[B8510100]                 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[1E520100]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[24520100]               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[BC510100]               mov   [tss.esp0], esp
000005B3 66C705[C0510100]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
000005C2 E6A1                         out   0A1h, al     ;  interrupts will be in use.)
                                                         ; (Even if related hardware component
                                                         ;  does not exist!)
                                      ; Enable NMI
000005C4 B07F                         mov   al, 7Fh            ; Clear bit 7 to enable NMI (again)
000005C6 E670                         out   70h, al
                                      ; 23/02/2015
000005C8 90                           nop
```

```
1132 000005C9 E471                      in     al, 71h             ; read in 71h just after writing out to 70h
1133                                                              ; for preventing unknown state (!?)
1134                                    ;
1135                                    ; Only a NMI can occur here... (Before a 'STI' instruction)
1136                                    ;
1137                                    ; 02/09/2014
1138 000005CB 6631DB                    xor    bx, bx
1139 000005CE 66BA0002                  mov    dx, 0200h    ; Row 2, column 0  ; 07/03/2015
1140 000005D2 E871170000                call   _set_cpos    ; 24/01/2016
1141                                    ;
1142                                    ; 06/11/2014
1143 000005D7 E8782C0000                call   memory_info
1144                                    ; 14/08/2015
1145                                    ;call getch ; 28/02/2015
1146                                drv_init:
1147 000005DC FB                        sti    ; Enable Interrupts
1148                                    ; 06/02/2015
1149 000005DD 8B15[F85C0000]            mov    edx, [hd0_type] ; hd0, hd1, hd2, hd3
1150 000005E3 668B1D[F65C0000]          mov    bx, [fd0_type] ; fd0, fd1
1151                                    ; 22/02/2015
1152 000005EA 6621DB                    and    bx, bx
1153 000005ED 751C                      jnz    short di1
1154                                    ;
1155 000005EF 09D2                      or     edx, edx
1156 000005F1 752A                      jnz    short di2
1157                                    ;
1158                                setup_error:
1159 000005F3 BE[56130100]              mov    esi, setup_error_msg
1160                                psem:
1161 000005F8 AC                        lodsb
1162 000005F9 08C0                      or     al, al
1163                                    ;jz    short haltx ; 22/02/2015
1164 000005FB 7427                      jz     short di3
1165 000005FD 56                        push   esi
1166                                    ; 13/05/2016
1167 000005FE BB07000000                mov    ebx, 7 ; Black background,
1168                                                   ; light gray forecolor
1169                                                   ; Video page 0 (BH=0)
1170 00000603 E8AA160000                call   _write_tty
1171 00000608 5E                        pop    esi
1172 00000609 EBED                      jmp    short psem
1173
1174                                  di1:
1175                                    ; supress 'jmp short T6'
1176                                    ;  (activate fdc motor control code)
1177 0000060B 66C705[EB060000]90-       mov    word [T5], 9090h ; nop
1178 00000613 90
1179                                    ;
1180                                    ;mov   ax, int_0Eh  ; IRQ 6 handler
1181                                    ;mov   di, 0Eh*4    ; IRQ 6 vector
1182                                    ;stosw
1183                                    ;mov   ax, cs
1184                                    ;stosw
1185                                    ;; 16/02/2015
1186                                      ;;mov     dword [DISKETTE_INT], fdc_int ; IRQ 6 handler
1187                                    ;
1188 00000614 E8AF3B0000                CALL   DSKETTE_SETUP; Initialize Floppy Disks
1189                                    ;
1190 00000619 09D2                      or     edx, edx
1191 0000061B 7407                        jz     short di3
1192                                  di2:
1193 0000061D E8EC3B0000                call         DISK_SETUP   ; Initialize Fixed Disks
1194 00000622 72CF                        jc     short setup_error
1195                                  di3:
1196 00000624 E8FF2B0000                call   setup_rtc_int; 22/05/2015 (dsectrpm.s)
1197                                    ;
1198 00000629 E8BE0B0100                call   display_disks ; 07/03/2015  (Temporary)
1199                                  ;haltx:
1200                                    ; 14/08/2015
1201                                    ;call  getch ; 22/02/2015
1202                                    ;sti   ; Enable interrupts (for CPU)
1203                                    ;      ; 29/01/2016
1204                                    ;      sub    ah, ah ;  read time count
1205                                    ;      call   int1Ah
1206                                    ;      mov    edx, ecx ; 18.2 * seconds
1207                                    ;md_info_msg_wait1:
1208                                    ;      ; 29/01/2016
1209                                    ;      mov    ah, 1
1210                                    ;      call   int16h
1211                                    ;      jz     short md_info_msg_wait2
1212                                    ;      xor    ah, ah ; 0
1213                                    ;        call    int16h
1214                                    ;      jmp    short md_info_msg_ok
1215                                    ;md_info_msg_wait2:
1216                                    ;      sub    ah, ah  ; read time count
1217                                    ;      call   int1Ah
1218                                    ;      cmp    edx, ecx ; ; 18.2 * seconds
1219                                    ;      jna    short md_info_msg_wait3
1220                                    ;      xchg   edx, ecx
1221                                    ;md_info_msg_wait3:
1222                                    ;      sub    ecx, edx
1223                                    ;      cmp    ecx, 127 ; 7 seconds (18.2 * 7)
1224                                    ;      jb     short md_info_msg_wait1
1225                                    ;md_info_msg_ok:
1226                                    ; 08/09/2016
1227 0000062E 0F20C0                    mov    eax, cr0
1228 00000631 A810                      test   al, 10h  ; Bit 4, ET (Extension Type)
1229 00000633 7408                      jz     short sysinit
1230                                    ; 27/02/2017
1231 00000635 FE05[E05F0100]            inc    byte [fpready]
1232                                    ; 80387 (FPU) is ready
1233 0000063B DBE3                      fninit ; Initialize Floating-Point Unit
1234                                sysinit:
```

```
                                        ; 30/06/2015
0000063D E80C5C0000                     call   sys_init
                                        ;
                                        ;jmp   cpu_reset ; 22/02/2015
                                hang:
                                        ; 23/02/2015
                                        ;sti               ; Enable interrupts
00000642 F4                             hlt
                                        ;
                                        ;nop
                                        ;; 03/12/2014
                                        ;; 28/08/2014
                                        ;mov   ah, 11h
                                        ;call  getc
                                        ;jz      _c8
                                        ;
                                        ; 23/02/2015
                                        ; 06/02/2015
                                        ; 07/09/2014
00000643 31DB                           xor    ebx, ebx
00000645 8A1D[4E520100]                 mov    bl, [ptty]   ; active_page
0000064B 89DE                           mov    esi, ebx
0000064D 66D1E6                         shl    si, 1
00000650 81C6[50520100]                 add    esi, ttychr
00000656 668B06                         mov    ax, [esi]
00000659 6621C0                         and    ax, ax
                                        ;jz    short _c8
0000065C 74E4                           jz     short hang
0000065E 66C7060000                     mov    word [esi], 0
00000663 80FB03                         cmp    bl, 3        ; Video page 3
                                        ;jb    short _c8
00000666 72DA                           jb     short hang
                                        ;
                                        ; 13/05/2016
                                        ; 07/09/2014
                                nxtl:
00000668 6653                           push   bx
0000066A 66BB0E00                       mov    bx, 0Eh      ; Yellow character
                                                            ; on black background
                                                            ; bh = 0 (video page 0)
                                                            ; Retro UNIX 386 v1 - Video Mode 0
                                                            ; (PC/AT Video Mode 3 - 80x25 Alpha.)
0000066E 6650                           push   ax
00000670 E83D160000                     call   _write_tty
00000675 6658                           pop    ax
00000677 665B                           pop    bx
00000679 3C0D                           cmp    al, 0Dh            ; carriage return (enter)
                                        ;jne   short _c8
0000067B 75C5                           jne    short hang
0000067D B00A                           mov    al, 0Ah            ; next line
0000067F EBE7                           jmp    short nxtl
                                
                                ;_c8:
                                ;       ; 25/08/2014
                                ;       cli                       ; Disable interrupts
                                ;       mov    al, [scounter + 1]
                                ;       and    al, al
                                ;       jnz    hang
                                ;       call   rtc_p
                                ;       jmp     hang
                                
                                        ; 27/08/2014
                                        ; 20/08/2014
                                printk:
                                          ;mov    edi, [scr_row]
                                pkl:
00000681 AC                             lodsb
00000682 08C0                           or     al, al
00000684 7404                           jz     short pkr
00000686 66AB                           stosw
00000688 EBF7                           jmp    short pkl
                                pkr:
0000068A C3                             retn
                                
                                ; 28/02/2017
                                ; 22/01/2017
                                ; 15/01/2017
                                ; 14/01/2017
                                ; 02/01/2017
                                ; 25/12/2016
                                ; 19/12/2016
                                ; 10/12/2016 (callback)
                                ; 06/06/2016
                                ; 23/05/2016
                                ; 22/05/2016 - TRDOS 386 (TRDOS v2.0) Timer Event Modifications
                                ; 25/07/2015
                                ; 14/05/2015 (multi tasking -time sharing- 'clock', x_timer)
                                ; 17/02/2015
                                ; 06/02/2015 (unix386.s)
                                ; 11/12/2014 - 22/12/2014 (dsectrm2.s)
                                ;
                                ; IBM PC-XT Model 286 Source Code - BIOS2.ASM (06/10/85)
                                ;
                                ;-- HARDWARE INT  08 H - ( IRQ LEVEL 0 ) ---------------------------------------
                                ;     THIS ROUTINE HANDLES THE TIMER INTERRUPT FROM FROM CHANNEL 0 OF        :
                                ;     THE 8254 TIMER.  INPUT FREQUENCY IS 1.19318 MHZ AND THE DIVISOR        :
                                ;     IS 65536, RESULTING IN APPROXIMATELY 18.2 INTERRUPTS EVERY SECOND.     :
                                ;                                                                 :
                                ;     THE INTERRUPT HANDLER MAINTAINS A COUNT (40:6C) OF INTERRUPTS SINCE    :
                                ;     POWER ON TIME, WHICH MAY BE USED TO ESTABLISH TIME OF DAY.        :
                                ;     THE INTERRUPT HANDLER ALSO DECREMENTS THE MOTOR CONTROL COUNT (40:40)  :
                                ;     OF THE DISKETTE, AND WHEN IT EXPIRES, WILL TURN OFF THE           :
```

```
                                ;     DISKETTE MOTOR(s), AND RESET THE MOTOR RUNNING FLAGS.                  :
                                ;     THE INTERRUPT HANDLER WILL ALSO INVOKE A USER ROUTINE THROUGH           :
                                ;     INTERRUPT 1CH AT EVERY TIME TICK.  THE USER MUST CODE A          :
                                ;     ROUTINE AND PLACE THE CORRECT ADDRESS IN THE VECTOR TABLE.       :
                                ;-------------------------------------------------------------------------------
                                ;

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

0000068B 1E                           push   ds
0000068C 06                           push   es
0000068D 0FA0                         push   fs
0000068F 0FA8                         push   gs

00000691 60                           pushad ; eax, ecx, edx, ebx, esp -before pushad-, ebp, esi, edi
00000692 66B91000                     mov     cx, KDATA
00000696 8ED9                           mov     ds, cx
00000698 8EC1                           mov     es, cx
0000069A 8EE1                           mov     fs, cx
0000069C 8EE9                           mov     gs, cx

0000069E 0F20D9                       mov    ecx, cr3
000006A1 890D[5C040300]               mov    [cr3reg], ecx ; save current cr3 register value/content

                                      ; 14/01/2017
000006A7 3B0D[20520100]               cmp    ecx, [k_page_dir]
000006AD 7409                         je     short T3

000006AF 8B0D[20520100]                     mov    ecx, [k_page_dir]
000006B5 0F22D9                             mov    cr3, ecx
                                T3:
                                      ;sti                        ; INTERRUPTS BACK ON
000006B8 66FF05[A0520100]             INC    word [TIMER_LOW]     ; INCREMENT TIME
000006BF 7507                         JNZ    short T4             ; GO TO TEST_DAY
000006C1 66FF05[A2520100]             INC    word [TIMER_HIGH]    ; INCREMENT HIGH WORD OF TIME
                                T4:                               ; TEST_DAY
000006C8 66833D[A2520100]18           CMP    word [TIMER_HIGH],018H    ; TEST FOR COUNT EQUALING 24 HOURS
000006D0 7519                         JNZ    short T5             ; GO TO DISKETTE_CTL
000006D2 66813D[A0520100]B0-          CMP    word [TIMER_LOW],0B0H
000006DA 00
000006DB 750E                         JNZ    short T5             ; GO TO DISKETTE_CTL

                                ;-----       TIMER HAS GONE 24 HOURS
                                      ;;SUB  AX,AX
                                      ;MOV   [TIMER_HIGH],AX
                                      ;MOV   [TIMER_LOW],AX
000006DD 29C0                         sub    eax, eax
000006DF A3[A0520100]                 mov    [TIMER_LH], eax
                                      ;
000006E4 C605[A4520100]01             MOV    byte [TIMER_OFL],1

                                ;-----       TEST FOR DISKETTE TIME OUT

                                T5:
                                      ; 23/12/2014
000006EB EB1D                         jmp    short T6             ; will be replaced with nop, nop
                                                                    ; (9090h) if a floppy disk
                                                                    ; is detected.
                                      ;mov   al,[CS:MOTOR_COUNT]
000006ED A0[A7520100]                 mov    al, [MOTOR_COUNT]
000006F2 FEC8                         dec    al
                                      ;mov   [CS:MOTOR_COUNT], al      ; DECREMENT DISKETTE MOTOR CONTROL
000006F4 A2[A7520100]                 mov    [MOTOR_COUNT], al
                                      ;mov   [ORG_MOTOR_COUNT], al
000006F9 750F                         JNZ    short T6             ; RETURN IF COUNT NOT OUT
000006FB B0F0                         mov    al,0F0h
                                      ;AND   [CS:MOTOR_STATUS],al      ; TURN OFF MOTOR RUNNING BITS
000006FD 2005[A6520100]               and    [MOTOR_STATUS], al
                                      ;and   [ORG_MOTOR_STATUS], al
00000703 B00C                         MOV    AL,0CH               ; bit 3 = enable IRQ & DMA,
                                                                    ; bit 2 = enable controller
                                                                    ;     1 = normal operation
                                                                    ;     0 = reset
                                                                    ; bit 0, 1 = drive select
                                                                    ; bit 4-7 = motor running bits
00000705 66BAF203                     MOV    DX,03F2H             ; FDC CTL PORT
00000709 EE                           OUT    DX,AL                ; TURN OFF THE MOTOR
                                T6:
                                      ;inc   word [CS:wait_count]      ; 22/12/2014 (byte -> word)
                                                                    ; TIMER TICK INTERRUPT
                                      ;;inc  word [wait_count] ;;27/02/2015
                                      ;INT   1CH                  ; TRANSFER CONTROL TO A USER ROUTINE
                                      ;cli
0000070A E857040000                   call   u_timer                   ; TRANSFER CONTROL TO A USER ROUTINE
                                      ; 23/05/2016
0000070F E823EC0000                   call   clock                ; Multi Tasking control procedure
                                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[D4030300]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[D8030300]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[B45F0100]02               cmp    byte [priority], 2
00000732 733A                           jnb    short T8  ; current process has a timer event (15/01/2017)
                                        ; 22/05/2016
00000734 803D[B55F0100]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[B55F0100]                   mov    [p_change], al ; 0
                                        ;mov   byte [priority], 2 ; 15/01/2017 (there is a timer event)

00000744 803D[5B030300]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[A8030300]                   mov    [u.quant], al ; 0

                                timer_int_return: ; 23/05/2016 - jump from 'rtc_int' ('rtc_int_2')
00000752 8B0D[5C040300]                 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[5B030300]                 inc    byte [sysflg]       ; now, we are in system space
                                        ;
00000769 E98FBD0000                       jmp     sysrelease ; change running process immediatelly

                                T8:
                                        ; 13/01/2017 (eax -> ebx)
                                        ; callback checking... (19/12/2016)
0000076E 31DB                           xor    ebx, ebx
00000770 871D[D0030300]                 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[D4030300]                 inc    byte [u.t_lock]
00000780 8A0D[5B030300]                 mov    cl, [sysflg]
00000786 880D[D5030300]                 mov    [u.t_mode], cl

0000078C 8B2D[BC510100]                 mov    ebp, [tss.esp0] ; kernel stack address (for ring 0)
00000792 83ED14                         sub    ebp, 20             ; eip, cs, eflags, esp, ss
00000795 892D[5C030300]                 mov    [u.sp], ebp
0000079B 8925[60030300]                 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[64030300]                   mov    [u.r0], eax

000007AA E8A3DF0000                     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[5B030300]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
000007BB 50                             push   eax
000007BC 50                             push   eax
000007BD 89E7                           mov    edi, esp
000007BF 893D[60030300]                 mov    [u.usp], edi
000007C5 B908000000                     mov    ecx, ((ESPACE/4) - 4) ; except DS, ES, FS, GS
000007CA F3A5                           rep    movsd
000007CC B104                           mov    cl, 4
000007CE F3AB                           rep    stosd
000007D0 893D[5C030300]                 mov    [u.sp], edi
000007D6 89EE                           mov    esi, ebp
000007D8 B105                           mov    cl, 5 ; EIP (u), CS (UCODE), EFLAGS (u), ESP (u), SS (UDATA)
000007DA F3A5                           rep    movsd
```

```
1544
1545 000007DC 8B0D[B8030300]              mov    ecx, [u.pgdir]
1546 000007E2 890D[5C040300]              mov    [cr3reg], ecx
1547
1548                                      ; 13/01/207 (eax -> ebx)
1549                                      ; EBX = callback routine address (virtual, not physical address!)
1550
1551                                      ; 09/01/2017
1552                                      ; !!! CALLBACK ROUTINE MUST BE ENDED/RETURNED WITH 'sysrele'
1553                                      ;     system call !!!
1554                                      ; 25/12/2016
1555                                      ; Callback Note: (19/12/2016)
1556                                      ; !!! CALLBACK ROUTINE MUST BE ENDED/RETURNED WITH 'RETN' !!!
1557                                      ;     pushf ; save flags
1558                                      ;     <callback service code>
1559                                      ;     popf  ; restore flags
1560                                      ;     retn ; return to normal running address
1561                                      ;
1562
1563                                      ; 15/01/2017
1564                                      ; 14/01/2017
1565                                      ; 13/01/2017 (eax -> ebx)
1566                                      ; 10/01/2017
1567                                  set_callback_addr:
1568                                      ; 09/01/2017 (**)
1569                                      ; 02/01/2017 (*)
1570                                      ; 25/12/2016 (*)
1571                                      ; 19/12/2016 (TRDOS 386 feature only!)
1572                                      ;
1573                                      ; This routine sets return address
1574                                      ; to start of user's interrupt
1575                                      ; service (callback) address
1576                                      ;; and sets callback 'retn' address to normal
1577                                      ;; return address of user's running code!
1578                                      ;
1579                                      ; INPUT:
1580                                      ;     EBX = callback routine/service address
1581                                      ;           (virtual, not physical address!)
1582                                      ;     [u.sp] = kernel stack, points to
1583                                      ;            user's EIP,CS,EFLAGS,ESP,SS
1584                                      ;            registers.
1585                                      ; OUTPUT:
1586                                      ;     EIP (user) = callback (service) address
1587                                      ;     CS (user) = UCODE
1588                                      ;     EFLAGS (user) = flags before callback
1589                                      ;       ESP (user) = ESP-4 (user, before callback)
1590                                      ;     [ESP](user) = EIP (user) before callback
1591                                      ;
1592                                      ; Note: If CPU was in user mode while entering
1593                                      ;     the timer interrupt service routine,
1594                                      ;     'IRET' will get return to callback routine
1595                                      ;     immediately. If CPU was in system/kernel mode
1596                                      ;     'iret' will get return to system call and
1597                                      ;     then, callback routine will be return address
1598                                      ;     from system call. (User's callback/service code
1599                                      ;     will be able to return to normal return address
1600                                      ;     via an 'retn' at the end.)
1601                                      ;
1602                                      ; Note(**): User's callback service code must be ended
1603                                      ;     with a 'sysrele' sytstem call ! (09/01/2017)
1604                                      ;
1605                                      ;     For example:
1606                                      ;
1607                                      ;     timer_callback:
1608                                      ;         ...
1609                                      ;         inc      dword [time_counter]
1610                                      ;         ...
1611                                      ;         mov eax, 39 ; 'sysrele'
1612                                      ;         int 40h ; TRDOS 386 system call (interrupt)
1613                                      ;
1614                                      ;
1615                                      ;; Note(*): User's callback service code must preserve cpu
1616                                      ;;     flags if it has any instructions which changes
1617                                      ;;     flags in the service code. (25/12/2016)
1618                                      ;;
1619                                      ;;     For example:
1620                                      ;;
1621                                      ;;     timer_callback:
1622                                      ;;         pushf ; save flags
1623                                      ;;         ; this instruction changes zero flag
1624                                      ;;         inc      dword [time_counter]
1625                                      ;;         popf ; restore flags
1626                                      ;;         retn ; return to normal user code
1627                                      ;;              (which is interrupted by the
1628                                      ;;               timer interput)
1629                                      ;;
1630
1631                                      ; 15/01/2017
1632 000007E8 8B2D[5C030300]              mov    ebp, [u.sp]; kernel's stack, points to EIP (user)
1633 000007EE 895D00                     mov    [ebp], ebx
1634 000007F1 E95CFFFFFF                 jmp    timer_int_return
1635
1636                                      ; 15/01/2017
1637                                      ; 13/01/2017
1638                                      ; 19/12/2016
1639                                      ; 06/06/2016
1640                                      ; 23/05/2016
1641                                      ; 22/05/2016
1642                                      ; 19/05/2016 - TRDOS 386 (TRDOS v2.0)
1643                                      ; 26/02/2015
1644                                      ; 07/09/2014
1645                                      ; 25/08/2014
1646                                  rtc_int:       ; Real Time Clock Interrupt (IRQ 8)
```

```
                                        ; 22/05/2016
000007F6 1E                             push   ds ; ** ; 23/05/2016
000007F7 50                             push   eax ; *
000007F8 66B81000                       mov    ax, KDATA
000007FC 8ED8                           mov    ds, ax
                                        ;
000007FE 8A25[9E520100]                 mov    ah, [RTC_2Hz] ;  2 Hz interrupt to 1 Hz function
00000804 80F401                         xor    ah, 1
00000807 8825[9E520100]                 mov    [RTC_2Hz], ah ; 1 = 0.5 second, 0 = 1 second
0000080D 753B                           jnz    short rtc_int_return ; half second
                                        ; 1 second
                                rtc_int_0:
                                        ; 22/05/2016
0000080F 58                             pop    eax ; *
                                        ;
                                        ; 14/10/2015 ('timer_int')
                                        ; Here, we are simulating system call entry (for task switch)
                                        ; (If multitasking is enabled,
                                        ; 'clock' procedure may jump to 'sysrelease')
                                        ;push  ds ; ** ; 23/05/2016
00000810 06                             push   es
00000811 0FA0                           push   fs
00000813 0FA8                           push   gs
00000815 60                             pushad  ; eax, ecx, edx, ebx, esp -before pushad-, ebp, esi, edi
00000816 66B91000                       mov     cx, KDATA
                                          ;mov    ds, cx ; 06/06/2016
0000081A 8EC1                             mov     es, cx
0000081C 8EE1                             mov     fs, cx
0000081E 8EE9                             mov     gs, cx
                                        ;
00000820 0F20D9                         mov    ecx, cr3
00000823 890D[5C040300]                 mov    [cr3reg], ecx ; save current cr3 register value/content
                                        ;
00000829 803D[D4030300]00               cmp    byte [u.t_lock], 0 ; timer lock (callback) status ?
00000830 7711                           ja     short rtc_int_1        ; yes
                                        ; 15/01/2017
00000832 3B0D[20520100]                 cmp    ecx, [k_page_dir]
00000838 7409                           je     short rtc_int_1
00000839 8B0D[20520100]                 mov    ecx, [k_page_dir]
00000840 0F22D9                         mov    cr3, ecx
                                rtc_int_1:
                                        ; Timer event (kernel) functions must be performed with
                                        ; 1 second intervals - TRDOS 386 (TRDOS v2.0) feature ! -
                                        ;
                                        ; 25/08/2014
00000843 E81A030000                     call   rtc_p  ; 19/05/2016 - major modification
                                        ; 23/05/2016
00000848 28E4                           sub    ah, ah ; 0
                                        ; 22/05/2016 - TRDOS 386 timer event modifications
                                rtc_int_return: ; 19/05/2016
                                        ; 22/02/2015 - dsectpm.s
                                        ; [ source: http://wiki.osdev.org/RTC ]
                                        ; read status register C to complete procedure
                                        ;(it is needed to get a next IRQ 8)
0000084A B00C                           mov    al, 0Ch ;
0000084C E670                           out    70h, al ; select register C
0000084E 90                             nop
0000084F E471                           in     al, 71h ; just throw away contents
                                        ; 22/02/2015
00000851 B020                           MOV    AL,EOI       ; END OF INTERRUPT
                                        ;CLI                ; DISABLE INTERRUPTS TILL STACK CLEARED
00000853 E6A0                           OUT    INTB00,AL    ; FOR CONTROLLER #2
                                        ; 23/05/2016
00000855 B020                           MOV    AL,EOI       ; GET END OF INTERRUPT MASK
00000857 FA                             CLI                 ; DISABLE INTERRUPTS TILL STACK CLEARED
00000858 E620                           OUT    INTA00,AL    ; END OF INTERRUPT TO 8259 - 1
                                        ;
                                        ; 23/05/2016
0000085A 20E4                           and    ah, ah
0000085C 0F84B7FEFFFF                     jz      rtc_int_2
                                        ; ah = 1 (half second)
00000862 58                             pop    eax ; *
00000863 1F                             pop    ds  ; **
00000864 CF                             iretd
                                ; ///////////////
                                        ; 28/08/2014
                                irq0:
00000865 6A00                             push       dword 0
00000867 EB48                           jmp    short which_irq
                                irq1:
00000869 6A01                             push       dword 1
0000086B EB44                           jmp    short which_irq
                                irq2:
0000086D 6A02                             push       dword 2
0000086F EB40                           jmp    short which_irq
                                irq3:
                                        ; 20/11/2015
                                        ; 24/10/2015
00000871 2EFF15[F5F50000]               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[F1F50000]               call   dword [cs:com1_irq4]
00000883 6A04                             push       dword 4
```

```
1750 00000885 EB2A                      jmp    short which_irq
1751                                irq5:
1752 00000887 6A05                        push       dword 5
1753 00000889 EB26                      jmp    short which_irq
1754                                irq6:
1755 0000088B 6A06                        push       dword 6
1756 0000088D EB22                      jmp    short which_irq
1757                                irq7:
1758 0000088F 6A07                        push       dword 7
1759 00000891 EB1E                      jmp    short which_irq
1760                                irq8:
1761 00000893 6A08                        push       dword 8
1762 00000895 EB1A                      jmp    short which_irq
1763                                irq9:
1764 00000897 6A09                        push       dword 9
1765 00000899 EB16                      jmp    short which_irq
1766                                irq10:
1767 0000089B 6A0A                        push       dword 10
1768 0000089D EB12                      jmp    short which_irq
1769                                irq11:
1770 0000089F 6A0B                        push       dword 11
1771 000008A1 EB0E                      jmp    short which_irq
1772                                irq12:
1773 000008A3 6A0C                        push       dword 12
1774 000008A5 EB0A                      jmp    short which_irq
1775                                irq13:
1776 000008A7 6A0D                        push       dword 13
1777 000008A9 EB06                      jmp    short which_irq
1778                                irq14:
1779 000008AB 6A0E                        push       dword 14
1780 000008AD EB02                      jmp    short which_irq
1781                                irq15:
1782 000008AF 6A0F                        push       dword 15
1783                                    ;jmp   short which_irq
1784
1785                                    ; 22/01/2017
1786                                    ; 19/10/2015
1787                                    ; 29/08/2014
1788                                    ; 21/08/2014
1789                                which_irq:
1790 000008B1 870424                    xchg   eax, [esp]  ; 28/08/2014
1791 000008B4 53                        push   ebx
1792 000008B5 56                        push   esi
1793 000008B6 57                        push   edi
1794 000008B7 1E                        push   ds
1795 000008B8 06                        push   es
1796                                    ;
1797 000008B9 88C3                      mov    bl, al
1798                                    ;
1799 000008BB B810000000                mov    eax, KDATA
1800 000008C0 8ED8                      mov    ds, ax
1801 000008C2 8EC0                      mov    es, ax
1802                                    ; 19/10/2015
1803 000008C4 FC                        cld
1804                                      ; 27/08/2014
1805 000008C5 8105[48100100]A000-         add     dword [scr_row], 0A0h
1806 000008CD 0000
1807                                    ;
1808 000008CF B417                      mov    ah, 17h      ; blue (1) background,
1809                                                  ; light gray (7) forecolor
1810 000008D1 8B3D[48100100]              mov     edi, [scr_row]
1811 000008D7 B049                      mov    al, 'I'
1812 000008D9 66AB                      stosw
1813 000008DB B052                      mov    al, 'R'
1814 000008DD 66AB                      stosw
1815 000008DF B051                      mov    al, 'Q'
1816 000008E1 66AB                      stosw
1817 000008E3 B020                      mov    al, ' '
1818 000008E5 66AB                      stosw
1819 000008E7 88D8                      mov    al, bl
1820 000008E9 3C0A                      cmp    al, 10
1821 000008EB 7208                      jb     short ii1
1822 000008ED B031                      mov    al, '1'
1823 000008EF 66AB                      stosw
1824 000008F1 88D8                      mov    al, bl
1825 000008F3 2C0A                      sub    al, 10
1826                                ii1:
1827 000008F5 0430                      add    al, '0'
1828 000008F7 66AB                      stosw
1829 000008F9 B020                      mov    al, ' '
1830 000008FB 66AB                      stosw
1831 000008FD B021                      mov    al, '!'
1832 000008FF 66AB                      stosw
1833 00000901 B020                      mov    al, ' '
1834 00000903 66AB                      stosw
1835                                    ; 23/02/2015
1836 00000905 80FB07                    cmp    bl, 7 ; check for IRQ 8 to IRQ 15
1837 00000908 7604                      jna    ii2
1838                                    ; 22/01/2017
1839 0000090A B020                      mov    al, 20h  ; END OF INTERRUPT COMMAND TO
1840 0000090C E6A0                      out    0A0h, al ; the 2nd 8259
1841                                ii2:
1842 0000090E B020                      mov    al, 20h  ; END OF INTERRUPT COMMAND TO
1843 00000910 E620                      out    20h, al ; the 2nd 8259
1844 00000912 E9CD010000                jmp     iiret
1845                                    ;
1846                                    ; 22/08/2014
1847                                    ;mov   al, 20h ; END OF INTERRUPT COMMAND TO 8259
1848                                    ;out   20h, al      ; 8259 PORT
1849                                    ;
1850                                    ;pop   es
1851                                    ;pop   ds
1852                                    ;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:
00000980 6A18                             push      dword 24
00000982 EB2A                           jmp   short cpu_except
                                exc25:
00000984 6A19                             push      dword 25
00000986 EB26                           jmp   short cpu_except
                                exc26:
00000988 6A1A                             push      dword 26
0000098A EB22                           jmp   short cpu_except
                                exc27:
0000098C 6A1B                             push      dword 27
0000098E EB1E                           jmp   short cpu_except
                                exc28:
00000990 6A1C                             push      dword 28
00000992 EB1A                           jmp   short cpu_except
                                exc29:
00000994 6A1D                             push      dword 29
```

```
1956 00000996 EB16                      jmp    short cpu_except
1957                                 exc30:
1958 00000998 6A1E                     push       dword 30
1959 0000099A EB04                   jmp    short cpu_except_en
1960                                 exc31:
1961 0000099C 6A1F                     push       dword 31
1962 0000099E EB0E                     jmp     short cpu_except
1963
1964                                   ; 19/10/2015
1965                                   ; 19/09/2015
1966                                   ; 01/09/2015
1967                                   ; 28/08/2015
1968                                   ; 28/08/2014
1969                                 cpu_except_en:
1970 000009A0 87442404                 xchg   eax, [esp+4] ; Error code
1971 000009A4 36A3[78050300]           mov    [ss:error_code], eax
1972 000009AA 58                       pop    eax  ; Exception number
1973 000009AB 870424                   xchg   eax, [esp]
1974                                          ; eax = eax before exception
1975                                          ; [esp] -> exception number
1976                                          ; [esp+4] -> EIP to return
1977                                   ; 22/01/2017
1978                                   ; 19/10/2015
1979                                   ; 19/09/2015
1980                                   ; 01/09/2015
1981                                   ; 28/08/2015
1982                                   ; 29/08/2014
1983                                   ; 28/08/2014
1984                                   ; 25/08/2014
1985                                   ; 21/08/2014
1986                                 cpu_except:  ; CPU Exceptions
1987 000009AE FC                       cld
1988 000009AF 870424                   xchg   eax, [esp]
1989                                          ; eax = Exception number
1990                                          ; [esp] = eax (before exception)
1991 000009B2 53                       push   ebx
1992 000009B3 56                       push   esi
1993 000009B4 57                       push   edi
1994 000009B5 1E                       push   ds
1995 000009B6 06                       push   es
1996                                   ; 28/08/2015
1997 000009B7 66BB1000                 mov    bx, KDATA
1998 000009BB 8EDB                     mov    ds, bx
1999 000009BD 8EC3                     mov    es, bx
2000 000009BF 0F20DB                   mov    ebx, cr3
2001 000009C2 53                       push   ebx ; (*) page directory
2002                                   ; 19/10/2015
2003 000009C3 FC                       cld
2004                                   ; 25/03/2015
2005 000009C4 8B1D[20520100]           mov    ebx, [k_page_dir]
2006 000009CA 0F22DB                   mov    cr3, ebx
2007                                   ; 28/08/2015
2008 000009CD 83F80E                   cmp    eax, 0Eh ; 14, PAGE FAULT
2009 000009D0 750F                     jne    short cpu_except_nfp
2010 000009D2 E87B440000               call   page_fault_handler
2011 000009D7 21C0                     and    eax, eax
2012 000009D9 0F8401010000               jz   iiretp ; 01/09/2015
2013 000009DF B00E                     mov    al, 0Eh ; 14
2014                                 cpu_except_nfp:
2015                                   ; 23/08/2016
2016 000009E1 803D[C25E0000]03         cmp    byte [CRT_MODE], 3
2017 000009E8 7409                     je     short cpu_except_mode_3
2018 000009EA 50                       push   eax
2019 000009EB B003                     mov    al, 3
2020 000009ED E8730B0000               call   _set_mode
2021 000009F2 58                       pop    eax
2022                                 cpu_except_mode_3:
2023                                   ; 02/04/2015
2024 000009F3 BB[42060000]             mov    ebx, hang
2025 000009F8 875C241C                 xchg   ebx, [esp+28]
2026                                          ; EIP (points to instruction which faults)
2027                                          ; New EIP (hang)
2028 000009FC 891D[7C050300]             mov    [FaultOffset], ebx
2029 00000A02 C744242008000000           mov    dword [esp+32], KCODE ; kernel's code segment
2030 00000A0A 814C242400020000           or     dword [esp+36], 200h ; enable interrupts (set IF)
2031                                     ;
2032 00000A12 88C4                       mov    ah, al
2033 00000A14 240F                       and    al, 0Fh
2034 00000A16 3C09                       cmp    al, 9
2035 00000A18 7602                       jna    short h1ok
2036 00000A1A 0407                       add    al, 'A'-':'
2037                                 h1ok:
2038 00000A1C C0EC04                     shr    ah, 4
2039 00000A1F 80FC09                     cmp    ah, 9
2040 00000A22 7603                       jna    short h2ok
2041 00000A24 80C407                     add    ah, 'A'-':'
2042                                 h2ok:
2043 00000A27 86E0                       xchg   ah, al
2044 00000A29 66053030                   add    ax, '00'
2045 00000A2D 66A3[A0120100]             mov    [excnstr], ax
2046                                     ;
2047                                     ; 29/08/2014
2048 00000A33 A1[7C050300]               mov    eax, [FaultOffset]
2049 00000A38 51                         push   ecx
2050 00000A39 52                         push   edx
2051 00000A3A 89E3                       mov    ebx, esp
2052                                     ; 28/08/2015
2053 00000A3C B910000000                 mov    ecx, 16        ; divisor value to convert binary number
2054                                                       ; to hexadecimal string
2055                                     ;mov   ecx, 10          ; divisor to convert
2056                                                         ; binary number to decimal string
2057                                 b2d1:
2058 00000A41 31D2                       xor    edx, edx
```

```
2059 00000A43 F7F1                      div    ecx
2060 00000A45 6652                      push   dx
2061 00000A47 39C8                      cmp    eax, ecx
2062 00000A49 73F6                      jnb    short b2d1
2063 00000A4B BF[AB120100]              mov    edi, EIPstr ; EIP value
2064                                                ; points to instruction which faults
2065                                    ; 28/08/2015
2066 00000A50 89C2                      mov    edx, eax
2067                                b2d2:
2068                                    ;add   al, '0'
2069 00000A52 8A82[1B330000]            mov    al, [edx+hexchrs]
2070 00000A58 AA                        stosb              ; write hexadecimal digit to its place
2071 00000A59 39E3                      cmp    ebx, esp
2072 00000A5B 7606                      jna    short b2d3
2073 00000A5D 6658                      pop    ax
2074 00000A5F 88C2                      mov    dl, al
2075 00000A61 EBEF                      jmp    short b2d2
2076                                b2d3:
2077 00000A63 B068                      mov    al, 'h' ; 28/08/2015
2078 00000A65 AA                        stosb
2079 00000A66 B020                      mov    al, 20h           ; space
2080 00000A68 AA                        stosb
2081 00000A69 30C0                      xor    al, al     ; to do it an ASCIIZ string
2082 00000A6B AA                        stosb
2083                                    ;
2084 00000A6C 5A                        pop    edx
2085 00000A6D 59                        pop    ecx
2086                                    ;
2087 00000A6E B44F                      mov    ah, 4Fh       ; red (4) background,
2088                                                  ; white (F) forecolor
2089 00000A70 BE[90120100]              mov    esi, exc_msg ; message offset
2090                                    ;
2091                                    ; 20/01/2017 (!cpu exception!)
2092                                    ;
2093 00000A75 8105[48100100]A000-         add    dword [scr_row], 0A0h
2094 00000A7D 0000
2095 00000A7F 8B3D[48100100]              mov    edi, [scr_row]
2096                                    ;
2097 00000A85 C605[5B030300]00          mov    byte [sysflg], 0  ; system mode
2098 00000A8C FB                          sti
2099                                    ;
2100 00000A8D E8EFFBFFFF                call   printk
2101                                    ;
2102 00000A92 B410                      mov    ah, 10h
2103 00000A94 E87D010000                call   int16h ; getc
2104                                    ;
2105 00000A99 B003                      mov    al, 3
2106 00000A9B E8C50A0000                call   _set_mode
2107                                    ;
2108 00000AA0 B801000000                mov    eax, 1
2109 00000AA5 E9BABB0000                jmp    sysexit ; terminate process !!!
2110
2111                                    ; 22/01/2017
2112                                    ; 18/04/2016
2113                                    ; 28/08/2015
2114                                    ; 23/02/2015
2115                                    ; 20/08/2014
2116                               ignore_int:
2117 00000AAA 50                        push   eax
2118 00000AAB 53                        push   ebx ; 23/02/2015
2119 00000AAC 56                        push   esi
2120 00000AAD 57                        push   edi
2121 00000AAE 1E                        push   ds
2122 00000AAF 06                        push   es
2123                                    ; 18/04/2016
2124 00000AB0 66B81000                  mov    ax, KDATA
2125 00000AB4 8ED8                      mov    ds, ax
2126 00000AB6 8EC0                      mov    es, ax
2127                                    ; 28/08/2015
2128 00000AB8 0F20D8                    mov    eax, cr3
2129 00000ABB 50                        push   eax ; (*) page directory
2130                                    ;
2131 00000ABC B467                      mov    ah, 67h       ; brown (6) background,
2132                                                  ; light gray (7) forecolor
2133 00000ABE BE[58110100]              mov    esi, int_msg ; message offset
2134                               piemsg:
2135                                      ; 27/08/2014
2136 00000AC3 8105[48100100]A000-         add     dword [scr_row], 0A0h
2137 00000ACB 0000
2138 00000ACD 8B3D[48100100]              mov     edi, [scr_row]
2139                                      ;
2140 00000AD3 E8A9FBFFFF                call   printk
2141                                    ;
2142                                    ; 23/02/2015
2143 00000AD8 B020                      mov    al, 20h  ; END OF INTERRUPT COMMAND TO
2144 00000ADA E6A0                      out    0A0h, al ; the 2nd 8259
2145                                    ; 22/08/2014
2146 00000ADC B020                      mov    al, 20h ; END OF INTERRUPT COMMAND TO 8259
2147 00000ADE E620                      out    20h, al       ; 8259 PORT
2148                               iiretp:
2149                                    ; 22/01/2017
2150                                    ; 01/09/2015
2151                                    ; 28/08/2015
2152 00000AE0 58                        pop    eax ; (*) page directory
2153 00000AE1 0F22D8                    mov    cr3, eax
2154                               iiret:
2155 00000AE4 07                        pop    es
2156 00000AE5 1F                        pop    ds
2157 00000AE6 5F                        pop    edi
2158 00000AE7 5E                        pop    esi
2159 00000AE8 5B                        pop    ebx ; 29/08/2014
2160 00000AE9 58                        pop    eax
2161 00000AEA CF                        iretd
```

```
                                        ; 23/05/2016
                                        ; 22/08/2014
                                        ; IBM PC/AT BIOS source code ----- 10/06/85 (bios.asm)
                                        ; (INT 1Ah)
                                        ;; Linux (v0.12) source code (main.c) by Linus Torvalds (1991)
                                time_of_day:
00000AEB E8EE500000                     call  UPD_IPR                   ; WAIT TILL UPDATE NOT IN PROGRESS
00000AF0 726F                             jc      short time_of_day_retn ; 23/05/2016
00000AF2 B000                           mov   al, CMOS_SECONDS
00000AF4 E800510000                     call  CMOS_READ
00000AF9 A2[90520100]                   mov   [time_seconds], al
00000AFE B002                           mov   al, CMOS_MINUTES
00000B00 E8F4500000                     call  CMOS_READ
00000B05 A2[91520100]                   mov   [time_minutes], al
00000B0A B004                           mov   al, CMOS_HOURS
00000B0C E8E8500000                     call  CMOS_READ
00000B11 A2[92520100]                     mov     [time_hours], al
00000B16 B006                           mov   al, CMOS_DAY_WEEK
00000B18 E8DC500000                     call  CMOS_READ
00000B1D A2[93520100]                   mov   [date_wday], al
00000B22 B007                           mov   al, CMOS_DAY_MONTH
00000B24 E8D0500000                     call  CMOS_READ
00000B29 A2[94520100]                   mov   [date_day], al
00000B2E B008                           mov   al, CMOS_MONTH
00000B30 E8C4500000                     call  CMOS_READ
00000B35 A2[95520100]                   mov   [date_month], al
00000B3A B009                           mov   al, CMOS_YEAR
00000B3C E8B8500000                     call  CMOS_READ
00000B41 A2[96520100]                   mov   [date_year], al
00000B46 B032                           mov   al, CMOS_CENTURY
00000B48 E8AC500000                     call  CMOS_READ
00000B4D A2[97520100]                   mov   [date_century], al
                                        ;
00000B52 B000                           mov   al, CMOS_SECONDS
00000B54 E8A0500000                     call  CMOS_READ
00000B59 3A05[90520100]                 cmp   al, [time_seconds]
00000B5F 758A                           jne   short time_of_day

                                time_of_day_retn:
00000B61 C3                             retn

                                        ; 15/01/2017
                                        ; 10/06/2016
                                        ; 07/06/2016
                                        ; 06/06/2016
                                        ; 23/05/2016
                                rtc_p:
00000B62 B101                           mov   cl, 1 ; 15/01/2017
00000B64 EB02                           jmp   short rtc_p0
                                u_timer:
                                        ; Timer Events with 18.2 Hz Timer Ticks
                                        ; (and also timer events with RTC seconds)
00000B66 28C9                           sub   cl, cl ; mov cl, 0 ; 15/01/2017
                                rtc_p0:
                                        ; 19/05/2016 - TRDOS 386 (TRDOS v2.0)
                                        ; Major Modification:
                                        ; Check and Perform Timer Events (for RTC)
                                        ; 25/08/2014 - 07/09/2014
                                        ; Retro UNIX 386 v1:
                                        ; Print Real Time Clock content

                                        ; 15/01/2017
00000B68 880D[B45F0100]                 mov   byte [priority], cl ; 0 or 1 (not 2)
00000B6E 8A2D[B75F0100]                 mov   ch, [timer_events]
00000B74 20ED                           and   ch, ch
00000B76 7420                           jz    short rtc_p3

00000B78 BE[60040300]                   mov   esi, timer_set  ; beginning address of
                                                              ; timer events space
                                rtc_p1:
00000B7D 8B06                           mov   eax, [esi]
00000B7F 20C0                           and   al, al ; 0 = free, >0 = process no.
00000B81 7416                           jz    short rtc_p4
                                        ;
00000B83 C1C810                         ror   eax, 16
                                        ; ah = response value, al = interrupt type
                                        ; 15/01/2017
                                        ; cl = interrupt source
                                        ;       1 = RTC, 0 = PIT
00000B86 38C8                           cmp   al, cl
00000B88 750A                           jne   short rtc_p2 ; not as requested or undefined !
00000B8A 3C01                           cmp   al, 1 ; 1 ; RTC interrupt ?
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[0C010300]               mov   [ebx+p.tcb-4], edi ; user's callback service addr
00000BC4 3A05[B3030300]               cmp   al, [u.uno]
00000BCA 7521                         jne   short rtc_p9
00000BCC 893D[D0030300]               mov   [u.tcb], edi
                                rtc_p7:
                                      ; 15/01/2017
00000BD2 B002                         mov   al, 2
00000BD4 A2[B45F0100]                 mov   [priority], al ; 2
                                      ; 10/01/2017
                                      ;mov   byte [u.pri], 2
00000BD9 A2[A9030300]                 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[B3030300]               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 E8F7E60000                   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)     :
                                <1>       ;                                                                       :
                                <1>       ;                     REGISTER     DELAY                                :
                                <1>       ;                      VALUE       VALUE                                :
                                <1>       ;                     ------------------                                :
                                <1>       ;                   00H        250 ms                                   :
                                <1>       ;                   01H        500 ms                                   :
                                <1>       ;                   02H        750 ms                                   :
                                <1>       ;                   03H       1000 ms                                   :
                                <1>       ;-------------------------------------------------------------------------:
                                <1>       ;     (AH)= 05H  PLACE ASCII CHARACTER/SCAN CODE COMBINATION IN KEYBOARD  :
                                <1>       ;               BUFFER AS IF STRUCK FROM KEYBOARD                        :
                                <1>       ;               ENTRY:  (CL) = ASCII CHARACTER                          :
                                <1>       ;                       (CH) = SCAN CODE                                 :
                                <1>       ;               EXIT:   (AH) = 00H = SUCCESSFUL OPERATION                :
                                <1>       ;                       (AL) = 01H = UNSUCCESSFUL - BUFFER FULL          :
                                <1>       ;               FLAGS:  CARRY IF ERROR                                   :
                                <1>       ;-------------------------------------------------------------------------:
                                <1>       ;     (AH)= 10H  EXTENDED READ INTERFACE FOR THE ENHANCED KEYBOARD,       :
                                <1>       ;               OTHERWISE SAME AS FUNCTION AH=0                          :
                                <1>       ;-------------------------------------------------------------------------:
                                <1>       ;     (AH)= 11H  EXTENDED ASCII STATUS FOR THE ENHANCED KEYBOARD,         :
```

```
                                <1>       ;               OTHERWISE SAME AS FUNCTION AH=1                          :
                                <1>       ;---------------------------------------------------------------------------:
                                <1>       ;     (AH)= 12H  RETURN THE EXTENDED SHIFT STATUS IN AX REGISTER           :
                                <1>       ;               AL = BITS FROM KB_FLAG, AH = BITS FOR LEFT AND RIGHT       :
                                <1>       ;               CTL AND ALT KEYS FROM KB_FLAG_1 AND KB_FLAG_3              :
                                <1>       ; OUTPUT                                                                 :
                                <1>       ;     AS NOTED ABOVE, ONLY (AX) AND FLAGS CHANGED                        :
                                <1>       ;     ALL REGISTERS RETAINED                                              :
                                <1>       ;---------------------------------------------------------------------------
                                <1>
                                <1> ; 15/01/2017
                                <1> ; 14/01/2017
                                <1> ; 02/01/2017
                                <1> ; 29/05/2016
                                <1> ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
                                <1> int32h:  ; Keyboard BIOS
                                <1>
                                <1> KEYBOARD_IO_1:
                                <1>       ;sti                    ; INTERRUPTS BACK ON
                                <1>       ; 29/05/2016
00000C1E 80642408BE             <1>         and     byte [esp+8], 10111110b ; clear zero flag and cary flag
                                <1>       ;
00000C23 1E                     <1>       push   ds               ; SAVE CURRENT DS
00000C24 53                     <1>       push   ebx              ; SAVE BX TEMPORARILY
                                <1>       ;push  ecx              ; SAVE CX TEMPORARILY
00000C25 66BB1000               <1>         mov     bx, KDATA
00000C29 8EDB                   <1>       mov    ds, bx           ; PUT SEGMENT VALUE OF DATA AREA INTO DS
                                <1>
                                <1>       ; 14/01/2017
00000C2B 8B1C24                 <1>       mov    ebx, [esp]
                                <1>       ;; 15/01/2017
                                <1>       ; 02/01/2017
                                <1>       ;;mov  byte [intflg], 32h  ; keyboard interrupt
00000C2E FB                     <1>       sti
                                <1>       ;
                                <1>
00000C2F 08E4                   <1>       or     ah, ah           ; CHECK FOR (AH)= 00H
00000C31 743A                   <1>       jz     short _K1        ; ASCII_READ
00000C33 FECC                   <1>       dec    ah                  ; CHECK FOR (AH)= 01H
00000C35 7453                   <1>         jz      short _K2          ; ASCII_STATUS
00000C37 FECC                   <1>       dec    ah               ; CHECK FOR (AH)= 02H
00000C39 0F8494000000           <1>         jz      _K3                ; SHIFT STATUS
00000C3F FECC                   <1>       dec    ah               ; CHECK FOR (AH)= 03H
00000C41 0F8493000000           <1>         jz      _K300              ; SET TYPAMATIC RATE/DELAY
00000C47 80EC02                 <1>       sub    ah, 2            ; CHECK FOR (AH)= 05H
00000C4A 0F84BC000000           <1>         jz      _K500              ; KEYBOARD WRITE
                                <1> _KIO1:
00000C50 80EC0B                 <1>       sub    ah, 11           ; AH =  10H
00000C53 740C                   <1>       jz     short _K1E       ; EXTENDED ASCII READ
00000C55 FECC                   <1>       dec    ah               ; CHECK FOR (AH)= 11H
00000C57 7422                   <1>       jz     short _K2E       ; EXTENDED_ASCII_STATUS
00000C59 FECC                   <1>       dec    ah               ; CHECK FOR (AH)= 12H
00000C5B 7458                   <1>       jz     short _K3E       ; EXTENDED_SHIFT_STATUS
                                <1> _KIO_EXIT:
                                <1>       ; 02/01/2017
00000C5D FA                     <1>       cli
                                <1>       ;;mov  byte [intflg], 0 ;; 15/01/2017
                                <1>       ;
                                <1>       ;pop   ecx              ; RECOVER REGISTER
00000C5E 5B                     <1>       pop    ebx              ; RECOVER REGISTER
00000C5F 1F                     <1>       pop    ds               ; RECOVER SEGMENT
00000C60 CF                     <1>       iretd                   ; INVALID COMMAND, EXIT
                                <1>
                                <1>       ;----- ASCII CHARACTER
                                <1> _K1E:
00000C61 E8D3000000             <1>       call   _K1S             ; GET A CHARACTER FROM THE BUFFER (EXTENDED)
00000C66 E848010000             <1>       call   _KIO_E_XLAT      ; ROUTINE TO XLATE FOR EXTENDED CALLS
00000C6B EBF0                   <1>       jmp    short _KIO_EXIT     ; GIVE IT TO THE CALLER
                                <1> _K1:
00000C6D E8C7000000             <1>       call   _K1S             ; GET A CHARACTER FROM THE BUFFER
00000C72 E847010000             <1>       call   _KIO_S_XLAT      ; ROUTINE TO XLATE FOR STANDARD CALLS
00000C77 72F4                   <1>       jc     short _K1        ; CARRY SET MEANS TROW CODE AWAY
                                <1> _K1A:
00000C79 EBE2                   <1>       jmp    short _KIO_EXIT     ; RETURN TO CALLER
                                <1>
                                <1>       ;----- ASCII STATUS
                                <1> _K2E:
00000C7B E804010000             <1>       call   _K2S             ; TEST FOR CHARACTER IN BUFFER (EXTENDED)
00000C80 7420                   <1>       jz     short _K2B       ; RETURN IF BUFFER EMPTY
00000C82 9C                     <1>       pushf                   ; SAVE ZF FROM TEST
00000C83 E82B010000             <1>       call   _KIO_E_XLAT      ; ROUTINE TO XLATE FOR EXTENDED CALLS
00000C88 EB17                   <1>       jmp    short _K2A       ; GIVE IT TO THE CALLER
                                <1> _K2:
00000C8A E8F5000000             <1>       call   _K2S             ; TEST FOR CHARACTER IN BUFFER
00000C8F 7411                   <1>       jz     short _K2B       ; RETURN IF BUFFER EMPTY
00000C91 9C                     <1>       pushf                   ; SAVE ZF FROM TEST
00000C92 E827010000             <1>       call   _KIO_S_XLAT      ; ROUTINE TO XLATE FOR STANDARD CALLS
00000C97 7308                   <1>       jnc    short _K2A       ; CARRY CLEAR MEANS PASS VALID CODE
00000C99 9D                     <1>       popf                    ; INVALID CODE FOR THIS TYPE OF CALL
00000C9A E89A000000             <1>       call   _K1S             ; THROW THE CHARACTER AWAY
00000C9F EBE9                   <1>       jmp    short _K2        ; GO LOOK FOR NEXT CHAR, IF ANY
                                <1> _K2A:
00000CA1 9D                     <1>       popf                    ; RESTORE ZF FROM TEST
                                <1> _K2B:
                                <1>       ; 02/01/2017
00000CA2 FA                     <1>       cli
                                <1>       ;; mov byte [intflg], 0 ;; 15/01/2017
                                <1>       ;
                                <1>       ;pop   ecx              ; RECOVER REGISTER
00000CA3 5B                     <1>       pop    ebx              ; RECOVER REGISTER
00000CA4 1F                     <1>       pop    ds               ; RECOVER SEGMENT
                                <1>       ; (*) 29/05/2016
                                <1>       ; (*) retf 4            ; THROW AWAY (e)FLAGS
```

```
2574 00000CA5 7208              <1>       jc    short _k2d
2575 00000CA7 7505              <1>       jnz   short _k2c
2576 00000CA9 804C240840        <1>       or    byte [esp+8], 01000000b   ; set zero flag bit of eflags register
2577                            <1> _k2c:
2578 00000CAE CF                <1>       iretd
2579                            <1> _k2d:
2580                            <1>       ; 29/05/2016 -set carry flag on stack-
2581                            <1>       ; [esp] = EIP
2582                            <1>       ; [esp+4] = CS
2583                            <1>       ; [esp+8] = E-FLAGS
2584 00000CAF 804C240801        <1>       or    byte [esp+8], 1  ; set carry bit of eflags register
2585                            <1>       ; [esp+12] = ESP (user)
2586                            <1>       ; [esp+16] = SS (User)
2587 00000CB4 CF                <1>       iretd
2588                            <1>
2589                            <1>
2590                            <1>       ; (*) 29/05/2016 - 'ref 4' intruction causes to stack fault
2591                            <1>       ; (OUTER-PRIVILEGE-LEVEL)
2592                            <1>       ; INTEL 80386 PROGRAMMER'S REFERENCE MANUAL 1986
2593                            <1>       ; // RETF instruction:
2594                            <1>       ;
2595                            <1>       ; IF OperandMode=32 THEN
2596                            <1>       ;    Load CS:EIP from stack;
2597                            <1>       ;    Set CS RPL to CPL;
2598                            <1>       ;    Increment eSP by 8 plus the immediate offset if it exists;
2599                            <1>       ;    Load SS:eSP from stack;
2600                            <1>       ; ELSE (* OperandMode=16 *)
2601                            <1>       ;    Load CS:IP from stack;
2602                            <1>       ;    Set CS RPL to CPL;
2603                            <1>       ;    Increment eSP by 4 plus the immediate offset if it exists;
2604                            <1>       ;    Load SS:eSP from stack;
2605                            <1>       ; FI;
2606                            <1>       ;
2607                            <1>       ; //
2608                            <1>
2609                            <1>       ;----- SHIFT STATUS
2610                            <1> _K3E:                                  ; GET THE EXTENDED SHIFT STATUS FLAGS
2611 00000CB5 8A25[8E5E0000]   <1>       mov   ah, [KB_FLAG_1]           ; GET SYSTEM SHIFT KEY STATUS
2612 00000CBB 80E404           <1>       and   ah, SYS_SHIFT       ; MASK ALL BUT SYS KEY BIT
2613                            <1>       ;mov  cl, 5               ; SHIFT THEW SYSTEMKEY BIT OVER TO
2614                            <1>       ;shl  ah, cl              ; BIT 7 POSITION
2615 00000CBE C0E405           <1>         shl ah, 5
2616 00000CC1 A0[8E5E0000]     <1>       mov   al, [KB_FLAG_1]           ; GET SYSTEM SHIFT STATES BACK
2617 00000CC6 2473             <1>       and   al, 01110011b       ; ELIMINATE SYS SHIFT, HOLD_STATE AND INS_SHIFT
2618 00000CC8 08C4             <1>       or    ah, al                 ; MERGE REMAINING BITS INTO AH
2619 00000CCA A0[905E0000]     <1>       mov   al, [KB_FLAG_3]           ; GET RIGHT CTL AND ALT
2620 00000CCF 240C             <1>       and   al, 00001100b       ; ELIMINATE LC_E0 AND LC_E1
2621 00000CD1 08C4             <1>       or    ah, al              ; OR THE SHIFT FLAGS TOGETHER
2622                            <1> _K3:
2623 00000CD3 A0[8D5E0000]     <1>       mov   al, [KB_FLAG]       ; GET THE SHIFT STATUS FLAGS
2624                            <1>       ;jmp  short _KIO_EXIT           ; RETURN TO CALLER
2625 00000CD8 EB83             <1>       jmp   _KIO_EXIT
2626                            <1>
2627                            <1>       ;----- SET TYPAMATIC RATE AND DELAY
2628                            <1> _K300:
2629 00000CDA 3C05             <1>       cmp   al, 5               ; CORRECT FUNCTION CALL?
2630                            <1>       ;jne  short _KIO_EXIT           ; NO, RETURN
2631 00000CDC 0F857BFFFFFF     <1>       jne   _KIO_EXIT
2632 00000CE2 F6C3E0           <1>       test  bl, 0E0h            ; TEST FOR OUT-OF-RANGE RATE
2633 00000CE5 0F8572FFFFFF     <1>         jnz     _KIO_EXIT             ; RETURN IF SO
2634 00000CEB F6C7FC           <1>       test  BH, 0FCh            ; TEST FOR OUT-OF-RANGE DELAY
2635 00000CEE 0F8569FFFFFF     <1>         jnz     _KIO_EXIT             ; RETURN IF SO
2636 00000CF4 B0F3             <1>       mov   al, KB_TYPA_RD            ; COMMAND FOR TYPAMATIC RATE/DELAY
2637 00000CF6 E8DA060000       <1>       call  SND_DATA            ; SEND TO KEYBOARD
2638                            <1>       ;mov  cx, 5               ; SHIFT COUNT
2639                            <1>       ;shl  bh, cl              ; SHIFT DELAY OVER
2640 00000CFB C0E705           <1>       shl   bh, 5
2641 00000CFE 88D8             <1>       mov   al, bl              ; PUT IN RATE
2642 00000D00 08F8             <1>       or    al, bh              ; AND DELAY
2643 00000D02 E8CE060000       <1>       call  SND_DATA            ; SEND TO KEYBOARD
2644 00000D07 E951FFFFFF       <1>         jmp     _KIO_EXIT             ; RETURN TO CALLER
2645                            <1>
2646                            <1>       ;----- WRITE TO KEYBOARD BUFFER
2647                            <1> _K500:
2648 00000D0C 56               <1>       push  esi                 ; SAVE SI (esi)
2649 00000D0D FA               <1>       cli                       ;
2650 00000D0E 8B1D[9E5E0000]   <1>       mov   ebx, [BUFFER_TAIL]  ; GET THE 'IN TO' POINTER TO THE BUFFER
2651 00000D14 89DE             <1>       mov   esi, ebx            ; SAVE A COPY IN CASE BUFFER NOT FULL
2652 00000D16 E8D3000000       <1>       call  _K4                 ; BUMP THE POINTER TO SEE IF BUFFER IS FULL
2653 00000D1B 3B1D[9A5E0000]   <1>       cmp   ebx, [BUFFER_HEAD]  ; WILL THE BUFFER OVERRUN IF WE STORE THIS?
2654 00000D21 740D             <1>       je    short _K502         ; YES - INFORM CALLER OF ERROR
2655 00000D23 66890E           <1>       mov   [esi], cx           ; NO - PUT ASCII/SCAN CODE INTO BUFFER
2656 00000D26 891D[9E5E0000]   <1>       mov   [BUFFER_TAIL], ebx  ; ADJUST 'IN TO' POINTER TO REFLECT CHANGE
2657 00000D2C 28C0             <1>       sub   al, al              ; TELL CALLER THAT OPERATION WAS SUCCESSFUL
2658 00000D2E EB02             <1>       jmp   short _K504         ; SUB INSTRUCTION ALSO RESETS CARRY FLAG
2659                            <1> _K502:
2660 00000D30 B001             <1>       mov   al, 01h                   ; BUFFER FULL INDICATION
2661                            <1> _K504:
2662 00000D32 FB               <1>       sti
2663 00000D33 5E               <1>       pop   esi                 ; RECOVER SI (esi)
2664 00000D34 E924FFFFFF       <1>         jmp     _KIO_EXIT             ; RETURN TO CALLER WITH STATUS IN AL
2665                            <1>
2666                            <1>       ;----- READ THE KEY TO FIGURE OUT WHAT TO DO -----
2667                            <1> _K1S:
2668 00000D39 FA               <1>       cli   ; 03/12/2014
2669 00000D3A 8B1D[9A5E0000]   <1>         mov     ebx, [BUFFER_HEAD]    ; GET POINTER TO HEAD OF BUFFER
2670 00000D40 3B1D[9E5E0000]   <1>         cmp     ebx, [BUFFER_TAIL]    ; TEST END OF BUFFER
2671                            <1>       ;jne  short _K1U          ; IF ANYTHING IN BUFFER SKIP INTERRUPT
2672 00000D46 750F             <1>       jne   short _k1x ; 03/12/2014
2673                            <1>       ;
2674                            <1>       ; 03/12/2014
2675                            <1>       ; 28/08/2014
2676                            <1>       ; PERFORM OTHER FUNCTION ?? here !
```

```
                        <1>       ;; MOV AX, 9002h          ; MOVE IN WAIT CODE & TYPE
                        <1>       ;; INT       15H                ; PERFORM OTHER FUNCTION
                        <1> _K1T:                                 ; ASCII READ
00000D48 FB             <1>       sti                     ; INTERRUPTS BACK ON DURING LOOP
00000D49 90             <1>       nop                     ; ALLOW AN INTERRUPT TO OCCUR
                        <1> _K1U:
00000D4A FA             <1>       cli                     ; INTERRUPTS BACK OFF
00000D4B 8B1D[9A5E0000] <1>         mov        ebx, [BUFFER_HEAD]  ; GET POINTER TO HEAD OF BUFFER
00000D51 3B1D[9E5E0000] <1>         cmp     ebx, [BUFFER_TAIL]     ; TEST END OF BUFFER
                        <1> _k1x:
00000D57 53             <1>       push   ebx                ; SAVE ADDRESS
00000D58 9C             <1>       pushf                     ; SAVE FLAGS
00000D59 E82F070000     <1>       call   MAKE_LED           ; GO GET MODE INDICATOR DATA BYTE
00000D5E 8A1D[8F5E0000] <1>       mov    bl, [KB_FLAG_2]    ; GET PREVIOUS BITS
00000D64 30C3           <1>       xor    bl, al             ; SEE IF ANY DIFFERENT
00000D66 80E307         <1>       and    bl, 07h      ; KB_LEDS    ; ISOLATE INDICATOR BITS
00000D69 7406           <1>       jz     short _K1V         ; IF NO CHANGE BYPASS UPDATE
00000D6B E8C9060000     <1>       call   SND_LED1
00000D70 FA             <1>       cli                       ; DISABLE INTERRUPTS
                        <1> _K1V:
00000D71 9D             <1>       popf                      ; RESTORE FLAGS
00000D72 5B             <1>       pop    ebx                ; RESTORE ADDRESS
00000D73 74D3           <1>         je      short _K1T              ; LOOP UNTIL SOMETHING IN BUFFER
                        <1>       ;
00000D75 668B03         <1>       mov    ax, [ebx]          ; GET SCAN CODE AND ASCII CODE
00000D78 E871000000     <1>         call    _K4                     ; MOVE POINTER TO NEXT POSITION
00000D7D 891D[9A5E0000] <1>         mov     [BUFFER_HEAD], ebx      ; STORE VALUE IN VARIABLE
00000D83 C3             <1>       retn                      ; RETURN
                        <1>
                        <1>       ;----- READ THE KEY TO SEE IF ONE IS PRESENT -----
                        <1> _K2S:
00000D84 FA             <1>       cli                       ; INTERRUPTS OFF
00000D85 8B1D[9A5E0000] <1>         mov     ebx, [BUFFER_HEAD]      ; GET HEAD POINTER
00000D8B 3B1D[9E5E0000] <1>         cmp     ebx, [BUFFER_TAIL]      ; IF EQUAL (Z=1) THEN NOTHING THERE
00000D91 668B03         <1>       mov    ax, [ebx]
00000D94 9C             <1>       pushf                     ; SAVE FLAGS
00000D95 6650           <1>       push   ax                 ; SAVE CODE
00000D97 E8F1060000     <1>       call   MAKE_LED           ; GO GET MODE INDICATOR DATA BYTE
00000D9C 8A1D[8F5E0000] <1>       mov    bl, [KB_FLAG_2]    ; GET PREVIOUS BITS
00000DA2 30C3           <1>       xor    bl, al             ; SEE IF ANY DIFFERENT
00000DA4 80E307         <1>       and    bl, 07h ; KB_LEDS  ; ISOLATE INDICATOR BITS
00000DA7 7405           <1>       jz     short _K2T         ; IF NO CHANGE BYPASS UPDATE
00000DA9 E874060000     <1>       call   SND_LED                  ; GO TURN ON MODE INDICATORS
                        <1> _K2T:
00000DAE 6658           <1>       pop    ax                 ; RESTORE CODE
00000DB0 9D             <1>       popf                      ; RESTORE FLAGS
00000DB1 FB             <1>       sti                       ; INTERRUPTS BACK ON
00000DB2 C3             <1>       retn                      ; RETURN
                        <1>
                        <1>       ;----- ROUTINE TO TRANSLATE SCAN CODE PAIRS FOR EXTENDED CALLS -----
                        <1> _KIO_E_XLAT:
00000DB3 3CF0           <1>       cmp    al, 0F0h           ; IS IT ONE OF THE FILL-INs?
00000DB5 7506           <1>       jne    short _KIO_E_RET   ; NO, PASS IT ON
00000DB7 08E4           <1>         or   ah, ah             ; AH = 0 IS SPECIAL CASE
00000DB9 7402           <1>         jz   short _KIO_E_RET       ; PASS THIS ON UNCHANGED
00000DBB 30C0           <1>       xor    al, al             ; OTHERWISE SET AL = 0
                        <1> _KIO_E_RET:
00000DBD C3             <1>       retn                      ; GO BACK
                        <1>
                        <1>       ;----- ROUTINE TO TRANSLATE SCAN CODE PAIRS FOR STANDARD CALLS -----
                        <1> _KIO_S_XLAT:
00000DBE 80FCE0         <1>       cmp    ah, 0E0h           ; IS IT KEYPAD ENTER OR / ?
00000DC1 750F           <1>       jne    short _KIO_S2      ; NO, CONTINUE
00000DC3 3C0D           <1>       cmp    al, 0Dh                  ; KEYPAD ENTER CODE?
00000DC5 7408           <1>         je   short _KIO_S1      ; YES, MASSAGE A BIT
00000DC7 3C0A           <1>       cmp    al, 0Ah                  ; CTRL KEYPAD ENTER CODE?
00000DC9 7404           <1>         je   short _KIO_S1      ; YES, MASSAGE THE SAME
00000DCB B435           <1>       mov    ah, 35h                  ; NO, MUST BE KEYPAD /
                        <1> _kio_ret: ; 03/12/2014
00000DCD F8             <1>       clc
00000DCE C3             <1>       retn
                        <1>       ;jmp   short _KIO_USE            ; GIVE TO CALLER
                        <1> _KIO_S1:
00000DCF B41C           <1>       mov    ah, 1Ch                  ; CONVERT TO COMPATIBLE OUTPUT
                        <1>       ;jmp   short _KIO_USE            ; GIVE TO CALLER
00000DD1 C3             <1>       retn
                        <1> _KIO_S2:
00000DD2 80FC84         <1>       cmp    ah, 84h                  ; IS IT ONE OF EXTENDED ONES?
00000DD5 7715           <1>       ja     short _KIO_DIS           ; YES, THROW AWAY AND GET ANOTHER CHAR
00000DD7 3CF0           <1>       cmp    al, 0F0h           ; IS IT ONE OF THE FILL-INs?
00000DD9 7506           <1>         jne  short _KIO_S3      ; NO, TRY LAST TEST
00000DDB 08E4           <1>       or     ah, ah             ; AH = 0 IS SPECIAL CASE
00000DDD 740C           <1>         jz   short _KIO_USE           ; PASS THIS ON UNCHANGED
00000DDF EB0B           <1>       jmp    short _KIO_DIS           ; THROW AWAY THE REST
                        <1> _KIO_S3:
00000DE1 3CE0           <1>       cmp    al, 0E0h           ; IS IT AN EXTENSION OF A PREVIOUS ONE?
                        <1>       ;jne   short _KIO_USE           ; NO, MUST BE A STANDARD CODE
00000DE3 75E8           <1>       jne    short _kio_ret
00000DE5 08E4           <1>       or     ah, ah             ; AH = 0 IS SPECIAL CASE
00000DE7 7402           <1>         jz   short _KIO_USE           ; JUMP IF AH = 0
00000DE9 30C0           <1>       xor    al, al             ; CONVERT TO COMPATIBLE OUTPUT
                        <1>       ;jmp   short _KIO_USE           ; PASS IT ON TO CALLER
                        <1> _KIO_USE:
                        <1>       ;clc                      ; CLEAR CARRY TO INDICATE GOOD CODE
00000DEB C3             <1>       retn                      ; RETURN
                        <1> _KIO_DIS:
00000DEC F9             <1>       stc                       ; SET CARRY TO INDICATE DISCARD CODE
00000DED C3             <1>       retn                      ; RETURN
                        <1>
                        <1>       ;----- INCREMENT BUFFER POINTER ROUTINE -----
                        <1> _K4:
00000DEE 43             <1>       inc     ebx
00000DEF 43             <1>       inc    ebx                ; MOVE TO NEXT WORD IN LIST
```

```
2780 00000DF0 3B1D[965E0000]      <1>         cmp     ebx, [BUFFER_END]      ; AT END OF BUFFER?
2781                              <1>         ;jne    short _K5               ; NO, CONTINUE
2782 00000DF6 7206                <1>       jb     short _K5
2783 00000DF8 8B1D[925E0000]      <1>         mov     ebx, [BUFFER_START]     ; YES, RESET TO BUFFER BEGINNING
2784                              <1> _K5:
2785 00000DFE C3                  <1>       retn
2786                              <1>
2787                              <1> ; 20/02/2015
2788                              <1> ; 05/12/2014
2789                              <1> ; 26/08/2014
2790                              <1> ; KEYBOARD (HARDWARE) INTERRUPT -  IRQ LEVEL 1
2791                              <1> ; (INT_09h - Retro UNIX 8086 v1 - U9.ASM, 07/03/2014)
2792                              <1> ;
2793                              <1> ; Derived from "KB_INT_1" procedure of IBM "pc-at"
2794                              <1> ; rombios source code (06/10/1985)
2795                              <1> ; 'keybd.asm', HARDWARE INT 09h - (IRQ Level 1)
2796                              <1>
2797                              <1> ; EQUATES (IBM PC-XT-286 BIOS, 1986, 'POSQEQU.INC')
2798                              <1>
2799                              <1> ;--------- 8042 COMMANDS -----------------------------------------------------
2800                              <1> ENA_KBD           equ   0AEh          ; ENABLE KEYBOARD COMMAND
2801                              <1> DIS_KBD           equ   0ADh          ; DISABLE KEYBOARD COMMAND
2802                              <1> SHUT_CMD    equ   0FEh          ; CAUSE A SHUTDOWN COMMAND
2803                              <1> ;--------- 8042 KEYBOARD INTERFACE AND DIAGNOSTIC CONTROL REGISTERS ------------
2804                              <1> STATUS_PORT equ   064h          ; 8042 STATUS PORT
2805                              <1> INPT_BUF_FULL     equ   00000010b     ; 1 = +INPUT BUFFER FULL
2806                              <1> PORT_A            equ   060h          ; 8042 KEYBOARD SCAN CODE/CONTROL PORT
2807                              <1> ;---------- 8042 KEYBOARD RESPONSE ---------------------------------------------
2808                              <1> KB_ACK            equ   0FAh          ; ACKNOWLEDGE PROM TRANSMISSION
2809                              <1> KB_RESEND   equ   0FEh          ; RESEND REQUEST
2810                              <1> KB_OVER_RUN equ   0FFh          ; OVER RUN SCAN CODE
2811                              <1> ;---------- KEYBOARD/LED COMMANDS ----------------------------------------------
2812                              <1> KB_ENABLE   equ   0F4h          ; KEYBOARD ENABLE
2813                              <1> LED_CMD           equ   0EDh          ; LED WRITE COMMAND
2814                              <1> KB_TYPA_RD  equ   0F3h          ; TYPAMATIC RATE/DELAY COMMAND
2815                              <1> ;---------- KEYBOARD SCAN CODES ------------------------------------------------
2816                              <1> NUM_KEY           equ   69            ; SCAN CODE FOR     NUMBER LOCK KEY
2817                              <1> SCROLL_KEY  equ   70            ; SCAN CODE FOR     SCROLL LOCK KEY
2818                              <1> ALT_KEY           equ   56            ; SCAN CODE FOR     ALTERNATE SHIFT KEY
2819                              <1> CTL_KEY           equ   29            ; SCAN CODE FOR     CONTROL KEY
2820                              <1> CAPS_KEY    equ   58            ; SCAN CODE FOR     SHIFT LOCK KEY
2821                              <1> DEL_KEY           equ   83            ; SCAN CODE FOR     DELETE KEY
2822                              <1> INS_KEY           equ   82            ; SCAN CODE FOR     INSERT KEY
2823                              <1> LEFT_KEY    equ   42            ; SCAN CODE FOR     LEFT SHIFT
2824                              <1> RIGHT_KEY   equ   54            ; SCAN CODE FOR     RIGHT SHIFT
2825                              <1> SYS_KEY           equ   84            ; SCAN CODE FOR     SYSTEM KEY
2826                              <1> ;---------- ENHANCED KEYBOARD SCAN CODES ----------------------------------------
2827                              <1> ID_1        equ   0ABh          ; 1ST ID CHARACTER FOR KBX
2828                              <1> ID_2        equ   041h          ; 2ND ID CHARACTER FOR KBX
2829                              <1> ID_2A       equ   054h          ; ALTERNATE 2ND ID CHARACTER FOR KBX
2830                              <1> F11_M       equ   87            ; F11 KEY MAKE
2831                              <1> F12_M       equ   88            ; F12 KEY MAKE
2832                              <1> MC_E0       equ   224           ; GENERAL MARKER CODE
2833                              <1> MC_E1       equ   225           ; PAUSE KEY MARKER CODE
2834                              <1> ;---------- FLAG EQUATES WITHIN @KB_FLAG----------------------------------------
2835                              <1> RIGHT_SHIFT  equ   00000001b    ; RIGHT SHIFT KEY DEPRESSED
2836                              <1> LEFT_SHIFT   equ   00000010b    ; LEFT SHIFT KEY DEPRESSED
2837                              <1> CTL_SHIFT    equ   00000100b    ; CONTROL SHIFT KEY DEPRESSED
2838                              <1> ALT_SHIFT    equ   00001000b    ; ALTERNATE SHIFT KEY DEPRESSED
2839                              <1> SCROLL_STATE equ   00010000b    ; SCROLL LOCK STATE IS ACTIVE
2840                              <1> NUM_STATE    equ   00100000b    ; NUM LOCK STATE IS ACTIVE
2841                              <1> CAPS_STATE   equ   01000000b    ; CAPS LOCK STATE IS ACTIVE
2842                              <1> INS_STATE    equ   10000000b    ; INSERT STATE IS ACTIVE
2843                              <1> ;---------- FLAG EQUATES WITHIN @KB_FLAG_1 -------------------------------------
2844                              <1> L_CTL_SHIFT  equ   00000001b    ; LEFT CTL KEY DOWN
2845                              <1> L_ALT_SHIFT  equ   00000010b    ; LEFT ALT KEY DOWN
2846                              <1> SYS_SHIFT    equ   00000100b    ; SYSTEM KEY DEPRESSED AND HELD
2847                              <1> HOLD_STATE   equ   00001000b    ; SUSPEND KEY HAS BEEN TOGGLED
2848                              <1> SCROLL_SHIFT equ   00010000b    ; SCROLL LOCK KEY IS DEPRESSED
2849                              <1> NUM_SHIFT    equ   00100000b    ; NUM LOCK KEY IS DEPRESSED
2850                              <1> CAPS_SHIFT   equ   01000000b    ; CAPS LOCK KEY IS DEPRE55ED
2851                              <1> INS_SHIFT    equ   10000000b    ; INSERT KEY IS DEPRESSED
2852                              <1> ;---------- FLAGS EQUATES WITHIN @KB_FLAG_2 -------------------------------------
2853                              <1> KB_LEDS           equ   00000111b    ; KEYBOARD LED STATE BITS
2854                              <1> ;           equ   00000001b    ; SCROLL LOCK INDICATOR
2855                              <1> ;           equ   00000010b    ; NUM LOCK INDICATOR
2856                              <1> ;           equ   00000100b    ; CAPS LOCK INDICATOR
2857                              <1> ;           equ   00001000b    ; RESERVED (MUST BE ZERO)
2858                              <1> KB_FA       equ   00010000b    ; ACKNOWLEDGMENT RECEIVED
2859                              <1> KB_FE       equ   00100000b    ; RESEND RECEIVED FLAG
2860                              <1> KB_PR_LED   equ   01000000b    ; MODE INDICATOR UPDATE
2861                              <1> KB_ERR            equ   10000000b    ; KEYBOARD TRANSMIT ERROR FLAG
2862                              <1> ;----------- FLAGS EQUATES WITHIN @KB_FLAG_3 -----------------------------------
2863                              <1> LC_E1       equ   00000001b    ; LAST CODE WAS THE E1 HIDDEN CODE
2864                              <1> LC_E0       equ   00000010b    ; LAST CODE WAS THE E0 HIDDEN CODE
2865                              <1> R_CTL_SHIFT  equ   00000100b    ; RIGHT CTL KEY DOWN
2866                              <1> R_ALT_SHIFT  equ   00001000b    ; RIGHT ALT KEY DOWN
2867                              <1> GRAPH_ON     equ   00001000b    ; ALT GRAPHICS KEY DOWN (WT ONLY)
2868                              <1> KBX          equ   00010000b    ; ENHANCED KEYBOARD INSTALLED
2869                              <1> SET_NUM_LK   equ   00100000b    ; FORCE NUM LOCK IF READ ID AND KBX
2870                              <1> LC_AB        equ   01000000b    ; LAST CHARACTER WAS FIRST ID CHARACTER
2871                              <1> RD_ID        equ   10000000b    ; DOING A READ ID (MUST BE BIT0)
2872                              <1> ;
2873                              <1> ;----------- INTERRUPT EQUATES -------------------------------------------------
2874                              <1> EOI          equ   020h         ; END OF INTERRUPT COMMAND TO 8259
2875                              <1> INTA00            equ   020h         ; 8259 PORT
2876                              <1>
2877                              <1>
2878                              <1> kb_int:
2879                              <1>
2880                              <1> ; 17/10/2015 ('ctrlbrk')
2881                              <1> ; 05/12/2014
2882                              <1> ; 04/12/2014 (derived from pc-xt-286 bios source code -1986-)
```

```
2883                            <1> ; 26/08/2014
2884                            <1> ;
2885                            <1> ; 03/06/86  KEYBOARD BIOS
2886                            <1> ;
2887                            <1> ;--- HARDWARE INT 09H -- (IRQ LEVEL 1) ----------------------------------------
2888                            <1> ;                                                                           ;
2889                            <1> ;     KEYBOARD INTERRUPT ROUTINE                                            ;
2890                            <1> ;                                                                           ;
2891                            <1> ;-------------------------------------------------------------------------------
2892                            <1>
2893                            <1> KB_INT_1:
2894 00000DFF FB               <1>       sti                     ; ENABLE INTERRUPTS
2895                            <1>       ;push  ebp
2896 00000E00 50               <1>       push   eax
2897 00000E01 53               <1>       push   ebx
2898 00000E02 51               <1>       push   ecx
2899 00000E03 52               <1>       push   edx
2900 00000E04 56               <1>       push   esi
2901 00000E05 57               <1>       push   edi
2902 00000E06 1E               <1>       push   ds
2903 00000E07 06               <1>       push   es
2904 00000E08 FC               <1>       cld                     ; FORWARD DIRECTION
2905 00000E09 66B81000         <1>       mov    ax, KDATA
2906 00000E0D 8ED8             <1>       mov    ds, ax
2907 00000E0F 8EC0             <1>       mov    es, ax
2908                            <1>       ;
2909                            <1>       ;-----WAIT FOR KEYBOARD DISABLE COMMAND TO BE ACCEPTED
2910 00000E11 B0AD             <1>       mov    al, DIS_KBD      ; DISABLE THE KEYBOARD COMMAND
2911 00000E13 E8A9050000       <1>       call   SHIP_IT                ; EXECUTE DISABLE
2912 00000E18 FA               <1>       cli                     ; DISABLE INTERRUPTS
2913 00000E19 B900000100       <1>       mov    ecx, 10000h      ; SET MAXIMUM TIMEOUT
2914                            <1> KB_INT_01:
2915 00000E1E E464             <1>       in     al, STATUS_PORT          ; READ ADAPTER STATUS
2916 00000E20 A802             <1>       test   al, INPT_BUF_FULL   ; CHECK INPUT BUFFER FULL STATUS BIT
2917 00000E22 E0FA             <1>       loopnz KB_INT_01           ; WAIT FOR COMMAND TO BE ACCEPTED
2918                            <1>       ;
2919                            <1>       ;-----READ CHARACTER FROM KEYBOARD INTERFACE
2920 00000E24 E460             <1>       in     al, PORT_A          ; READ IN THE CHARACTER
2921                            <1>       ;
2922                            <1>       ;-----SYSTEM HOOK INT 15H - FUNCTION 4FH (ON HARDWARE INT LEVEL 9H)
2923                            <1>       ;MOV   AH, 04FH            ; SYSTEM INTERCEPT - KEY CODE FUNCTION
2924                            <1>       ;STC                       ; SET CY=1 (IN CASE OF IRET)
2925                            <1>       ;INT   15H                 ; CASETTE CALL (AL)=KEY SCAN CODE
2926                            <1>       ;                          ; RETURNS CY=1 FOR INVALID FUNCTION
2927                            <1>       ;JC    KB_INT_02           ; CONTINUE IF CARRY FLAG SET ((AL)=CODE)
2928                            <1>       ;JMP   K26                 ; EXIT IF SYSTEM HANDLES SCAN CODE
2929                            <1>       ;                          ; EXIT HANDLES HARDWARE EOI AND ENABLE
2930                            <1>       ;
2931                            <1>       ;-----CHECK FOR A RESEND COMMAND TO KEYBOARD
2932                            <1> KB_INT_02:                       ;        (AL)= SCAN CODE
2933 00000E26 FB               <1>       sti                     ; ENABLE INTERRUPTS AGAIN
2934 00000E27 3CFE             <1>       cmp    al, KB_RESEND       ; IS THE INPUT A RESEND
2935 00000E29 7411             <1>         je     short KB_INT_4          ; GO IF RESEND
2936                            <1>       ;
2937                            <1>       ;-----CHECK FOR RESPONSE TO A COMMAND TO KEYBOARD
2938 00000E2B 3CFA             <1>       cmp    al, KB_ACK          ; IS THE INPUT AN ACKNOWLEDGE
2939 00000E2D 751A             <1>         jne    short KB_INT_2          ; GO IF NOT
2940                            <1>       ;
2941                            <1>       ;-----A COMMAND TO THE KEYBOARD WAS ISSUED
2942 00000E2F FA               <1>       cli                     ; DISABLE INTERRUPTS
2943 00000E30 800D[8F5E0000]10 <1>       or     byte [KB_FLAG_2], KB_FA ; INDICATE ACK RECEIVED
2944 00000E37 E97A020000       <1>         jmp    K26                   ; RETURN IF NOT (ACK RETURNED FOR DATA)
2945                            <1>       ;
2946                            <1>       ;-----RESEND THE LAST BYTE
2947                            <1> KB_INT_4:
2948 00000E3C FA               <1>       cli                     ; DISABLE INTERRUPTS
2949 00000E3D 800D[8F5E0000]20 <1>       or     byte [KB_FLAG_2], KB_FE ; INDICATE RESEND RECEIVED
2950 00000E44 E96D020000       <1>         jmp    K26                   ; RETURN IF NOT ACK RETURNED FOR DATA)
2951                            <1>       ;
2952                            <1> ;-----       UPDATE MODE INDICATORS IF CHANGE IN STATE
2953                            <1> KB_INT_2:
2954 00000E49 6650             <1>       push   ax                  ; SAVE DATA IN
2955 00000E4B E83D060000       <1>       call   MAKE_LED            ; GO GET MODE INDICATOR DATA BYTE
2956 00000E50 8A1D[8F5E0000]   <1>       mov    bl, [KB_FLAG_2]     ; GET PREVIOUS BITS
2957 00000E56 30C3             <1>       xor    bl, al              ; SEE IF ANY DIFFERENT
2958 00000E58 80E307           <1>       and    bl, KB_LEDS         ; ISOLATE INDICATOR BITS
2959 00000E5B 7405             <1>       jz     short UP0           ; IF NO CHANGE BYPASS UPDATE
2960 00000E5D E8C0050000       <1>       call   SND_LED                  ; GO TURN ON MODE INDICATORS
2961                            <1> UP0:
2962 00000E62 6658             <1>       pop    ax                  ; RESTORE DATA IN
2963                            <1> ;-------------------------------------------------------------------------
2964                            <1> ;     START OF KEY PROCESSING                                           ;
2965                            <1> ;-------------------------------------------------------------------------
2966 00000E64 88C4             <1>       mov    ah, al              ; SAVE SCAN CODE IN AH ALSO
2967                            <1>       ;
2968                            <1>       ;-----TEST FOR OVERRUN SCAN CODE FROM KEYBOARD
2969 00000E66 3CFF             <1>       cmp    al, KB_OVER_RUN           ; IS THIS AN OVERRUN CHAR
2970 00000E68 0F843F050000     <1>         je     K62                   ; BUFFER_FULL_BEEP
2971                            <1>       ;
2972                            <1> K16:
2973 00000E6E 8A3D[905E0000]   <1>       mov    bh, [KB_FLAG_3]           ; LOAD FLAGS FOR TESTING
2974                            <1>       ;
2975                            <1>       ;-----TEST TO SEE IF A READ_ID IS IN PROGRESS
2976 00000E74 F6C7C0           <1>       test   bh, RD_ID+LC_AB     ; ARE WE DOING A READ ID?
2977 00000E77 7449             <1>       jz     short NOT_ID        ; CONTINUE IF NOT
2978 00000E79 7917             <1>       jns    short TST_ID_2            ; IS THE RD_ID FLAG ON?
2979 00000E7B 3CAB             <1>       cmp    al, ID_1            ; IS THIS THE 1ST ID CHARACTER?
2980 00000E7D 7507             <1>       jne    short RST_RD_ID
2981 00000E7F 800D[905E0000]40 <1>       or     byte [KB_FLAG_3], LC_AB ; INDICATE 1ST ID WAS OK
2982                            <1> RST_RD_ID:
2983 00000E86 8025[905E0000]7F <1>       and    byte [KB_FLAG_3], ~RD_ID ; RESET THE READ ID FLAG
2984                            <1>         ;jmp    short ID_EX           ; AND EXIT
2985 00000E8D E924020000       <1>       jmp    K26
```

```
                                <1>       ;
                                <1> TST_ID_2:
00000E92 8025[905E0000]BF       <1>       and    byte [KB_FLAG_3], ~LC_AB ; RESET FLAG
00000E99 3C54                   <1>       cmp    al, ID_2A           ; IS THIS THE 2ND ID CHARACTER?
00000E9B 7419                   <1>         je   short KX_BIT        ; JUMP IF SO
00000E9D 3C41                   <1>       cmp    al, ID_2            ; IS THIS THE 2ND ID CHARACTER?
                                <1>         ;jne short ID_EX         ; LEAVE IF NOT
00000E9F 0F8511020000           <1>       jne    K26
                                <1>       ;
                                <1>       ;----- A READ ID SAID THAT IT WAS ENHANCED KEYBOARD
00000EA5 F6C720                 <1>       test   bh, SET_NUM_LK             ; SHOULD WE SET NUM LOCK?
00000EA8 740C                   <1>         jz     short KX_BIT           ; EXIT IF NOT
00000EAA 800D[8D5E0000]20       <1>       or     byte [KB_FLAG], NUM_STATE ; FORCE NUM LOCK ON
00000EB1 E86C050000             <1>       call   SND_LED                   ; GO SET THE NUM LOCK INDICATOR
                                <1> KX_BIT:
00000EB6 800D[905E0000]10       <1>       or     byte [KB_FLAG_3], KBX     ; INDICATE ENHANCED KEYBOARD WAS FOUND
00000EBD E9F4010000             <1> ID_EX:       jmp     K26              ; EXIT
                                <1>       ;
                                <1> NOT_ID:
00000EC2 3CE0                   <1>       cmp    al, MC_E0           ; IS THIS THE GENERAL MARKER CODE?
00000EC4 750C                   <1>       jne    short TEST_E1
00000EC6 800D[905E0000]12       <1>       or     byte [KB_FLAG_3], LC_E0+KBX ; SET FLAG BIT, SET KBX, AND
                                <1>       ;jmp   short EXIT          ; THROW AWAY THIS CODE
00000ECD E9EB010000             <1>       jmp    K26A
                                <1> TEST_E1:
00000ED2 3CE1                   <1>       cmp    al, MC_E1           ; IS THIS THE PAUSE KEY?
00000ED4 750C                   <1>       jne    short NOT_HC
00000ED6 800D[905E0000]11       <1>       or     byte [KB_FLAG_3], LC_E1+KBX ; SET FLAG BIT, SET KBX, AND
00000EDD E9DB010000             <1> EXIT: jmp    K26A                ; THROW AWAY THIS CODE
                                <1>       ;
                                <1> NOT_HC:
00000EE2 247F                   <1>       and    al, 07Fh            ; TURN OFF THE BREAK BIT
00000EE4 F6C702                 <1>       test   bh, LC_E0           ; LAST CODE THE E0 MARKER CODE
00000EE7 7414                   <1>       jz     short NOT_LC_E0           ; JUMP IF NOT
                                <1>       ;
00000EE9 BF[7A5D0000]           <1>       mov    edi, _K6+6          ; IS THIS A SHIFT KEY?
00000EEE AE                     <1>       scasb
00000EEF 0F84C1010000           <1>         je     K26 ; K16B              ; YES, THROW AWAY & RESET FLAG
00000EF5 AE                     <1>       scasb
00000EF6 757C                   <1>       jne    short K16A          ; NO, CONTINUE KEY PROCESSING
                                <1>       ;jmp   short K16B          ; YES, THROW AWAY & RESET FLAG
00000EF8 E9B9010000             <1>       jmp    K26
                                <1>       ;
                                <1> NOT_LC_E0:
00000EFD F6C701                 <1>       test   bh, LC_E1           ; LAST CODE THE E1 MARKER CODE?
00000F00 7435                   <1>       jz     short T_SYS_KEY           ; JUMP IF NOT
00000F02 B904000000             <1>       mov    ecx, 4              ; LENGHT OF SEARCH
00000F07 BF[785D0000]           <1>       mov    edi, _K6+4          ; IS THIS AN ALT, CTL, OR SHIFT?
00000F0C F2AE                   <1>       repne  scasb               ; CHECK IT
                                <1>       ;je    short EXIT          ; THROW AWAY IF SO
00000F0E 0F84A9010000           <1>       je     K26A
                                <1>       ;
00000F14 3C45                   <1>       cmp    al, NUM_KEY         ; IS IT THE PAUSE KEY?
                                <1>       ;jne   short K16B          ; NO, THROW AWAY & RESET FLAG
00000F16 0F859A010000           <1>       jne    K26
00000F1C F6C480                 <1>       test   ah, 80h                   ; YES, IS IT THE BREAK OF THE KEY?
                                <1>       ;jnz   short K16B          ; YES, THROW THIS AWAY, TOO
00000F1F 0F8591010000           <1>       jnz    K26
                                <1>         ; 20/02/2015
00000F25 F605[8E5E0000]08       <1>       test   byte [KB_FLAG_1],HOLD_STATE ;  NO, ARE WE PAUSED ALREADY?
                                <1>       ;jnz   short K16B          ;  YES, THROW AWAY
00000F2C 0F8584010000           <1>       jnz    K26
00000F32 E9E1020000             <1>       jmp     K39P                    ; NO, THIS IS THE REAL PAUSE STATE
                                <1>       ;
                                <1>       ;----- TEST FOR SYSTEM KEY
                                <1> T_SYS_KEY:
00000F37 3C54                   <1>       cmp    al, SYS_KEY         ; IS IT THE SYSTEM KEY?
00000F39 7539                   <1>       jnz    short K16A          ; CONTINUE IF NOT
                                <1>       ;
00000F3B F6C480                 <1>       test   ah, 80h                   ; CHECK IF THIS A BREAK CODE
00000F3E 7524                   <1>       jnz    short K16C          ; DO NOT TOUCH SYSTEM INDICATOR IF TRUE
                                <1>       ;
00000F40 F605[8E5E0000]04       <1>       test   byte [KB_FLAG_1], SYS_SHIFT ; SEE IF IN SYSTEM KEY HELD DOWN
                                <1>         ;jnz short K16B          ; IF YES, DO NOT PROCESS SYSTEM INDICATOR
00000F47 0F8569010000           <1>       jnz     K26
                                <1>       ;
00000F4D 800D[8E5E0000]04       <1>       or     byte [KB_FLAG_1], SYS_SHIFT ; INDICATE SYSTEM KEY DEPRESSED
00000F54 B020                   <1>       mov    al, EOI                   ; END OF INTERRUPT COMMAND
00000F56 E620                   <1>       out    20h, al ;out INTA00, al   ; SEND COMMAND TO INTERRUPT CONTROL PORT
                                <1>                                      ; INTERRUPT-RETURN-NO-EOI
00000F58 B0AE                   <1>       mov    al, ENA_KBD         ; INSURE KEYBOARD IS ENABLED
00000F5A E862040000             <1>       call   SHIP_IT                   ; EXECUTE ENABLE
                                <1>       ; !!! SYSREQ !!! function/system call (INTERRUPT) must be here !!!
                                <1>       ;MOV   AL, 8500H           ; FUNCTION VALUE FOR MAKE OF SYSTEM KEY
                                <1>       ;STI                       ; MAKE SURE INTERRUPTS ENABLED
                                <1>       ;INT   15H                 ; USER INTERRUPT
00000F5F E965010000             <1>         jmp     K27A                    ; END PROCESSING
                                <1>       ;
                                <1> ;K16B:       jmp    K26                ; IGNORE SYSTEM KEY
                                <1>       ;
                                <1> K16C:
00000F64 8025[8E5E0000]FB       <1>       and    byte [KB_FLAG_1], ~SYS_SHIFT ; TURN OFF SHIFT KEY HELD DOWN
00000F6B B020                   <1>       mov    al, EOI                   ; END OF INTERRUPT COMMAND
00000F6D E620                   <1>       out    20h, al ;out INTA00, al ; SEND COMMAND TO INTERRUPT CONTROL PORT
                                <1>                                      ; INTERRUPT-RETURN-NO-EOI
                                <1>       ;MOV   AL, ENA_KBD         ; INSURE KEYBOARD IS ENABLED
                                <1>       ;CALL  SHIP_IT                   ; EXECUTE ENABLE
                                <1>       ;
                                <1>       ;MOV   AX, 8501H           ; FUNCTION VALUE FOR BREAK OF SYSTEM KEY
                                <1>       ;STI                       ; MAKE SURE INTERRUPTS ENABLED
                                <1>       ;INT   15H                 ; USER INTERRUPT
                                <1>       ;JMP   K27A                ; INGONRE SYSTEM KEY
                                <1>       ;
```

```
3089 00000F6F E94E010000          <1>       jmp     K27               ; IGNORE SYSTEM KEY
3090                              <1>       ;
3091                              <1>       ;----- TEST FOR SHIFT KEYS
3092                              <1> K16A:
3093 00000F74 8A1D[8D5E0000]      <1>       mov     bl, [KB_FLAG]       ; PUT STATE FLAGS IN BL
3094 00000F7A BF[745D0000]        <1>       mov     edi, _K6            ; SHIFT KEY TABLE offset
3095 00000F7F B908000000          <1>       mov     ecx, _K6L           ; LENGTH
3096 00000F84 F2AE                <1>       repne   scasb               ; LOOK THROUGH THE TABLE FOR A MATCH
3097 00000F86 88E0                <1>       mov     al, ah              ; RECOVER SCAN CODE
3098 00000F88 0F8510010000        <1>         jne     K25                       ; IF NO MATCH, THEN SHIFT NOT FOUND
3099                              <1>       ;
3100                              <1>       ;------      SHIFT KEY FOUND
3101                              <1> K17:
3102 00000F8E 81EF[755D0000]      <1>         sub     edi, _K6+1              ; ADJUST PTR TO SCAN CODE MATCH
3103 00000F94 8AA7[7C5D0000]      <1>             mov     ah, [edi+_K7]             ; GET MASK INTO AH
3104 00000F9A B102                <1>       mov     cl, 2               ; SETUP COUNT FOR FLAG SHIFTS
3105 00000F9C A880                <1>       test    al, 80h                   ; TEST FOR BREAK KEY
3106 00000F9E 0F8596000000        <1>         jnz     K23                     ; JUMP OF BREAK
3107                              <1>       ;
3108                              <1>       ;----- SHIFT MAKE FOUND, DETERMINE SET OR TOGGLE
3109                              <1> K17C:
3110 00000FA4 80FC10              <1>       cmp     ah, SCROLL_SHIFT
3111 00000FA7 732B                <1>       jae     short K18           ; IF SCROLL SHIFT OR ABOVE, TOGGLE KEY
3112                              <1>       ;
3113                              <1>       ;----- PLAIN SHIFT KEY, SET SHIFT ON
3114 00000FA9 0825[8D5E0000]      <1>       or      [KB_FLAG], ah       ; TURN ON SHIFT BIT
3115 00000FAF A80C                <1>         test    al, CTL_SHIFT+ALT_SHIFT ; IS IT ALT OR CTRL?
3116                              <1>       ;jnz    short K17D          ; YES, MORE FLAGS TO SET
3117 00000FB1 0F84FF000000        <1>       jz      K26                 ; NO, INTERRUPT RETURN
3118                              <1> K17D:
3119 00000FB7 F6C702              <1>       test    bh, LC_E0           ; IS THIS ONE OF NEW KEYS?
3120 00000FBA 740B                <1>       jz      short K17E          ; NO, JUMP
3121 00000FBC 0825[905E0000]      <1>       or      [KB_FLAG_3], ah           ; SET BITS FOR RIGHT CTRL, ALT
3122 00000FC2 E9EF000000          <1>       jmp     K26                 ; INTERRUPT RETURN
3123                              <1> K17E:
3124 00000FC7 D2EC                <1>       shr     ah, cl              ; MOVE FLAG BITS TWO POSITIONS
3125 00000FC9 0825[8E5E0000]      <1>       or      [KB_FLAG_1], ah           ; SET BITS FOR LEFT CTRL, ALT
3126 00000FCF E9E2000000          <1>       jmp     K26
3127                              <1>       ;
3128                              <1>       ;----- TOGGLED SHIFT KEY, TEST FOR 1ST MAKE OR NOT
3129                              <1> K18:                            ; SHIFT-TOGGLE
3130 00000FD4 F6C304              <1>       test    bl, CTL_SHIFT             ; CHECK CTL SHIFT STATE
3131                              <1>         ;jz       short K18A              ; JUMP IF NOT CTL STATE
3132 00000FD7 0F85C1000000        <1>         jnz     K25                     ; JUMP IF CTL STATE
3133                              <1> K18A:
3134 00000FDD 3C52                <1>       cmp     al, INS_KEY         ; CHECK FOR INSERT KEY
3135 00000FDF 7524                <1>       jne     short K22           ; JUMP IF NOT INSERT KEY
3136 00000FE1 F6C308              <1>       test    bl, ALT_SHIFT             ; CHECK FOR ALTERNATE SHIFT
3137                              <1>               ;jz    short K18B          ; JUMP IF NOT ALTERNATE SHIFT
3138 00000FE4 0F85B4000000        <1>         jnz     K25                     ; JUMP IF ALTERNATE SHIFT
3139                              <1> K18B:
3140 00000FEA F6C702              <1>       test    bh, LC_E0 ;20/02/2015     ; IS THIS NEW INSERT KEY?
3141 00000FED 7516                <1>       jnz     short K22           ; YES, THIS ONE'S NEVER A '0'
3142                              <1> K19:
3143 00000FEF F6C320              <1>       test    bl, NUM_STATE             ; CHECK FOR BASE STATE
3144 00000FF2 750C                <1>       jnz     short K21           ; JUMP IF NUM LOCK IS ON
3145 00000FF4 F6C303              <1>       test    bl, LEFT_SHIFT+RIGHT_SHIFT ; TEST FOR SHIFT STATE
3146 00000FF7 740C                <1>       jz      short K22           ; JUMP IF BASE STATE
3147                              <1> K20:                            ; NUMERIC ZERO, NOT INSERT KEY
3148 00000FF9 88C4                <1>       mov     ah, al              ; PUT SCAN CODE BACK IN AH
3149 00000FFB E99E000000          <1>         jmp     K25                     ; NUMERAL '0', STNDRD. PROCESSING
3150                              <1> K21:                            ; MIGHT BE NUMERIC
3151 00001000 F6C303              <1>       test    bl, LEFT_SHIFT+RIGHT_SHIFT
3152 00001003 74F4                <1>       jz      short K20           ; IS NUMERIC, STD. PROC.
3153                              <1>       ;
3154                              <1> K22:                            ; SHIFT TOGGLE KEY HIT; PROCESS IT
3155 00001005 8425[8E5E0000]      <1>       test    ah, [KB_FLAG_1]     ; IS KEY ALREADY DEPRESSED
3156 0000100B 0F85A5000000        <1>         jnz     K26                     ; JUMP IF KEY ALREADY DEPRESSED
3157                              <1> K22A:
3158 00001011 0825[8E5E0000]      <1>         or      [KB_FLAG_1], ah ; INDICATE THAT THE KEY IS DEPRESSED
3159 00001017 3025[8D5E0000]      <1>       xor     [KB_FLAG], ah       ; TOGGLE THE SHIFT STATE
3160                              <1>       ;
3161                              <1>       ;----- TOGGLE LED IF CAPS, NUM  OR SCROLL KEY DEPRESSED
3162 0000101D F6C470              <1>       test    ah, CAPS_SHIFT+NUM_SHIFT+SCROLL_SHIFT ; SHIFT TOGGLE?
3163 00001020 7409                <1>       jz      short K22B          ; GO IF NOT
3164                              <1>       ;
3165 00001022 6650                <1>       push    ax                  ; SAVE SCAN CODE AND SHIFT MASK
3166 00001024 E8F9030000          <1>       call    SND_LED                   ; GO TURN MODE INDICATORS ON
3167 00001029 6658                <1>       pop     ax                  ; RESTORE SCAN CODE
3168                              <1> K22B:
3169 0000102B 3C52                <1>       cmp     al, INS_KEY         ; TEST FOR 1ST MAKE OF INSERT KEY
3170 0000102D 0F8583000000        <1>         jne     K26                     ; JUMP IF NOT INSERT KEY
3171 00001033 88C4                <1>       mov     ah, al                ; SCAN CODE IN BOTH HALVES OF AX
3172 00001035 E999000000          <1>         jmp     K28                     ; FLAGS UPDATED, PROC. FOR BUFFER
3173                              <1>       ;
3174                              <1>       ;----- BREAK SHIFT FOUND
3175                              <1> K23:                            ; BREAK-SHIFT-FOUND
3176 0000103A 80FC10              <1>       cmp     ah, SCROLL_SHIFT    ; IS THIS A TOGGLE KEY
3177 0000103D F6D4                <1>       not     ah                  ; INVERT MASK
3178 0000103F 7355                <1>       jae     short K24           ; YES, HANDLE BREAK TOGGLE
3179 00001041 2025[8D5E0000]      <1>       and     [KB_FLAG], ah       ; TURN OFF SHIFT BIT
3180 00001047 80FCFB              <1>       cmp     ah, ~CTL_SHIFT            ; IS THIS ALT OR CTL?
3181 0000104A 7730                <1>       ja      short K23D          ; NO, ALL DONE
3182                              <1>       ;
3183 0000104C F6C702              <1>       test    bh, LC_E0           ; 2ND ALT OR CTL?
3184 0000104F 7408                <1>       jz      short K23A          ; NO, HANSLE NORMALLY
3185 00001051 2025[905E0000]      <1>       and     [KB_FLAG_3], ah           ; RESET BIT FOR RIGHT ALT OR CTL
3186 00001057 EB08                <1>       jmp     short K23B          ; CONTINUE
3187                              <1> K23A:
3188 00001059 D2FC                <1>       sar     ah, cl              ; MOVE THE MASK BIT TWO POSITIONS
3189 0000105B 2025[8E5E0000]      <1>       and     [KB_FLAG_1], ah           ; RESET BIT FOR LEFT ALT AND CTL
3190                              <1> K23B:
3191 00001061 88C4                <1>       mov     ah, al              ; SAVE SCAN CODE
```

```
3192 00001063 A0[905E0000]        <1>       mov    al, [KB_FLAG_3]           ; GET RIGHT ALT & CTRL FLAGS
3193 00001068 D2E8                <1>       shr    al, cl              ; MOVE TO BITS 1 & 0
3194 0000106A 0A05[8E5E0000]      <1>       or     al, [KB_FLAG_1]           ; PUT IN LEFT ALŞT & CTL FLAGS
3195 00001070 D2E0                <1>       shl    al, cl              ; MOVE BACK TO BITS 3 & 2
3196 00001072 240C                <1>       and    al, ALT_SHIFT+CTL_SHIFT ; FILTER OUT OTHER GARBAGE
3197 00001074 0805[8D5E0000]      <1>       or     [KB_FLAG], al       ; PUT RESULT IN THE REAL FLAGS
3198 0000107A 88E0                <1>       mov    al, ah
3199                              <1> K23D:
3200 0000107C 3CB8                <1>       cmp    al, ALT_KEY+80h           ; IS THIS ALTERNATE SHIFT RELEASE
3201 0000107E 7536                <1>       jne    short K26           ; INTERRUPT RETURN
3202                              <1>       ;
3203                              <1>       ;----- ALTERNATE SHIFT KEY RELEASED, GET THE VALUE INTO BUFFER
3204 00001080 A0[915E0000]        <1>       mov    al, [ALT_INPUT]
3205 00001085 B400                <1>       mov    ah, 0               ; SCAN CODE OF 0
3206 00001087 8825[915E0000]      <1>       mov    [ALT_INPUT], ah     ; ZERO OUT THE FIELD
3207 0000108D 3C00                <1>       cmp    al, 0               ; WAS THE INPUT = 0?
3208 0000108F 7425                <1>       je     short K26           ; INTERRUPT_RETURN
3209                              <1>         ; 29/01/2016
3210                              <1>       ;jmp     K61                    ; IT WASN'T, SO PUT IN BUFFER
3211 00001091 E9D0020000          <1>       jmp    _K60
3212                              <1>       ;
3213                              <1> K24:                            ; BREAK-TOGGLE
3214 00001096 2025[8E5E0000]      <1>       and    [KB_FLAG_1], ah     ; INDICATE NO LONGER DEPRESSED
3215 0000109C EB18                <1>       jmp    short K26           ; INTERRUPT_RETURN
3216                              <1>       ;
3217                              <1>       ;----- TEST FOR HOLD STATE
3218                              <1>                                   ; AL, AH = SCAN CODE
3219                              <1> K25:                            ; NO-SHIFT-FOUND
3220 0000109E 3C80                <1>       cmp    al, 80h                  ; TEST FOR BREAK KEY
3221 000010A0 7314                <1>       jae    short K26           ; NOTHING FOR BREAK CHARS FROM HERE ON
3222 000010A2 F605[8E5E0000]08    <1>       test   byte [KB_FLAG_1], HOLD_STATE ; ARE WE IN HOLD STATE
3223 000010A9 7428                <1>       jz     short K28           ; BRANCH AROUND TEST IF NOT
3224 000010AB 3C45                <1>       cmp    al, NUM_KEY
3225 000010AD 7407                <1>       je     short K26             ; CAN'T END HOLD ON NUM_LOCK
3226 000010AF 8025[8E5E0000]F7    <1>       and    byte [KB_FLAG_1], ~HOLD_STATE ; TURN OFF THE HOLD STATE BIT
3227                              <1>       ;
3228                              <1> K26:
3229 000010B6 8025[905E0000]FC    <1>       and    byte [KB_FLAG_3], ~(LC_E0+LC_E1) ; RESET LAST CHAR H.C. FLAG
3230                              <1> K26A:                           ; INTERRUPT-RETURN
3231 000010BD FA                  <1>       cli                         ; TURN OFF INTERRUPTS
3232 000010BE B020                <1>       mov    al, EOI                  ; END OF INTERRUPT COMMAND
3233 000010C0 E620                <1>       out    20h, al      ;out INTA00, al     ; SEND COMMAND TO INTERRUPT CONTROL PORT
3234                              <1> K27:                            ; INTERRUPT-RETURN-NO-EOI
3235 000010C2 B0AE                <1>       mov    al, ENA_KBD         ; INSURE KEYBOARD IS ENABLED
3236 000010C4 E8F8020000          <1>       call   SHIP_IT                  ; EXECUTE ENABLE
3237                              <1> K27A:
3238 000010C9 FA                  <1>       cli                         ; DISABLE INTERRUPTS
3239                              <1>       ;;mov  byte [intflg], 0 ; 07/01/2017 ;; 15/01/2017
3240 000010CA 07                  <1>       pop    es                   ; RESTORE REGISTERS
3241 000010CB 1F                  <1>       pop    ds
3242 000010CC 5F                  <1>       pop    edi
3243 000010CD 5E                  <1>       pop    esi
3244 000010CE 5A                  <1>       pop    edx
3245 000010CF 59                  <1>       pop    ecx
3246 000010D0 5B                  <1>       pop    ebx
3247 000010D1 58                  <1>       pop    eax
3248                              <1>       ;pop   ebp
3249 000010D2 CF                  <1>       iretd                       ; RETURN
3250                              <1>
3251                              <1>       ;----- NOT IN HOLD STATE
3252                              <1> K28:                            ; NO-HOLD-STATE
3253 000010D3 3C58                <1>       cmp    al, 88              ; TEST FOR OUT-OF-RANGE SCAN CODES
3254 000010D5 77DF                <1>       ja     short K26           ; IGNORE IF OUT-OF-RANGE
3255                              <1>       ;
3256 000010D7 F6C308              <1>       test   bl, ALT_SHIFT               ; ARE WE IN ALTERNATE SHIFT
3257                              <1>         ;jz  short K28A         ; IF NOT ALTERNATE
3258 000010DA 0F84F1000000        <1>         jz      K38
3259                              <1>       ;
3260 000010E0 F6C710              <1>       test   bh, KBX                  ; IS THIS THE ENCHANCED KEYBOARD?
3261 000010E3 740D                <1>       jz     short K29           ; NO, ALT STATE IS REAL
3262                              <1>        ;28/02/2015
3263 000010E5 F605[8E5E0000]04    <1>       test   byte [KB_FLAG_1], SYS_SHIFT ; YES, IS SYSREQ KEY DOWN?
3264                              <1>       ;jz    short K29            ;  NO, ALT STATE IS REAL
3265 000010EC 0F85DF000000        <1>       jnz    K38                  ; YES, THIS IS PHONY ALT STATE
3266                              <1>         ;                         ; DUE TO PRESSING SYSREQ
3267                              <1> ;K28A:       jmp    short K38
3268                              <1>       ;
3269                              <1>       ;----- TEST FOR RESET KEY SEQUENCE (CTL ALT DEL)
3270                              <1> K29:                            ; TEST-RESET
3271 000010F2 F6C304              <1>       test   bl, CTL_SHIFT               ; ARE WE IN CONTROL SHIFT ALSO?
3272 000010F5 740B                <1>       jz     short K31           ; NO_RESET
3273 000010F7 3C53                <1>       cmp    al, DEL_KEY         ; CTL-ALT STATE, TEST FOR DELETE KEY
3274 000010F9 7507                <1>       jne    short K31           ; NO_RESET, IGNORE
3275                              <1>       ;
3276                              <1>       ;----- CTL-ALT-DEL HAS BEEN FOUND
3277                              <1>       ; 26/08/2014
3278                              <1> cpu_reset:
3279                              <1>       ; IBM PC/AT ROM BIOS source code - 10/06/85 (TEST4.ASM - PROC_SHUTDOWN)
3280                              <1>       ; Send FEh (system reset command) to the keyboard controller.
3281 000010FB B0FE                <1>       mov    al, SHUT_CMD        ; SHUTDOWN COMMAND
3282 000010FD E664                <1>       out    STATUS_PORT, al           ; SEND TO KEYBOARD CONTROL PORT
3283                              <1> khere:
3284 000010FF F4                  <1>       hlt                         ; WAIT FOR 80286 RESET
3285 00001100 EBFD                <1>       jmp    short khere         ; INSURE HALT
3286                              <1>
3287                              <1>       ;
3288                              <1>       ;----- IN ALTERNATE SHIFT, RESET NOT FOUND
3289                              <1> K31:                            ; NO-RESET
3290 00001102 3C39                <1>       cmp    al, 57              ; TEST FOR SPACE KEY
3291 00001104 7507                <1>       jne    short K311          ; NOT THERE
3292 00001106 B020                <1>       mov    al, ' '                  ; SET SPACE CHAR
3293 00001108 E948020000          <1>         jmp     K57                    ; BUFFER_FILL
3294                              <1> K311:
```

```
3295 0000110D 3C0F              <1>       cmp    al, 15             ; TEST FOR TAB KEY
3296 0000110F 7509              <1>       jne    short K312         ; NOT THERE
3297 00001111 66B800A5          <1>       mov    ax, 0A500h         ; SET SPECIAL CODE FOR ALT-TAB
3298 00001115 E93B020000        <1>         jmp     K57                     ; BUFFER_FILL
3299                            <1> K312:
3300 0000111A 3C4A              <1>       cmp    al, 74             ; TEST FOR KEY PAD -
3301 0000111C 0F84A2000000      <1>         je      K37B                    ; GO PROCESS
3302 00001122 3C4E              <1>       cmp    al, 78             ; TEST FOR KEY PAD +
3303 00001124 0F849A000000      <1>         je      K37B                    ; GO PROCESS
3304                            <1>       ;
3305                            <1>       ;----- LOOK FOR KEY PAD ENTRY
3306                            <1> K32:                           ; ALT-KEY-PAD
3307 0000112A BF[505D0000]      <1>       mov    edi, K30           ; ALT-INPUT-TABLE offset
3308 0000112F B90A000000        <1>       mov    ecx, 10                 ; LOOK FOR ENTRY USING KEYPAD
3309 00001134 F2AE              <1>       repne  scasb              ; LOOK FOR MATCH
3310 00001136 7525              <1>       jne    short K33          ; NO_ALT_KEYPAD
3311 00001138 F6C702            <1>       test   bh, LC_E0          ; IS THIS ONE OF THE NEW KEYS?
3312 0000113B 0F858A000000      <1>         jnz     K37C                    ; YES, JUMP, NOT NUMPAD KEY
3313 00001141 81EF[515D0000]    <1>       sub    edi, K30+1         ; DI NOW HAS ENTRY VALUE
3314 00001147 A0[915E0000]      <1>       mov    al, [ALT_INPUT]    ; GET THE CURRENT BYTE
3315 0000114C B40A              <1>       mov    ah, 10             ; MULTIPLY BY 10
3316 0000114E F6E4              <1>       mul    ah
3317 00001150 6601F8            <1>       add    ax, di             ; ADD IN THE LATEST ENTRY
3318 00001153 A2[915E0000]      <1>       mov    [ALT_INPUT], al    ; STORE IT AWAY
3319                            <1> ;K32A:
3320 00001158 E959FFFFFF        <1>         jmp     K26                     ; THROW AWAY THAT KEYSTROKE
3321                            <1>       ;
3322                            <1>       ;----- LOOK FOR SUPERSHIFT ENTRY
3323                            <1> K33:                           ; NO-ALT-KEYPAD
3324 0000115D C605[915E0000]00  <1>         mov     byte [ALT_INPUT], 0     ; ZERO ANY PREVIOUS ENTRY INTO INPUT
3325 00001164 B91A000000        <1>       mov    ecx, 26                 ; (DI),(ES) ALREADY POINTING
3326 00001169 F2AE              <1>       repne  scasb              ; LOOK FOR MATCH IN ALPHABET
3327 0000116B 7450              <1>       je     short K37A         ; MATCH FOUND, GO FILLL THE BUFFER
3328                            <1>       ;
3329                            <1>       ;----- LOOK FOR TOP ROW OF ALTERNATE SHIFT
3330                            <1> K34:                           ; ALT-TOP-ROW
3331 0000116D 3C02              <1>       cmp    al, 2              ; KEY WITH '1' ON IT
3332 0000116F 7253              <1>       jb     short K37B         ; MUST BE ESCAPE
3333 00001171 3C0D              <1>       cmp    al, 13             ; IS IT IN THE REGION
3334 00001173 7705              <1>       ja     short K35          ; NO, ALT SOMETHING ELSE
3335 00001175 80C476            <1>       add    ah, 118                 ; CONVERT PSEUDO SCAN CODE TO RANGE
3336 00001178 EB43              <1>       jmp    short K37A         ; GO FILL THE BUFFER
3337                            <1>       ;
3338                            <1>       ;----- TRANSLATE ALTERNATE SHIFT PSEUDO SCAN CODES
3339                            <1> K35:                           ; ALT-FUNCTION
3340 0000117A 3C57              <1>       cmp    al, F11_M          ; IS IT F11?
3341 0000117C 7209              <1>       jb     short K35A ; 20/02/2015   ; NO, BRANCH
3342 0000117E 3C58              <1>       cmp    al, F12_M          ; IS IT F12?
3343 00001180 7705              <1>       ja     short K35A ; 20/02/2015   ; NO, BRANCH
3344 00001182 80C434            <1>       add    ah, 52             ; CONVERT TO PSEUDO SCAN CODE
3345 00001185 EB36              <1>       jmp    short K37A         ; GO FILL THE BUFFER
3346                            <1> K35A:
3347 00001187 F6C702            <1>       test   bh, LC_E0          ; DO WE HAVE ONE OF THE NEW KEYS?
3348 0000118A 7422              <1>       jz     short K37          ; NO, JUMP
3349 0000118C 3C1C              <1>       cmp    al, 28             ; TEST FOR KEYPAD ENTER
3350 0000118E 7509              <1>         jne     short K35B              ; NOT THERE
3351 00001190 66B800A6          <1>       mov    ax, 0A600h         ; SPECIAL CODE
3352 00001194 E9BC010000        <1>       jmp    K57                ; BUFFER FILL
3353                            <1> K35B:
3354 00001199 3C53              <1>       cmp    al, 83             ; TEST FOR DELETE KEY
3355 0000119B 742E              <1>       je     short K37C         ; HANDLE WITH OTHER EDIT KEYS
3356 0000119D 3C35              <1>       cmp    al, 53             ; TEST FOR KEYPAD /
3357                            <1>       ;jne   short K32A         ; NOT THERE, NO OTHER E0 SPECIALS
3358 0000119F 0F8511FFFFFF      <1>         jne     K26
3359 000011A5 66B800A4          <1>       mov    ax, 0A400h         ; SPECIAL CODE
3360 000011A9 E9A7010000        <1>       jmp    K57                ; BUFFER FILL
3361                            <1> K37:
3362 000011AE 3C3B              <1>       cmp    al, 59             ; TEST FOR FUNCTION KEYS (F1)
3363 000011B0 7212              <1>         jb      short K37B              ; NO FN, HANDLE W/OTHER EXTENDED
3364 000011B2 3C44              <1>       cmp    al, 68             ; IN KEYPAD REGION?
3365                            <1>         ;ja   short K32A         ; IF SO, IGNORE
3366 000011B4 0F87FCFEFFFF      <1>         ja      K26
3367 000011BA 80C42D            <1>       add    ah, 45             ; CONVERT TO PSEUDO SCAN CODE
3368                            <1> K37A:
3369 000011BD B000              <1>       mov    al, 0              ; ASCII CODE OF ZERO
3370 000011BF E991010000        <1>         jmp     K57                     ; PUT IT IN THE BUFFER
3371                            <1> K37B:
3372 000011C4 B0F0              <1>       mov    al, 0F0h           ; USE SPECIAL ASCII CODE
3373 000011C6 E98A010000        <1>       jmp     K57                    ; PUT IT IN THE BUFFER
3374                            <1> K37C:
3375 000011CB 0450              <1>       add    al, 80             ; CONVERT SCAN CODE (EDIT KEYS)
3376 000011CD 88C4              <1>       mov    ah, al             ; (SCAN CODE NOT IN AH FOR INSERT)
3377 000011CF EBEC              <1>       jmp     short K37A              ; PUT IT IN THE BUFFER
3378                            <1>       ;
3379                            <1>       ;----- NOT IN ALTERNATE SHIFT
3380                            <1> K38:                           ; NOT-ALT-SHIFT
3381                            <1>                                ; BL STILL HAS SHIFT FLAGS
3382 000011D1 F6C304            <1>       test   bl, CTL_SHIFT             ; ARE WE IN CONTROL SHIFT?
3383                            <1>       ;jnz   short K38A         ; YES, START PROCESSING
3384 000011D4 0F84B0000000      <1>         jz      K44                     ; NOT-CTL-SHIFT
3385                            <1>       ;
3386                            <1>       ;----- CONTROL SHIFT, TEST SPECIAL CHARACTERS
3387                            <1>       ;----- TEST FOR BREAK
3388                            <1> K38A:
3389 000011DA 3C46              <1>       cmp    al, SCROLL_KEY            ; TEST FOR BREAK
3390 000011DC 7531              <1>       jne    short K39          ; JUMP, NO-BREAK
3391 000011DE F6C710            <1>       test   bh, KBX                   ; IS THIS THE ENHANCED KEYBOARD?
3392 000011E1 7405              <1>       jz     short K38B         ; NO, BREAK IS VALID
3393 000011E3 F6C702            <1>       test   bh, LC_E0          ; YES, WAS LAST CODE AN E0?
3394 000011E6 7427              <1>       jz     short K39          ; NO-BREAK, TEST FOR PAUSE
3395                            <1> K38B:
3396 000011E8 8B1D[9A5E0000]    <1>       mov    ebx, [BUFFER_HEAD] ; RESET BUFFER TO EMPTY
3397 000011EE 891D[9E5E0000]    <1>       mov    [BUFFER_TAIL], ebx
```

```
3398 000011F4 C605[8C5E0000]80    <1>       mov    byte [BIOS_BREAK], 80h  ; TURN ON BIOS_BREAK BIT
3399                              <1>       ;
3400                              <1>       ;----- ENABLE KEYBOARD
3401 000011FB B0AE                <1>       mov    al, ENA_KBD         ; ENABLE KEYBOARD
3402 000011FD E8BF010000          <1>       call   SHIP_IT                   ; EXECUTE ENABLE
3403                              <1>       ;
3404                              <1>       ; CTRL+BREAK code here !!!
3405                              <1>       ;INT   1BH                 ; BREAK INTERRUPT VECTOR
3406                              <1>       ; 17/10/2015
3407 00001202 E8CF510000          <1>       call   ctrlbrk ; control+break subroutine
3408                              <1>       ;
3409 00001207 6629C0              <1>       sub    ax, ax              ; PUT OUT DUMMY CHARACTER
3410 0000120A E946010000          <1>         jmp     K57                    ; BUFFER_FILL
3411                              <1>       ;
3412                              <1>       ;----- TEST FOR PAUSE
3413                              <1> K39:                              ; NO_BREAK
3414 0000120F F6C710              <1>       test   bh, KBX                   ; IS THIS THE ENHANCED KEYBOARD?
3415 00001212 7537                <1>       jnz    short K41           ; YES, THEN THIS CAN'T BE PAUSE
3416 00001214 3C45                <1>       cmp    al, NUM_KEY         ; LOOK FOR PAUSE KEY
3417 00001216 7533                <1>       jne    short K41           ; NO-PAUSE
3418                              <1> K39P:
3419 00001218 800D[8E5E0000]08    <1>       or     byte [KB_FLAG_1], HOLD_STATE ; TURN ON THE HOLD FLAG
3420                              <1>       ;
3421                              <1>       ;----- ENABLE KEYBOARD
3422 0000121F B0AE                <1>       mov    al, ENA_KBD         ; ENABLE KEYBOARD
3423 00001221 E89B010000          <1>       call   SHIP_IT                   ; EXECUTE ENABLE
3424                              <1> K39A:
3425 00001226 B020                <1>       mov    al, EOI                   ; END OF INTERRUPT TO CONTROL PORT
3426 00001228 E620                <1>       out    20h, al ;out INTA00, al   ; ALLOW FURTHER KEYSTROKE INTERRUPTS
3427                              <1>       ;
3428                              <1>       ;----- DURING PAUSE INTERVAL, TURN COLOR CRT BACK ON
3429 0000122A 803D[C25E0000]07    <1>         cmp     byte [CRT_MODE], 7      ; IS THIS BLACK AND WHITE CARD
3430 00001231 740A                <1>         je      short K40                    ; YES, NOTHING TO DO
3431 00001233 66BAD803            <1>       mov    dx, 03D8h           ; PORT FOR COLOR CARD
3432 00001237 A0[C35E0000]        <1>         mov     al, [CRT_MODE_SET]     ; GET THE VALUE OF THE CURRENT MODE
3433 0000123C EE                  <1>       out    dx, al              ; SET THE CRT MODE, SO THAT CRT IS ON
3434                              <1>       ;
3435                              <1> K40:                              ; PAUSE-LOOP
3436 0000123D F605[8E5E0000]08    <1>         test    byte [KB_FLAG_1], HOLD_STATE ; CHECK HOLD STATE FLAG
3437 00001244 75F7                <1>       jnz    short K40           ; LOOP UNTIL FLAG TURNED OFF
3438                              <1>       ;
3439 00001246 E977FEFFFF          <1>         jmp     K27                     ; INTERRUPT_RETURN_NO_EOI
3440                              <1>         ;
3441                              <1>       ;----- TEST SPECIAL CASE KEY 55
3442                              <1> K41:                              ; NO-PAUSE
3443 0000124B 3C37                <1>       cmp    al, 55              ; TEST FOR */PRTSC KEY
3444 0000124D 7513                <1>       jne    short K42           ; NOT-KEY-55
3445 0000124F F6C710              <1>       test   bh, KBX                   ; IS THIS THE ENHANCED KEYBOARD?
3446 00001252 7405                <1>       jz     short K41A          ; NO, CTL-PRTSC IS VALID
3447 00001254 F6C702              <1>       test   bh, LC_E0           ; YES, WAS LAST CODE AN E0?
3448 00001257 7421                <1>       jz     short K42B          ; NO, TRANSLATE TO A FUNCTION
3449                              <1> K41A:
3450 00001259 66B80072            <1>       mov    ax, 114*256         ; START/STOP PRINTING SWITCH
3451 0000125D E9F3000000          <1>         jmp     K57                    ; BUFFER_FILL
3452                              <1>       ;
3453                              <1>       ;----- SET UP TO TRANSLATE CONTROL SHIFT
3454                              <1> K42:                              ; NOT-KEY-55
3455 00001262 3C0F                <1>       cmp    al, 15              ; IS IT THE TAB KEY?
3456 00001264 7414                <1>       je     short K42B          ; YES, XLATE TO FUNCTION CODE
3457 00001266 3C35                <1>       cmp    al, 53              ; IS IT THE / KEY?
3458 00001268 750E                <1>       jne    short K42A          ; NO, NO MORE SPECIAL CASES
3459 0000126A F6C702              <1>       test   bh, LC_E0           ; YES, IS IT FROM THE KEY PAD?
3460 0000126D 7409                <1>       jz     short K42A          ; NO, JUST TRANSLATE
3461 0000126F 66B80095            <1>       mov    ax, 9500h           ; YES, SPECIAL CODE FOR THIS ONE
3462 00001273 E9DD000000          <1>       jmp    K57                 ; BUFFER FILL
3463                              <1> K42A:
3464                              <1>       ;;mov  ebx, _K8            ; SET UP TO TRANSLATE CTL
3465 00001278 3C3B                <1>       cmp    al, 59              ; IS IT IN CHARACTER TABLE?
3466                              <1>         ;jb  short K45F              ; YES, GO TRANSLATE CHAR
3467                              <1>       ;;jb   K56 ; 20/02/2015
3468                              <1>       ;;jmp  K64 ; 20/02/2015
3469                              <1> K42B:
3470 0000127A BB[845D0000]        <1>       mov    ebx, _K8            ; SET UP TO TRANSLATE CTL
3471 0000127F 0F82AE000000        <1>       jb     K56 ;; 20/02/2015
3472 00001285 E9B9000000          <1>       jmp    K64
3473                              <1>         ;
3474                              <1>       ;----- NOT IN CONTROL SHIFT
3475                              <1> K44:                              ; NOT-CTL-SHIFT
3476 0000128A 3C37                <1>       cmp    al, 55              ; PRINT SCREEN KEY?
3477 0000128C 7528                <1>       jne    short K45           ; NOT PRINT SCREEN
3478 0000128E F6C710              <1>       test   bh, KBX                   ; IS THIS ENHANCED KEYBOARD?
3479 00001291 7407                <1>       jz     short K44A          ; NO, TEST FOR SHIFT STATE
3480 00001293 F6C702              <1>       test   bh, LC_E0           ; YES, LAST CODE A MARKER?
3481 00001296 7507                <1>       jnz    short K44B          ; YES, IS PRINT SCREEN
3482 00001298 EB41                <1>       jmp    short K45C          ; NO, TRANSLATE TO '*' CHARACTER
3483                              <1> K44A:
3484 0000129A F6C303              <1>       test   bl, LEFT_SHIFT+RIGHT_SHIFT ; NOT 101 KBD, SHIFT KEY DOWN?
3485 0000129D 743C                <1>       jz     short K45C          ; NO, TRANSLATE TO '*' CHARACTER
3486                              <1>       ;
3487                              <1>       ;----- ISSUE INTERRUPT TO INDICATE PRINT SCREEN FUNCTION
3488                              <1> K44B:
3489 0000129F B0AE                <1>       mov    al, ENA_KBD         ; INSURE KEYBOARD IS ENABLED
3490 000012A1 E81B010000          <1>       call   SHIP_IT                   ; EXECUTE ENABLE
3491 000012A6 B020                <1>       mov    al, EOI                   ; END OF CURRENT INTERRUPT
3492 000012A8 E620                <1>       out    20h, al ;out INTA00, al   ; SO FURTHER THINGS CAN HAPPEN
3493                              <1>       ; Print Screen !!!         ; ISSUE PRINT SCREEN INTERRUPT (INT 05h)
3494                              <1>       ;PUSH  BP                  ; SAVE POINTER
3495                              <1>       ;INT   5H                  ; ISSUE PRINT SCREEN INTERRUPT
3496                              <1>       ;POP   BP                  ; RESTORE POINTER
3497 000012AA 8025[905E0000]FC    <1>         and     byte [KB_FLAG_3], ~(LC_E0+LC_E1) ; ZERO OUT THESE FLAGS
3498 000012B1 E90CFEFFFF          <1>         jmp     K27                     ; GO BACK WITHOUT EOI OCCURRING
3499                              <1>       ;
3500                              <1>       ;----- HANDLE IN-CORE KEYS
```

```
                        <1> K45:                            ; NOT-PRINT-SCREEN
000012B6 3C3A           <1>         cmp    al, 58           ; TEST FOR IN-CORE AREA
000012B8 7734           <1>         ja     short K46        ; JUMP IF NOT
000012BA 3C35           <1>         cmp    al, 53           ; IS THIS THE '/' KEY?
000012BC 7505           <1>         jne    short K45A       ; NO, JUMP
000012BE F6C702         <1>         test   bh, LC_E0        ; WAS THE LAST CODE THE MARKER?
000012C1 7518           <1>         jnz    short K45C       ; YES, TRANSLATE TO CHARACTER
                        <1> K45A:
000012C3 B91A000000     <1>         mov    ecx, 26                  ; LENGHT OF SEARCH
000012C8 BF[5A5D0000]   <1>         mov    edi, K30+10      ; POINT TO TABLE OF A-Z CHARS
000012CD F2AE           <1>         repne  scasb            ; IS THIS A LETTER KEY?
                        <1>                 ; 20/02/2015
000012CF 7505           <1>         jne    short K45B           ; NO, SYMBOL KEY
                        <1>         ;
000012D1 F6C340         <1>         test   bl, CAPS_STATE           ; ARE WE IN CAPS_LOCK?
000012D4 750C           <1>         jnz    short K45D       ; TEST FOR SURE
                        <1> K45B:
000012D6 F6C303         <1>         test   bl, LEFT_SHIFT+RIGHT_SHIFT ; ARE WE IN SHIFT STATE?
000012D9 750C           <1>         jnz    short K45E       ; YES, UPPERCASE
                        <1>                                  ; NO, LOWERCASE
                        <1> K45C:
000012DB BB[DC5D0000]   <1>         mov    ebx, K10         ; TRANSLATE TO LOWERCASE LETTERS
000012E0 EB51           <1>         jmp    short K56
                        <1> K45D:                           ; ALMOST-CAPS-STATE
000012E2 F6C303         <1>         test   bl, LEFT_SHIFT+RIGHT_SHIFT ; CL ON. IS SHIFT ON, TOO?
000012E5 75F4           <1>         jnz    short K45C       ; SHIFTED TEMP OUT OF CAPS STATE
                        <1> K45E:
000012E7 BB[345E0000]   <1>         mov    ebx, K11         ; TRANSLATE TO UPPER CASE LETTERS
000012EC EB45           <1> K45F:   jmp    short K56
                        <1>         ;
                        <1>         ;----- TEST FOR KEYS F1 - F10
                        <1> K46:                            ; NOT IN-CORE AREA
000012EE 3C44           <1>         cmp    al, 68           ; TEST FOR F1 - F10
                        <1>         ;ja    short K47        ; JUMP IF NOT
                        <1>         ;jmp   short K53        ; YES, GO DO FN KEY PROCESS
000012F0 7635           <1>         jna    short K53
                        <1>         ;
                        <1>         ;----- HANDLE THE NUMERIC PAD KEYS
                        <1> K47:                            ; NOT F1 - F10
000012F2 3C53           <1>         cmp    al, 83           ; TEST NUMPAD KEYS
000012F4 772D           <1>         ja     short K52        ; JUMP IF NOT
                        <1>         ;
                        <1>         ;----- KEYPAD KEYS, MUST TEST NUM LOCK FOR DETERMINATION
                        <1> K48:
000012F6 3C4A           <1>         cmp    al , 74                  ; SPECIAL CASE FOR MINUS
000012F8 74ED           <1>         je     short K45E       ; GO TRANSLATE
000012FA 3C4E           <1>         cmp    al , 78                  ; SPECIAL CASE FOR PLUS
000012FC 74E9           <1>         je     short K45E       ; GO TRANSLATE
000012FE F6C702         <1>         test   bh, LC_E0        ; IS THIS ONE OFTHE NEW KEYS?
00001301 750A           <1>         jnz    short K49        ; YES, TRANSLATE TO BASE STATE
                        <1>         ;
00001303 F6C320         <1>         test   bl, NUM_STATE    ; ARE WE IN NUM LOCK
00001306 7514           <1>         jnz    short K50        ; TEST FOR SURE
00001308 F6C303         <1>         test   bl, LEFT_SHIFT+RIGHT_SHIFT ; ARE WE IN SHIFT STATE?
                        <1>         ;jnz   short K51        ; IF SHIFTED, REALLY NUM STATE
0000130B 75DA           <1>         jnz    short K45E
                        <1>         ;
                        <1>         ;----- BASE CASE FOR KEYPAD
                        <1> K49:
0000130D 3C4C           <1>         cmp    al, 76           ; SPECIAL CASE FOR BASE STATE 5
0000130F 7504           <1>         jne    short K49A       ; CONTINUE IF NOT KEYPAD 5
00001311 B0F0           <1>         mov    al, 0F0h         ; SPECIAL ASCII CODE
00001313 EB40           <1>         jmp    short K57        ; BUFFER FILL
                        <1> K49A:
00001315 BB[DC5D0000]   <1>         mov    ebx, K10         ; BASE CASE TABLE
0000131A EB27           <1>         jmp    short K64        ; CONVERT TO PSEUDO SCAN
                        <1>         ;
                        <1>         ;----- MIGHT BE NUM LOCK, TEST SHIFT STATUS
                        <1> K50:                            ; ALMOST-NUM-STATE
0000131C F6C303         <1>           test    bl, LEFT_SHIFT+RIGHT_SHIFT
0000131F 75EC           <1>         jnz    short K49        ; SHIFTED TEMP OUT OF NUM STATE
00001321 EBC4           <1> K51:    jmp    short K45E       ; REALLY NUM STATE
                        <1>         ;
                        <1>         ;----- TEST FOR THE NEW KEYS ON WT KEYBOARDS
                        <1> K52:                            ; NOT A NUMPAD KEY
00001323 3C56           <1>         cmp    al, 86           ; IS IT THE NEW WT KEY?
                        <1>         ;jne   short K53        ; JUMP IF NOT
                        <1>         ;jmp   short K45B       ; HANDLE WITH REST OF LETTER KEYS
00001325 74AF           <1>         je     short K45B
                        <1>         ;
                        <1>         ;----- MUST BE F11 OR F12
                        <1> K53:                            ; F1 - F10 COME HERE, TOO
00001327 F6C303         <1>         test   bl, LEFT_SHIFT+RIGHT_SHIFT ; TEST SHIFT STATE
0000132A 74E1           <1>         jz     short K49        ; JUMP, LOWER CASE PSEUDO SC'S
                        <1>                 ; 20/02/2015
0000132C BB[345E0000]   <1>         mov    ebx, K11         ; UPPER CASE PSEUDO SCAN CODES
00001331 EB10           <1>         jmp    short K64        ; TRANSLATE SCAN
                        <1>         ;
                        <1>         ;----- TRANSLATE THE CHARACTER
                        <1> K56:                            ; TRANSLATE-CHAR
00001333 FEC8           <1>         dec    al               ; CONVERT ORIGIN
00001335 D7             <1>         xlat                    ; CONVERT THE SCAN CODE TO ASCII
00001336 F605[905E0000]02   <1>     test   byte [KB_FLAG_3], LC_E0   ; IS THIS A NEW KEY?
0000133D 7416           <1>         jz     short K57        ; NO, GO FILL BUFFER
0000133F B4E0           <1>         mov    ah, MC_E0        ; YES, PUT SPECIAL MARKER IN AH
00001341 EB12           <1>         jmp    short K57        ; PUT IT INTO THE BUFFER
                        <1>         ;
                        <1>         ;----- TRANSLATE SCAN FOR PSEUDO SCAN CODES
                        <1> K64:                            ; TRANSLATE-SCAN-ORGD
00001343 FEC8           <1>         dec    al               ; CONVERT ORIGIN
00001345 D7             <1>                 xlat                        ; CTL TABLE SCAN
00001346 88C4           <1>         mov    ah, al           ; PUT VALUE INTO AH
00001348 B000           <1>         mov    al, 0            ; ZERO ASCII CODE
```

```
3604 0000134A F605[905E0000]02    <1>       test   byte [KB_FLAG_3], LC_E0   ; IS THIS A NEW KEY?
3605 00001351 7402                <1>       jz     short K57           ; NO, GO FILL BUFFER
3606 00001353 B0E0                <1>       mov    al, MC_E0           ; YES, PUT SPECIAL MARKER IN AL
3607                              <1>       ;
3608                              <1>       ;----- PUT CHARACTER INTO BUFFER
3609                              <1> K57:                             ; BUFFER_FILL
3610 00001355 3CFF                <1>       cmp    al, -1              ; IS THIS AN IGNORE CHAR
3611                              <1>         ;je  short K59           ; YES, DO NOTHING WITH IT
3612 00001357 0F8459FDFFFF        <1>       je      K26                ; YES, DO NOTHING WITH IT
3613 0000135D 80FCFF              <1>       cmp    ah, -1              ; LOOK FOR -1 PSEUDO SCAN
3614                              <1>         ;jne short K61           ; NEAR_INTERRUPT_RETURN
3615 00001360 0F8450FDFFFF        <1>       je      K26                ; INTERRUPT_RETURN
3616                              <1> ;K59:                            ; NEAR_INTERRUPT_RETURN
3617                              <1> ;     jmp    K26                 ; INTERRUPT_RETURN
3618                              <1>
3619                              <1> _K60: ; 29/01/2016
3620 00001366 80FC68              <1>       cmp    ah, 68h      ; ALT + F1 key
3621 00001369 721F                <1>       jb     short K61
3622 0000136B 80FC6F              <1>       cmp    ah, 6Fh ; ALT + F8 key
3623 0000136E 771A                <1>       ja     short K61
3624                              <1>       ;
3625 00001370 8A1D[4E520100]      <1>       mov    bl, [ACTIVE_PAGE]
3626 00001376 80C368              <1>       add    bl, 68h
3627 00001379 38E3                <1>       cmp    bl, ah
3628 0000137B 740D                <1>       je     short K61
3629 0000137D 6650                <1>       push   ax
3630 0000137F 88E0                <1>       mov    al, ah
3631 00001381 2C68                <1>       sub    al, 68h
3632 00001383 E8F4050000          <1>       call   set_active_page
3633 00001388 6658                <1>       pop    ax
3634                              <1> K61:                             ; NOT-CAPS-STATE
3635 0000138A 8B1D[9E5E0000]      <1>       mov    ebx, [BUFFER_TAIL]  ; GET THE END POINTER TO THE BUFFER
3636 00001390 89DE                <1>       mov    esi, ebx            ; SAVE THE VALUE
3637 00001392 E857FAFFFF          <1>       call   _K4                 ; ADVANCE THE TAIL
3638 00001397 3B1D[9A5E0000]      <1>       cmp    ebx, [BUFFER_HEAD]  ; HAS THE BUFFER WRAPPED AROUND
3639 0000139D 740E                <1>       je     short K62           ; BUFFER_FULL_BEEP
3640 0000139F 668906              <1>       mov    [esi], ax           ; STORE THE VALUE
3641 000013A2 891D[9E5E0000]      <1>       mov    [BUFFER_TAIL], ebx  ; MOVE THE POINTER UP
3642 000013A8 E909FDFFFF          <1>       jmp    K26
3643                              <1>       ;;cli                      ; TURN OFF INTERRUPTS
3644                              <1>       ;;mov  al, EOI                  ; END OF INTERRUPT COMMAND
3645                              <1>       ;;out  INTA00, al          ; SEND COMMAND TO INTERRUPT CONTROL PORT
3646                              <1>       ;MOV   AL, ENA_KBD         ; INSURE KEYBOARD IS ENABLED
3647                              <1>       ;CALL  SHIP_IT                  ; EXECUTE ENABLE
3648                              <1>       ;MOV   AX, 9102H           ; MOVE IN POST CODE & TYPE
3649                              <1>       ;INT   15H                 ; PERFORM OTHER FUNCTION
3650                              <1>       ;;and  byte [KB_FLAG_3],~(LC_E0+LC_E1) ; RESET LAST CHAR H.C. FLAG
3651                              <1>       ;JMP   K27A                ; INTERRUPT_RETURN
3652                              <1>       ;;jmp   K27
3653                              <1>       ;
3654                              <1>       ;----- BUFFER IS FULL SOUND THE BEEPER
3655                              <1> K62:
3656 000013AD B020                <1>       mov    al, EOI                  ; ENABLE INTERRUPT CONTROLLER CHIP
3657 000013AF E620                <1>       out    INTA00, al
3658 000013B1 66B9A602            <1>       mov    cx, 678                  ; DIVISOR FOR 1760 HZ
3659 000013B5 B304                <1>       mov    bl, 4               ; SHORT BEEP COUNT (1/16 + 1/64 DELAY)
3660 000013B7 E8E5090000          <1>       call   beep                ; GO TO COMMON BEEP HANDLER
3661 000013BC E901FDFFFF          <1>       jmp     K27                ; EXIT
3662                              <1>
3663                              <1> SHIP_IT:
3664                              <1>       ;--------------------------------------------------------------------------------
3665                              <1>       ; SHIP_IT
3666                              <1>       ;      THIS ROUTINES HANDLES TRANSMISSION OF COMMAND AND DATA BYTES
3667                              <1>       ;      TO THE KEYBOARD CONTROLLER.
3668                              <1>       ;--------------------------------------------------------------------------------
3669                              <1>       ;
3670 000013C1 6650                <1>       push   ax                  ; SAVE DATA TO SEND
3671                              <1>
3672                              <1>       ;----- WAIT FOR COMMAND TO ACCEPTED
3673 000013C3 FA                  <1>       cli                        ; DISABLE INTERRUPTS TILL DATA SENT
3674                              <1>       ; xor  ecx, ecx            ; CLEAR TIMEOUT COUNTER
3675 000013C4 B900000100          <1>       mov    ecx, 10000h
3676                              <1> S10:
3677 000013C9 E464                <1>       in     al, STATUS_PORT           ; READ KEYBOARD CONTROLLER STATUS
3678 000013CB A802                <1>       test   al, INPT_BUF_FULL   ; CHECK FOR ITS INPUT BUFFER BUSY
3679 000013CD E0FA                <1>       loopnz S10                 ; WAIT FOR COMMAND TO BE ACCEPTED
3680                              <1>
3681 000013CF 6658                <1>       pop    ax                  ; GET DATA TO SEND
3682 000013D1 E664                <1>       out    STATUS_PORT, al           ; SEND TO KEYBOARD CONTROLLER
3683 000013D3 FB                  <1>       sti                        ; ENABLE INTERRUPTS AGAIN
3684 000013D4 C3                  <1>       retn                       ; RETURN TO CALLER
3685                              <1>
3686                              <1> SND_DATA:
3687                              <1>       ; ------------------------------------------------------------------------------
3688                              <1>       ; SND_DATA
3689                              <1>       ;      THIS ROUTINES HANDLES TRANSMISSION OF COMMAND AND DATA BYTES
3690                              <1>       ;      TO THE KEYBOARD AND RECEIPT OF ACKNOWLEDGEMENTS. IT ALSO
3691                              <1>       ;      HANDLES ANY RETRIES IF REQUIRED
3692                              <1>       ; ------------------------------------------------------------------------------
3693                              <1>       ;
3694 000013D5 6650                <1>       push   ax                  ; SAVE REGISTERS
3695 000013D7 6653                <1>       push   bx
3696 000013D9 51                  <1>       push   ecx
3697 000013DA 88C7                <1>       mov    bh, al              ; SAVE TRANSMITTED BYTE FOR RETRIES
3698 000013DC B303                <1>       mov    bl, 3               ; LOAD RETRY COUNT
3699                              <1> SD0:
3700 000013DE FA                  <1>       cli                        ; DISABLE INTERRUPTS
3701 000013DF 8025[8F5E0000]CF    <1>       and    byte [KB_FLAG_2], ~(KB_FE+KB_FA) ; CLEAR ACK AND RESEND FLAGS
3702                              <1>       ;
3703                              <1>       ;----- WAIT FOR COMMAND TO BE ACCEPTED
3704 000013E6 B900000100          <1>       mov    ecx, 10000h         ; MAXIMUM WAIT COUNT
3705                              <1> SD5:
3706 000013EB E464                <1>       in     al, STATUS_PORT           ; READ KEYBOARD PROCESSOR STATUS PORT
```

```
3707 000013ED A802               <1>       test   al, INPT_BUF_FULL   ; CHECK FOR ANY PENDING COMMAND
3708 000013EF E0FA               <1>       loopnz SD5                 ; WAIT FOR COMMAND TO BE ACCEPTED
3709                             <1>       ;
3710 000013F1 88F8               <1>       mov    al, bh              ; REESTABLISH BYTE TO TRANSMIT
3711 000013F3 E660               <1>       out    PORT_A, al          ; SEND BYTE
3712 000013F5 FB                 <1>       sti                        ; ENABLE INTERRUPTS
3713                             <1>       ;mov   cx, 01A00h          ; LOAD COUNT FOR 10 ms+
3714 000013F6 B9FFFF0000         <1>       mov    ecx, 0FFFFh
3715                             <1> SD1:
3716 000013FB F605[8F5E0000]30   <1>       test   byte [KB_FLAG_2], KB_FE+KB_FA ; SEE IF EITHER BIT SET
3717 00001402 750F               <1>       jnz    short SD3           ; IF SET, SOMETHING RECEIVED GO PROCESS
3718 00001404 E2F5               <1>       loop   SD1                 ; OTHERWISE WAIT
3719                             <1> SD2:
3720 00001406 FECB               <1>       dec    bl                  ; DECREMENT RETRY COUNT
3721 00001408 75D4               <1>       jnz    short SD0           ; RETRY TRANSMISSION
3722 0000140A 800D[8F5E0000]80   <1>       or     byte [KB_FLAG_2], KB_ERR ; TURN ON TRANSMIT ERROR FLAG
3723 00001411 EB09               <1>       jmp    short SD4           ; RETRIES EXHAUSTED FORGET TRANSMISSION
3724                             <1> SD3:
3725 00001413 F605[8F5E0000]10   <1>       test   byte [KB_FLAG_2], KB_FA ; SEE IF THIS IS AN ACKNOWLEDGE
3726 0000141A 74EA               <1>       jz     short SD2           ; IF NOT, GO RESEND
3727                             <1> SD4:
3728 0000141C 59                 <1>       pop    ecx                 ; RESTORE REGISTERS
3729 0000141D 665B               <1>       pop    bx
3730 0000141F 6658               <1>       pop    ax
3731 00001421 C3                 <1>       retn                       ; RETURN, GOOD TRANSMISSION
3732                             <1>
3733                             <1> SND_LED:
3734                             <1>       ; ------------------------------------------------------------------------------
3735                             <1>       ; SND_LED
3736                             <1>       ;      THIS ROUTINES TURNS ON THE MODE INDICATORS.
3737                             <1>       ;
3738                             <1>       ;------------------------------------------------------------------------------
3739                             <1>       ;
3740 00001422 FA                 <1>       cli                        ; TURN OFF INTERRUPTS
3741 00001423 F605[8F5E0000]40   <1>       test   byte [KB_FLAG_2], KB_PR_LED ; CHECK FOR MODE INDICATOR UPDATE
3742 0000142A 755F               <1>       jnz    short SL1           ; DON'T UPDATE AGAIN IF UPDATE UNDERWAY
3743                             <1>       ;
3744 0000142C 800D[8F5E0000]40   <1>       or     byte [KB_FLAG_2], KB_PR_LED ; TURN ON UPDATE IN PROCESS
3745 00001433 B020               <1>       mov    al, EOI                    ; END OF INTERRUPT COMMAND
3746 00001435 E620               <1>       out    20h, al ;out INTA00, al    ; SEND COMMAND TO INTERRUPT CONTROL PORT
3747 00001437 EB11               <1>       jmp    short SL0           ; GO SEND MODE INDICATOR COMMAND
3748                             <1> SND_LED1:
3749 00001439 FA                 <1>       cli                        ; TURN OFF INTERRUPTS
3750 0000143A F605[8F5E0000]40   <1>       test   byte [KB_FLAG_2], KB_PR_LED ; CHECK FOR MODE INDICATOR UPDATE
3751 00001441 7548               <1>       jnz    short SL1           ; DON'T UPDATE AGAIN IF UPDATE UNDERWAY
3752                             <1>       ;
3753 00001443 800D[8F5E0000]40   <1>       or     byte [KB_FLAG_2], KB_PR_LED ; TURN ON UPDATE IN PROCESS
3754                             <1> SL0:
3755 0000144A B0ED               <1>       mov    al, LED_CMD         ; LED CMD BYTE
3756 0000144C E884FFFFFF         <1>       call   SND_DATA            ; SEND DATA TO KEYBOARD
3757 00001451 FA                 <1>       cli
3758 00001452 E836000000         <1>       call   MAKE_LED            ; GO FORM INDICATOR DATA BYTE
3759 00001457 8025[8F5E0000]F8   <1>       and    byte [KB_FLAG_2], 0F8h    ; ~KB_LEDS ; CLEAR MODE INDICATOR BITS
3760 0000145E 0805[8F5E0000]     <1>       or     [KB_FLAG_2], al     ; SAVE PRESENT INDICATORS FOR NEXT TIME
3761 00001464 F605[8F5E0000]80   <1>       test   byte [KB_FLAG_2], KB_ERR ; TRANSMIT ERROR DETECTED
3762 0000146B 750F               <1>       jnz    short SL2           ; IF SO, BYPASS SECOND BYTE TRANSMISSION
3763                             <1>       ;
3764 0000146D E863FFFFFF         <1>       call   SND_DATA            ; SEND DATA TO KEYBOARD
3765 00001472 FA                 <1>       cli                        ; TURN OFF INTERRUPTS
3766 00001473 F605[8F5E0000]80   <1>       test   byte [KB_FLAG_2], KB_ERR ; TRANSMIT ERROR DETECTED
3767 0000147A 7408               <1>       jz     short SL3           ; IF NOT, DON'T SEND AN ENABLE COMMAND
3768                             <1> SL2:
3769 0000147C B0F4               <1>       mov    al, KB_ENABLE       ; GET KEYBOARD CSA ENABLE COMMAND
3770 0000147E E852FFFFFF         <1>       call   SND_DATA            ; SEND DATA TO KEYBOARD
3771 00001483 FA                 <1>       cli                        ; TURN OFF INTERRUPTS
3772                             <1> SL3:
3773 00001484 8025[8F5E0000]3F   <1>       and    byte [KB_FLAG_2], ~(KB_PR_LED+KB_ERR) ; TURN OFF MODE INDICATOR
3774                             <1> SL1:                               ; UPDATE AND TRANSMIT ERROR FLAG
3775 0000148B FB                 <1>       sti                        ; ENABLE INTERRUPTS
3776 0000148C C3                 <1>       retn                       ; RETURN TO CALLER
3777                             <1>
3778                             <1> MAKE_LED:
3779                             <1>       ;------------------------------------------------------------------------------
3780                             <1>       ; MAKE_LED
3781                             <1>       ;      THIS ROUTINES FORMS THE DATA BYTE NECESSARY TO TURN ON/OFF
3782                             <1>       ;      THE MODE INDICATORS.
3783                             <1>       ;------------------------------------------------------------------------------
3784                             <1>       ;
3785                             <1>       ;push  cx                  ; SAVE CX
3786 0000148D A0[8D5E0000]       <1>       mov    al, [KB_FLAG]       ; GET CAPS & NUM LOCK INDICATORS
3787 00001492 2470               <1>       and    al, CAPS_STATE+NUM_STATE+SCROLL_STATE ; ISOLATE INDICATORS
3788                             <1>       ;mov   cl, 4               ; SHIFT COUNT
3789                             <1>       ;rol   al, cl              ; SHIFT BITS OVER TO TURN ON INDICATORS
3790 00001494 C0C004             <1>       rol    al, 4 ; 20/02/2015
3791 00001497 2407               <1>       and    al, 07h                  ; MAKE SURE ONLY MODE BITS ON
3792                             <1>       ;pop   cx
3793 00001499 C3                 <1>       retn                       ; RETURN TO CALLER
3794                             <1>
3795                             <1> ; % include 'kybdata.s'   ; KEYBOARD DATA
3796                             <1>
3797                             <1>
3798                             <1> ; /// End Of KEYBOARD FUNCTIONS ///
3799
3800                                 %include 'video.s' ; 07/03/2015
3801                             <1> ; ****************************************************************************
3802                             <1> ; TRDOS386.ASM (TRDOS 386 Kernel) - v2.0.0 - video.s
3803                             <1> ; ----------------------------------------------------------------------------
3804                             <1> ; Last Update: 09/12/2017
3805                             <1> ; ----------------------------------------------------------------------------
3806                             <1> ; Beginning: 16/01/2016
3807                             <1> ; ----------------------------------------------------------------------------
3808                             <1> ; Assembler: NASM version 2.11 (trdos386.s)
3809                             <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> ; video.inc (13/08/2015)
                        <1> ;
                        <1> ; Derived from 'IBM PC-AT' BIOS source code (1985)
                        <1> ; ********************************************************************************
                        <1>
                        <1> ; Retro UNIX 386 v1 Kernel - VIDEO.INC
                        <1> ; Last Modification: 13/08/2015
                        <1> ;             (Video Data is in 'VIDATA.INC')
                        <1> ;
                        <1> ; ///////// VIDEO (CGA) FUNCTIONS ///////////////
                        <1>
                        <1> ; 16/01/2016 (32 bit modifications, TRDOS386 - TRDOS v2.0, video.s)
                        <1> ; INT 31H (TRDOS 386) = INT 10H (IBM PC/AT REAL MODE)
                        <1>
                        <1> ; IBM PC-AT BIOS Source Code
                        <1> ; TITLE VIDEO1 --- 06/10/85  VIDEO DISPLAY BIOS
                        <1>
0000149A                <1> _int10h:
                        <1>       ; 23/03/2016
                        <1>       ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
0000149A 9C             <1>       pushfd
0000149B 0E             <1>       push   cs
0000149C E851000000     <1>       call   VIDEO_IO_1
000014A1 C3             <1>       retn
                        <1>
                        <1> ;--- INT 10 H ----------------------------------------------------------------
                        <1> ; VIDEO_IO                                                :
                        <1> ;     THESE ROUTINES PROVIDE THE CRT DISPLAY INTERFACE                 :
                        <1> ;     THE FOLLOWING FUNCTIONS ARE PROVIDED:                            :
                        <1> ;                                                          :
                        <1> ;    (AH)= 00H     SET MODE (AL) CONTAINS MODE VALUE                     :
                        <1> ;           (AL) = 00H  40X25 BW MODE (POWER ON DEFAULT)                 :
                        <1> ;           (AL) = 01H  40X25 COLOR                                      :
                        <1> ;           (AL) = 02H  80X25 BW                                         :
                        <1> ;           (AL) = 03H  80X25 COLOR                                      :
                        <1> ;                       GRAPHICS MODES                                   :
                        <1> ;           (AL) = 04H  320X200 COLOR                          :
                        <1> ;           (AL) = 05H  320X200 BW MODE                                  :
                        <1> ;           (AL) = 06H  640X200 BW MODE                                  :
                        <1> ;           (AL) = 07H   80X25 MONOCHROME (USED INTERNAL TO VIDEO ONLY)      :
                        <1> ;           *** NOTES -BW MODES OPERATE SAME AS COLOR MODES, BUT COLOR    :
                        <1> ;                      BURST IS NOT ENABLED                              :
                        <1> ;                     -CURSOR IS NOT DISPLAYED IN GRAPHICS MODE          :
                        <1> ;    (AH)= 01H     SET CURSOR TYPE                                       :
                        <1> ;           (CH) = BITS 4-0 = START LINE FOR CURSOR                      :
                        <1> ;                  ** HARDWARE WILL ALWAYS CAUSE BLINK                   :
                        <1> ;                  ** SETTING BIT 5 OR 6 WILL CAUSE ERRATIC BLINKING     :
                        <1> ;                     OR NO CURSOR AT ALL                                :
                        <1> ;           (CL) = BITS 4-0 = END LINE FOR CURSOR                        :
                        <1> ;    (AH)= 02H     SET CURSOR POSITION                                   :
                        <1> ;           (DH,DL) = ROW,COLUMN  (00H,00H) IS UPPER LEFT                :
                        <1> ;           (BH) = A PAGE NUMBER (MUST BE 00H FOR GRAPHICS MODES)            :
                        <1> ;    (AH)= 03H     READ CURSOR POSITION                                  :
                        <1> ;           (BH) = PAGE NUMBER (MUST BE 00H FOR GRAPHICS MODES)          :
                        <1> ;           ON EXIT (DH,DL) = ROW,COLUMN OF CURRENT CURSOR                   :
                        <1> ;                   (CH,CL) = CURSOR MODE CURRENTLY SET                  :
                        <1> ;    (AH)= 04H     READ LIGHT PEN POSITION                               :
                        <1> ;           ON EXIT:                                           :
                        <1> ;           (AH) = 00H -- LIGHT PEN SWITCH NOT DOWN/NOT TRIGGERED            :
                        <1> ;           (AH) = 01H -- VALID LIGHT PEN VALUE IN REGISTERS             :
                        <1> ;                  (DH,DL) = ROW,COLUMN OF CHARACTER LP POSITION             :
                        <1> ;                  (CH) = RASTER LINE (0-199)                            :
                        <1> ;                  (BX) = PIXEL COLUMN (0-319,639)                       :
                        <1> ;    (AH)= 05H     SELECT ACTIVE DISPLAY PAGE (VALID ONLY FOR ALPHA MODES)          :
                        <1> ;           (AL) = NEW PAGE VALUE (0-7 FOR MODES 0&1, 0-3 FOR MODES 2&3)     :
                        <1> ;    (AH)= 06H     SCROLL ACTIVE PAGE UP                                 :
                        <1> ;           (AL) = NUMBER OF LINES. ( LINES BLANKED AT BOTTOM OF WINDOW )    :
                        <1> ;                  (AL) = 00H MEANS BLANK ENTIRE WINDOW                  :
                        <1> ;           (CH,CL) = ROW,COLUMN OF UPPER LEFT CORNER OF SCROLL           :
                        <1> ;           (DH,DL) = ROW,COLUMN OF LOWER RIGHT CORNER OF SCROLL          :
                        <1> ;           (BH) = ATTRIBUTE TO BE USED ON BLANK LINE                     :
                        <1> ;    (AH)= 07H     SCROLL ACTIVE PAGE DOWN                                   :
                        <1> ;           (AL) = NUMBER OF LINES, INPUT LINES BLANKED AT TOP OF WINDOW      :
                        <1> ;                  (AL) = 00H MEANS BLANK ENTIRE WINDOW                  :
                        <1> ;           (CH,CL) = ROW,COLUMN OF UPPER LEFT CORNER OF SCROLL           :
                        <1> ;           (DH,DL) = ROW,COLUMN OF LOWER RIGHT CORNER OF SCROLL          :
                        <1> ;           (BH) = ATTRIBUTE TO BE USED ON BLANK LINE                     :
                        <1> ;                                                            :
                        <1> ;  CHARACTER HANDLING ROUTINES                                            :
                        <1> ;                                                            :
                        <1> ;    (AH)= 08H     READ ATTRIBUTE/CHARACTER AT CURRENT CURSOR POSITION       :
                        <1> ;           (BH) = DISPLAY PAGE (VALID FOR ALPHA MODES ONLY)              :
                        <1> ;           ON EXIT:                                         :
                        <1> ;           (AL) = CHAR READ                                 :
                        <1> ;           (AH) = ATTRIBUTE OF CHARACTER READ (ALPHA MODES ONLY)             :
                        <1> ;    (AH)= 09H     WRITE ATTRIBUTE/CHARACTER AT CURRENT CURSOR POSITION       :
                        <1> ;           (BH) = DISPLAY PAGE (VALID FOR ALPHA MODES ONLY)              :
                        <1> ;           (CX) = COUNT OF CHARACTERS TO WRITE                           :
                        <1> ;           (AL) = CHAR TO WRITE                                          :
                        <1> ;           (BL) = ATTRIBUTE OF CHARACTER (ALPHA)/COLOR OF CHAR (GRAPHICS)    :
                        <1> ;                    SEE NOTE ON WRITE DOT FOR BIT 7 OF BL = 1.          :
                        <1> ;    (AH) = 0AH    WRITE CHARACTER ONLY AT CURRENT CURSOR POSITION                  :
                        <1> ;           (BH) = DISPLAY PAGE (VALID FOR ALPHA MODES ONLY)              :
                        <1> ;           (CX) = COUNT OF CHARACTERS TO WRITE                           :
                        <1> ;           (AL) = CHAR TO WRITE                                          :
                        <1> ;                  NOTE: USE FUNCTION (AH)= 09H IN GRAPHICS MODES          :
                        <1> ;    FOR READ/WRITE CHARACTER INTERFACE WHILE IN GRAPHICS MODE, THE         :
                        <1> ;           CHARACTERS ARE FORMED FROM A CHARACTER GENERATOR IMAGE         :
                        <1> ;           MAINTAINED IN THE SYSTEM ROM. ONLY THE 1ST 128 CHARS       :
```

```
3913                                    <1> ;           ARE CONTAINED THERE. TO READ/WRITE THE SECOND 128 CHARS,   :
3914                                    <1> ;           THE USER MUST INITIALIZE THE POINTER AT INTERRUPT 1FH            :
3915                                    <1> ;           (LOCATION 0007CH) TO POINT TO THE 1K BYTE TABLE CONTAINING :
3916                                    <1> ;           THE CODE POINTS FOR THE SECOND 128 CHARS (128-255).        :
3917                                    <1> ;    FOR WRITE CHARACTER INTERFACE IN GRAPHICS MODE, THE REPLICATION FACTOR  :
3918                                    <1> ;           CONTAINED IN (CX) ON ENTRY WILL PRODUCE VALID RESULTS ONLY :
3919                                    <1> ;           FOR CHARACTERS CONTAINED ON THE SAME ROW. CONTINUATION TO  :
3920                                    <1> ;           SUCCEEDING LINES WILL NOT PRODUCE CORRECTLY.                :
3921                                    <1> ;                                                                      :
3922                                    <1> ;    GRAPHICS INTERFACE                                                :
3923                                    <1> ;    (AH)= 0BH     SET COLOR PALETTE                                   :
3924                                    <1> ;           (BH) = PALETTE COLOR ID BEING SET (0-127)                  :
3925                                    <1> ;           (BL) = COLOR VALUE TO BE USED WITH THAT COLOR ID           :
3926                                    <1> ;                  NOTE: FOR THE CURRENT COLOR CARD, THIS ENTRY POINT HAS    :
3927                                    <1> ;                          MEANING ONLY FOR 320X200 GRAPHICS.          :
3928                                    <1> ;                  COLOR ID = 0 SELECTS THE BACKGROUND COLOR (0-15)           :
3929                                    <1> ;                  COLOR ID = 1 SELECTS THE PALETTE TO BE USED:        :
3930                                    <1> ;                          0 = GREEN(1)/RED(2)/YELLOW(3)               :
3931                                    <1> ;                          1 = CYAN(1)/MAGENTA(2)/WHITE(3)                 :
3932                                    <1> ;                  IN 40X25 OR 80X25 ALPHA MODES, THE VALUE SET FOR    :
3933                                    <1> ;                          PALETTE COLOR 0 INDICATES THE BORDER COLOR  :
3934                                    <1> ;                          TO BE USED (VALUES 0-31, WHERE 16-31 SELECT :
3935                                    <1> ;                          THE HIGH INTENSITY BACKGROUND SET.          :
3936                                    <1> ;    (AH)= 0CH     WRITE DOT                                           :
3937                                    <1> ;           (DX) = ROW NUMBER                                  :
3938                                    <1> ;           (CX) = COLUMN NUMBER                                       :
3939                                    <1> ;           (AL) = COLOR VALUE                                 :
3940                                    <1> ;                   IF BIT 7 OF AL = 1, THEN THE COLOR VALUE IS EXCLUSIVE    :
3941                                    <1> ;                   ORed WITH THE CURRENT CONTENTS OF THE DOT           :
3942                                    <1> ;    (AH)= ODH     READ DOT                                            :
3943                                    <1> ;           (DX) = ROW NUMBER                                  :
3944                                    <1> ;           (CX) = COLUMN NUMBER                                       :
3945                                    <1> ;           (AL) = RETURNS THE DOT READ                                :
3946                                    <1> ;                                                                      :
3947                                    <1> ;    ASCII TELETYPE ROUTINE FOR OUTPUT                                 :
3948                                    <1> ;                                                                      :
3949                                    <1> ;    (AH)= 0EH     WRITE TELETYPE TO ACTIVE PAGE                               :
3950                                    <1> ;           (AL) = CHAR TO WRITE                                       :
3951                                    <1> ;           (BL) = FOREGROUND COLOR IN GRAPHICS MODE                   :
3952                                    <1> ;           NOTE -- SCREEN WIDTH IS CONTROLLED BY PREVIOUS MODE SET          :
3953                                    <1> ;    (AH)= 0FH     CURRENT VIDEO STATE                                 :
3954                                    <1> ;           RETURNS THE CURRENT VIDEO STATE                            :
3955                                    <1> ;           (AL) = MODE CURRENTLY SET ( SEE (AH)=00H FOR EXPLANATION)  :
3956                                    <1> ;           (AH) = NUMBER OR CHARACTER COLUMNS ON SCREEN                :
3957                                    <1> ;           (BH) = CURRENT ACTIVE DISPLAY PAGE                         :
3958                                    <1> ;    (AH)= 10H     RESERVED                                            :
3959                                    <1> ;    (AH)= 11H     RESERVED                                            :
3960                                    <1> ;    (AH)= 12H     RESERVED                                            :
3961                                    <1> ;    (AH)= 13H     WRITE STRING                                        :
3962                                    <1> ;                  ES:BP  -  POINTER T0 STRING TO BE WRITTEN           :
3963                                    <1> ;                  CX     -  LENGTH OF CHARACTER STRING TO WRITTEN            :
3964                                    <1> ;                  DX     -  CURSOR POSITION FOR STRING TO BE WRITTEN  :
3965                                    <1> ;                  BH     -  PAGE NUMBER                              :
3966                                    <1> ;           (AL)= 00H    WRITE CHARACTER STRING                       :
3967                                    <1> ;                  BL     -  ATTRIBUTE                                :
3968                                    <1> ;                  STRING IS  <CHAR,CHAR, ... ,CHAR>                  :
3969                                    <1> ;                  CURSOR NOT MOVED                            :
3970                                    <1> ;           (AL)= 01H    WRITE CHARACTER STRING AND MOVE CURSOR       :
3971                                    <1> ;                  BL     -  ATTRIBUTE                                :
3972                                    <1> ;                  STRING IS  <CHAR,CHAR, ... ,CHAR>                  :
3973                                    <1> ;                  CURSOR MOVED                                :
3974                                    <1> ;           (AL)= 02H    WRITE CHARACTER AND ATTRIBUTE STRING         :
3975                                    <1> ;                         (VALID FOR ALPHA MODES ONLY)                 :
3976                                    <1> ;                  STRING IS <CHAR,ATTR,CHAR,ATTR ..  ,CHAR,ATTR>           :
3977                                    <1> ;                  CURSOR IS NOT MOVED                         :
3978                                    <1> ;           (AL)= 03H WRITE CHARACTER AND ATTRIBUTE STRING AND MOVE CURSOR   :
3979                                    <1> ;                         (VALID FOR ALPHA MODES ONLY)                 :
3980                                    <1> ;                  STRING IS <CHAR,ATTR,CHAR,ATTR ..  ,CHAR,ATTR>           :
3981                                    <1> ;                  CURSOR IS MOVED                                    :
3982                                    <1> ;            NOTE:  CARRIAGE RETURN, LINE FEED, BACKSPACE, AND BELL ARE      :
3983                                    <1> ;                   TREATED AS COMMANDS RATHER THAN PRINTABLE CHARACTERS.    :
3984                                    <1> ;                                                                      :
3985                                    <1> ;    BX,CX,DX,SI,DI,BP,SP,DS,ES,SS PRESERVED DURING CALLS EXCEPT FOR         :
3986                                    <1> ;    BX,CX,DX RETURN VALUES ON FUNCTIONS 03H,04H,0DH AND 0FH. ON ALL CALLS   :
3987                                    <1> ;    AX IS MODIFIED.                                                    :
3988                                    <1> ;-----------------------------------------------------------------------------
3989                                    <1>
3990 000014A2 [4F150000]                <1> M1:   dd    SET_MODE     ; TABLE OF ROUTINES WITHIN VIDEO I/O
3991 000014A6 [B7180000]                <1>       dd    SET_CTYPE
3992 000014AA [EB180000]                <1>       dd    SET_CPOS
3993 000014AE [13190000]                <1>       dd    READ_CURSOR
3994                                    <1>       ;dd   VIDEO_RETURN ; READ_LPEN
3995 000014B2 [38150000]                <1>       dd    set_mode_ncm ; Set mode without clearing video memory
3996 000014B6 [59190000]                <1>       dd    ACT_DISP_PAGE
3997 000014BA [F0190000]                <1>       dd    SCROLL_UP
3998 000014BE [141B0000]                <1>       dd    SCROLL_DOWN
3999 000014C2 [951B0000]                <1>       dd    READ_AC_CURRENT
4000 000014C6 [ED1B0000]                <1>       dd    WRITE_AC_CURRENT
4001 000014CA [131C0000]                <1>       dd    WRITE_C_CURRENT
4002 000014CE [39250000]                <1>       dd    SET_COLOR
4003 000014D2 [A4250000]                <1>       dd    WRITE_DOT
4004 000014D6 [6F250000]                <1>       dd    READ_DOT
4005 000014DA [951C0000]                <1>       dd    WRITE_TTY
4006 000014DE [20150000]                <1>       dd    VIDEO_STATE
4007 000014E2 [EF2E0000]                <1>       dd    vga_pal_funcs ; 10/08/2016 (TRDOS 386)
4008 000014E6 [A52A0000]                <1>       dd    font_setup    ; 10/07/2016 (TRDOS 386)
4009 000014EA [54150000]                <1>       dd    VIDEO_RETURN ; RESERVED
4010 000014EE [021E0000]                <1>       dd    WRITE_STRING  ; 23/06/2016 (TRDOS 386)
4011                                    <1> M1L   EQU   $ - M1
4012                                    <1>
4013                                    <1> ; 14/01/2017
4014                                    <1> ; 02/01/2017
4015                                    <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[AC5F0100]       <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[4E520100]     <1>       mov    bh, [ACTIVE_PAGE] ; GET CURRENT ACTIVE PAGE
00001531 FA                 <1>       cli    ; 02/01/2017
00001532 5D                 <1>       pop    ebp          ; RECOVER REGISTERS
00001533 5F                 <1>       pop    edi
00001534 5E                 <1>       pop    esi
00001535 59                 <1>       pop    ecx          ; DISCARD SAVED BX
00001536 EB26               <1>       jmp    short M15    ; RETURN TO CALLER
                            <1>
                            <1> set_mode_ncm:
                            <1>       ; 04/07/2016 - TRDOS 386 (TRDOS v2.0)
                            <1>       ; set mode without clearing the video memory
                            <1>       ; (ony for graphics modes)
00001538 3C07               <1>       cmp    al, 7 ; IBM PC CGA modes
0000153A 7613               <1>       jna    short SET_MODE ; normal function (clear)
                            <1>       ; do not clear memory
0000153C A2[BB5F0100]       <1>       mov    [noclearmem], al ; > 0
00001541 E81F000000         <1>       call   _set_mode
00001546 C605[BB5F0100]00   <1>       mov    byte [noclearmem], 0
0000154D EB05               <1>         jmp     short VIDEO_RETURN
                            <1>
                            <1>       ; 10/08/2016
                            <1>       ; 08/08/2016
                            <1>       ; 30/07/2016
                            <1>       ; 29/07/2016
                            <1>       ; 27/07/2016
                            <1>       ; 26/07/2016
                            <1>       ; 25/07/2016
                            <1>       ; 23/07/2016
                            <1>       ; 18/07/2016
                            <1>       ; 02/07/2016
                            <1>       ; 26/06/2016
                            <1>       ; 24/06/2016
                            <1>       ; 29/05/2016 - TRDOS 386 (TRDOS v2.0)
                            <1> SET_MODE:
                            <1>       ; For 32 bit TRDOS and Retro UNIX 386:
                            <1>       ;     valid video mode: 03h only!
                            <1>       ;     (VGA modes will be selected with another routine)
                            <1>       ;
                            <1>       ; set_txt_mode ; 80*25 (16 fore colors, 8 back colors)
```

```
                            <1>
                            <1> ;--------------------------------------------------
                            <1> ; SET MODE                                      :
                            <1> ;     THIS ROUTINE INITIALIZES THE ATTACHMENT TO    :
                            <1> ;     THE SELECTED MODE, THE SCREEN IS BLANKED.     :
                            <1> ; INPUT                                          :
                            <1> ;     (AL) - MODE SELECTED (RANGE 0-7)          :
                            <1> ; OUTPUT                                        :
                            <1> ;     NONE                                      :
                            <1> ;--------------------------------------------------
                            <1>
0000154F E811000000         <1>       call  _set_mode ; 24/06/2016 (set_txt_mode)
                            <1>
                            <1> ; 12/05/2016
                            <1> ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
                            <1>
                            <1> ;-----      NORMAL RETURN FROM ALL VIDEO RETURNS
                            <1>
                            <1> VIDEO_RETURN:
00001554 A1[AC5F0100]       <1>       mov   eax, [video_eax] ; 12/05/2016
                            <1> _video_return:
00001559 FA                 <1>       cli ; 02/01/2017
0000155A 5D                 <1>       pop   ebp
0000155B 5F                 <1>       pop   edi
0000155C 5E                 <1>       pop   esi
0000155D 5B                 <1>       pop   ebx
                            <1> M15:  ; VIDEO_RETURN_C
                            <1>       ;;15/01/2017
                            <1>       ; 02/01/2017
                            <1>       ;;mov  byte [intflg], 0
                            <1>       ;
0000155E 59                 <1>       pop   ecx
0000155F 5A                 <1>       pop   edx
00001560 1F                 <1>       pop   ds
00001561 07                 <1>       pop   es    ; RECOVER SEGMENTS
00001562 CF                 <1>       iretd       ; ALL DONE
                            <1>
                            <1> set_txt_mode:
                            <1>       ; 29/07/2016
                            <1>       ; 27/06/2016
00001563 B003               <1>       mov   al, 3
                            <1>
                            <1> ; 10/08/2016
                            <1> ; 08/08/2016
                            <1> ; 30/07/2016
                            <1> ; 29/07/2016
                            <1> ; 27/07/2016
                            <1> ; 26/07/2016
                            <1> ; 25/07/2016
                            <1> ; 23/07/2016
                            <1> ; 18/07/2016
                            <1> ; 07/07/2016
                            <1> ; 04/07/2016
                            <1> ; 03/07/2016
                            <1> ; 02/07/2016
                            <1> ; 26/06/2016
                            <1> ; 24/06/2016 (set_txt_mode -> _set_mode)
                            <1> ; 17/06/2016
                            <1> ; 29/05/2016
                            <1> ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
                            <1> _set_mode:
                            <1>       ; 24/06/2016
00001565 3805[C25E0000]     <1>       cmp   [CRT_MODE], al  ; current mode = requested mode ?
0000156B 750D               <1>       jne   short _sm_0
0000156D 3C03               <1>       cmp   al, 3       ; text, 80*25 color, default mode
                            <1>                         ; for TRDOS 386 MainProg
0000156F 755F               <1>         jne     short _sm_2     ;  multiscreen is only for mode 3
                            <1>
                            <1>       ; If '_set_mode' procedure is called for video mode 3
                            <1>       ;     while video mode is 3, video page will be cleared
                            <1>       ;     and cursor position of video page will be reset.
                            <1>
                            <1>       ; 29/07/2016
00001571 800D[B95F0100]80   <1>       or    byte [p_crt_mode], 80h ; clear page indicator
00001578 EB5B               <1>         jmp     short _sm_3
                            <1> _sm_0:
0000157A 803D[C25E0000]03   <1>       cmp   byte [CRT_MODE], 3
00001581 7534               <1>         jne     short _sm_1
                            <1>
                            <1>       ; If '_set_mode' procedure is called for a video mode
                            <1>       ;     except video mode 3, while current video mode
                            <1>       ;     is 3; all video pages of mode 3 will be copied
                            <1>       ;     to 98000h address as backup, before mode change.
                            <1>
                            <1> _sm_save_pm:
                            <1>       ; 03/07/2016
                            <1>       ; save video pages
00001583 BE00800B00         <1>       mov   esi, 0B8000h
00001588 BF00800900         <1>       mov   edi, 98000h ; 30/07/2016
0000158D B900200000         <1>       mov   ecx, (0B8000h-0B0000h)/4
00001592 F3A5               <1>       rep   movsd
                            <1>
00001594 C605[B95F0100]03   <1>       mov   byte [p_crt_mode], 3 ; previous mode, backup sign
                            <1>       ;mov  cl, [ACTIVE_PAGE]
                            <1>       ;mov  [p_crt_page], cl
                            <1>
                            <1>       ; save cursor positions
0000159B BE[3E520100]       <1>       mov   esi, CURSOR_POSN
000015A0 BF[BE5F0100]       <1>       mov   edi, cursor_pposn    ; cursor positions backup
000015A5 B104               <1>       mov   cl, 4
000015A7 F3A5               <1>       rep   movsd
                            <1>
                            <1>       ; 29/07/2016
```

```
4222                                    <1>       ;mov   [ACTIVE_PAGE], cl ; 0
4223 000015A9 860D[4E520100]            <1>       xchg   cl, [ACTIVE_PAGE]
4224 000015AF 880D[BA5F0100]            <1>       mov    [p_crt_page], cl     ; previous page (for mode 3)
4225                                    <1>       ; [ACTIVE_PAGE] = 0
4226 000015B5 EB19                      <1>       jmp    short _sm_2
4227                                    <1>
4228                                    <1> _sm_1:
4229 000015B7 3C03                      <1>       cmp    al, 3        ; text, 80*25 color, default mode
4230                                    <1>                           ; for TRDOS 386 MainProg
4231 000015B9 7515                      <1>       jne    short _sm_2  ;  multiscreen is only for mode 3
4232                                    <1>
4233                                    <1>       ; If '_set_mode' procedure is called for video mode 3
4234                                    <1>       ;     while video mode is not 3 and if there is video
4235                                    <1>       ;     page backup for video mode 3, all (of 8) mode 3
4236                                    <1>       ;     video pages will be restored from 98000h.
4237                                    <1>
4238 000015BB 803D[B95F0100]03          <1>       cmp    byte [p_crt_mode], 3 ; previous mode, backup sign
4239 000015C2 750C                      <1>       jne    short _sm_2 ; there is no (multiscreen) video pages
4240                                    <1>                          ; to be restored
4241 000015C4 8A0D[BA5F0100]            <1>       mov    cl, [p_crt_page]
4242 000015CA 880D[4E520100]            <1>       mov    [ACTIVE_PAGE], cl
4243                                    <1>
4244                                    <1> _sm_2:
4245 000015D0 A2[C25E0000]              <1>       mov    [CRT_MODE], al  ; save mode in global variable
4246                                    <1> _sm_3:
4247                                    <1>       ; 30/07/2016
4248                                    <1>       ; 26/07/2016
4249                                    <1>       ; 25/07/2016
4250                                    <1>       ; set_mode_vga:
4251                                    <1>       ; 18/07/2016
4252                                    <1>       ; 14/07/2016
4253                                    <1>       ; 09/07/2016
4254                                    <1>       ; 04/07/2016
4255                                    <1>       ; 03/07/2016 (TRDOS 386 = TRDOS v2.0)
4256                                    <1>       ; /// video mode 13h ///
4257                                    <1>       ; derived from 'Plex86/Bochs VGABios' source code
4258                                    <1>       ; vgabios-0.7a (2011)
4259                                    <1>       ; by the LGPL VGABios developers Team (2001-2008)
4260                                    <1>       ; 'vgabios.c', 'vgatables.h'
4261                                    <1>       ;
4262                                    <1>       ; Oracle VirtualBox 5.0.24 VGABios Source Code
4263                                    <1>       ; ('vgabios.c', 'vgatables.h', 'vgafonts.h', 'vgarom.asm')
4264                                    <1>       ;
4265 000015D5 88C4                      <1>       mov    ah, al
4266 000015D7 B910000000                <1>       mov    ecx, vga_mode_count
4267 000015DC BE[DE5E0000]              <1>       mov    esi, vga_modes
4268 000015E1 31DB                      <1>       xor    ebx, ebx
4269                                    <1> _sm_4:
4270 000015E3 AC                        <1>       lodsb
4271 000015E4 38C4                      <1>       cmp    ah, al
4272 000015E6 740C                      <1>       je     short _sm_5
4273 000015E8 FEC3                      <1>       inc    bl
4274 000015EA E2F7                      <1>       loop   _sm_4
4275                                    <1>
4276                                    <1>       ; UNIMPLEMENTED VIDEO MODE !
4277 000015EC 31C0                      <1>       xor    eax, eax
4278 000015EE A3[AC5F0100]              <1>         mov     [video_eax], eax ; 0
4279 000015F3 C3                        <1>       retn
4280                                    <1>
4281                                    <1> ;-----      eBX POINTS TO CORRECT ROW OF INITIALIZATION TABLE
4282                                    <1>
4283                                    <1> _sm_5:      ; 25/07/2016
4284 000015F4 89DE                      <1>       mov    esi, ebx
4285 000015F6 81C6[2E5F0000]            <1>       add    esi, vga_memmodel
4286 000015FC 8A06                      <1>       mov    al, [esi]
4287 000015FE A2[D25F0100]              <1>       mov    [VGA_MTYPE], al
4288                                    <1>
4289 00001603 89DF                      <1>       mov    edi, ebx
4290 00001605 81C7[3E5F0000]            <1>       add    edi, vga_dac_s
4291 0000160B C0E302                    <1>       shl    bl, 2 ; byte -> dword
4292 0000160E 81C3[EE5E0000]            <1>       add    ebx, vga_mode_tbl_ptr
4293                                    <1>
4294                                    <1>       ;mov   dword [VGA_BASE], 0B8000h
4295                                    <1>       ;cmp   ah, 0Dh ; [CRT_MODE]
4296                                    <1>       ;jb    short M9
4297                                    <1>       ;mov   dword [VGA_BASE], 0A0000h
4298                                    <1> ;M9:
4299 00001614 8B33                      <1>       mov    esi, [ebx]
4300 00001616 89F3                      <1>       mov    ebx, esi
4301 00001618 83C614                    <1>       add    esi, vga_p_cm_pos ; ebx + 20
4302 0000161B 668B06                    <1>       mov    ax, [esi]       ; get the cursor mode from the table
4303 0000161E 66A3[DB5E0000]            <1>       mov    [CURSOR_MODE], ax ; save cursor mode (initial value)
4304                                    <1>       ; al = 6, ah = 7
4305                                    <1>       ; al = 0Dh, ah = 0Eh ; 25/07/2016
4306 00001624 E83B020000                <1>       call   cursor_shape_fix
4307                                    <1>       ; al = 14, ah = 15  (If [CHAR_HEIGHT] = 16)
4308 00001629 668906                    <1>       mov    [esi], ax
4309                                    <1>
4310 0000162C 56                        <1>       push   esi ; *
4311                                    <1>
4312 0000162D 8A25[C95E0000]            <1>       mov    ah, [VGA_MODESET_CTL]
4313 00001633 80E408                    <1>       and    ah, 8 ; default palette loading ?
4314 00001636 7524                      <1>       jnz    short _sm_6
4315 00001638 66BAC603                  <1>       mov    dx, 3C6h ; VGAREG_PEL_MASK (DAC mask register)
4316 0000163C B0FF                      <1>       mov    al, 0FFh ; PEL mask
4317 0000163E EE                        <1>       out    dx, al
4318 0000163F 8A27                      <1>       mov    ah, [edi] ; DAC model (selection number)
4319 00001641 E8ED0F0000                <1>       call   load_dac_palette
4320                                    <1>       ; ecx = 0
4321 00001646 F605[C95E0000]02          <1>       test   byte [VGA_MODESET_CTL], 2 ; gray scale summing
4322 0000164D 740D                      <1>       jz     short _sm_6
4323 0000164F 53                        <1>       push   ebx
4324 00001650 29DB                      <1>       sub    ebx, ebx ; sub bl, bl
```

```
4325 00001652 66B90001            <1>         mov     cx, 256
4326 00001656 E82B100000          <1>       call   gray_scale_summing
4327 0000165B 5B                  <1>       pop    ebx
4328                              <1> _sm_6:
4329                              <1>       ; Reset Attribute Ctl flip-flop
4330 0000165C 66BADA03            <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
4331 00001660 EC                  <1>       in     al, dx
4332                              <1>       ; Set Attribute Ctl
4333 00001661 89DE                <1>       mov    esi, ebx ; addr of params tbl for selected mode
4334 00001663 83C623              <1>       add    esi, 35  ; actl regs
4335 00001666 30E4                <1>       xor    ah, ah ; 0
4336 00001668 66BAC003            <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
4337                              <1> _sm_7:
4338 0000166C 88E0                <1>       mov    al, ah
4339 0000166E EE                  <1>       out    dx, al ; index
4340 0000166F AC                  <1>       lodsb
4341                              <1>       ; DX = 3C0h = VGAREG_ACTL_WRITE_DATA
4342 00001670 EE                  <1>       out    dx, al ; value
4343 00001671 FEC4                <1>       inc    ah
4344 00001673 80FC14              <1>       cmp    ah, 20 ; number of actl registers
4345 00001676 72F4                <1>       jb     short _sm_7
4346                              <1>       ;
4347 00001678 88E0                <1>       mov    al, ah ; 20
4348 0000167A EE                  <1>       out    dx, al ; index
4349 0000167B 28C0                <1>       sub    al, al ; 0
4350 0000167D EE                  <1>       out    dx, al ; value
4351                              <1>       ;
4352                              <1>       ; Set Sequencer Ctl
4353 0000167E 89DE                <1>       mov    esi, ebx ; addr of params tbl for selected mode
4354 00001680 83C605              <1>       add    esi, 5 ; sequ regs
4355                              <1>       ;
4356 00001683 66BAC403            <1>       mov    dx, 3C4h  ; VGAREG_SEQU_ADDRESS
4357 00001687 EE                  <1>       out    dx, al ; 0
4358 00001688 6642                <1>       inc    dx ; 3C5h ; VGAREG_SEQU_DATA
4359 0000168A B003                <1>       mov    al, 3
4360 0000168C EE                  <1>       out    dx, al
4361 0000168D B401                <1>       mov    ah, 1
4362                              <1> _sm_8:
4363 0000168F 88E0                <1>       mov    al, ah
4364                              <1>       ;mov   dx, 3C4h ; VGAREG_SEQU_ADDRESS
4365 00001691 664A                <1>       dec    dx
4366 00001693 EE                  <1>       out    dx, al ; index
4367 00001694 AC                  <1>       lodsb
4368 00001695 6642                <1>       inc    dx ; 3C5h ; VGAREG_SEQU_DATA
4369 00001697 EE                  <1>       out    dx, al
4370 00001698 80FC04              <1>       cmp    ah, 4 ; number of sequ regs
4371 0000169B 7304                <1>       jnb    short _sm_9
4372 0000169D FEC4                <1>       inc    ah
4373 0000169F EBEE                <1>       jmp    short _sm_8
4374                              <1> _sm_9:
4375                              <1>       ; Set Grafx Ctl
4376 000016A1 89DE                <1>       mov    esi, ebx ; addr of params tbl for selected mode
4377 000016A3 83C637              <1>       add    esi, 55 ; grdc regs
4378 000016A6 30E4                <1>       xor    ah, ah ; 0
4379                              <1> _sm_10:
4380 000016A8 88E0                <1>       mov    al, ah
4381 000016AA 66BACE03            <1>       mov    dx, 3CEh ; VGAREG_GRDC_ADDRESS
4382 000016AE EE                  <1>       out    dx, al
4383 000016AF AC                  <1>       lodsb
4384 000016B0 6642                <1>       inc    dx ; 3CFh ; VGAREG_GRDC_DATA
4385 000016B2 EE                  <1>       out    dx, al
4386 000016B3 FEC4                <1>       inc    ah
4387 000016B5 80FC09              <1>       cmp    ah, 9 ; number of grdc regs
4388 000016B8 72EE                <1>       jb     short _sm_10
4389                              <1>       ;
4390                              <1>       ; Disable CRTC write protection
4391 000016BA 66BAD403            <1>       mov    dx, 3D4h  ; VGAREG_VGA_CRTC_ADDRESS
4392                              <1>       ;mov   al, 11h
4393                              <1>       ;our   dx, al
4394                              <1>       ;inc   dx
4395                              <1>       ;sub   al, al
4396                              <1>       ;out   dx, al
4397 000016BE 66B81100            <1>       mov    ax, 11h
4398 000016C2 66EF                <1>       out    dx, ax
4399 000016C4 89DE                <1>       mov    esi, ebx ; addr of params tbl for selected mode
4400 000016C6 83C60A              <1>       add    esi, 10 ; crtc regs
4401                              <1>       ; ah = 0
4402                              <1> _sm_11:
4403 000016C9 88E0                <1>       mov    al, ah
4404                              <1>       ; dx = 3D4h = VGAREG_VGA_CRTC_ADDRESS
4405 000016CB EE                  <1>       out    dx, al ; index
4406 000016CC AC                  <1>       lodsb
4407 000016CD 6642                <1>       inc    dx  ; VGAREG_VGA_CRTC_ADDRESS + 1
4408 000016CF EE                  <1>       out    dx, al ; value
4409 000016D0 80FC18              <1>       cmp    ah, 24 ; number of crtc registers - 1
4410 000016D3 7306                <1>       jnb    short _sm_12
4411 000016D5 FEC4                <1>       inc    ah
4412 000016D7 664A                <1>       dec    dx ; 3D4h
4413 000016D9 EBEE                <1>       jmp    short _sm_11
4414                              <1> _sm_12:
4415                              <1>       ; Set the misc register
4416 000016DB 66BACC03            <1>       mov    dx, 3CCh ; VGAREG_READ_MISC_OUTPUT
4417 000016DF 8A4309              <1>       mov    al, [ebx+9] ; misc reg
4418 000016E2 EE                  <1>       out    dx, al
4419                              <1>       ;
4420                              <1>       ; Enable video
4421 000016E3 66BAC003            <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
4422 000016E7 B020                <1>       mov    al, 20h
4423 000016E9 EE                  <1>         out     dx, al   ; set bit 5 to 1
4424 000016EA 66BADA03            <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
4425 000016EE EC                  <1>       in     al, dx
4426                              <1>       ;
4427 000016EF 803D[BB5F0100]00    <1>       cmp    byte [noclearmem], 0
```

```
4428 000016F6 7740                <1>         ja      short _sm_15
4429                              <1>
4430                              <1>       ; 29/07/2016
4431 000016F8 31C0                <1>       xor    eax, eax
4432 000016FA B900400000          <1>       mov    ecx, 4000h ; 16K words (32K)
4433 000016FF 803D[D25F0100]02    <1>       cmp     byte [VGA_MTYPE], 2  ; CTEXT, MTEXT, CGA
4434 00001706 7715                <1>       ja     short _sm_14    ; no ? (0A0000h)
4435 00001708 BF00800B00          <1>       mov    edi, 0B8000h
4436 0000170D 7409                <1>       je     short _sm_13 ; CGA graphics mode
4437                              <1>       ; 08/08/2016
4438 0000170F A3[CE5F0100]        <1>       mov    [VGA_INT43H], eax ; 0 ; default font
4439 00001714 66B82007            <1>       mov    ax, 0720h    ; CGA text mode
4440                              <1> _sm_13:
4441 00001718 F366AB              <1>       rep    stosw
4442 0000171B EB1B                <1>       jmp    short _sm_15
4443                              <1>
4444                              <1> _sm_14:
4445 0000171D BF00000A00          <1>       mov    edi, 0A0000h
4446                              <1>       ; ecx = 16384 dwords  (64K)
4447 00001722 66BAC403            <1>       mov    dx, 3C4h ; VGAREG_SEQU_ADDRESS
4448 00001726 B002                <1>       mov    al, 2
4449 00001728 EE                  <1>       out    dx, al
4450                              <1>       ;mov   dx, 3C5h ; VGAREG_SEQU_DATA
4451 00001729 6642                <1>       inc    dx
4452 0000172B EC                  <1>       in     al, dx ; mmask
4453 0000172C 6650                <1>       push   ax
4454 0000172E B00F                <1>       mov    al, 0Fh ; all planes
4455 00001730 EE                  <1>       out    dx, al
4456 00001731 30C0                <1>       xor    al, al ; 0
4457 00001733 F3AB                <1>       rep    stosd  ; ecx = 163684 (64K)
4458 00001735 6658                <1>       pop    ax
4459 00001737 EE                  <1>       out    dx, al  ; mmask
4460                              <1> _sm_15:
4461                              <1>       ; ebx = addr of params tbl for selected mode
4462                              <1>       ; 10/08/2016
4463 00001738 668B03              <1>       mov    ax, [ebx] ; num of columns, 'twidth'
4464 0000173B A2[C45E0000]        <1>       mov    [CRT_COLS], al
4465                              <1>       ;; 26/07/2016
4466                              <1>       ;; CRTC_ADDRESS = 3D4h (always)
4467                              <1>       ;mov   ah, [ebx+1] ; num of rows, 'theightm1'
4468 00001740 FEC4                <1>       inc    ah ; 09/07/2016
4469 00001742 8825[CA5E0000]      <1>       mov    [VGA_ROWS], ah
4470                              <1>       ; 10/08/2016
4471 00001748 8A4302              <1>       mov    al, [ebx+2]
4472 0000174B A2[C65E0000]        <1>       mov    [CHAR_HEIGHT], al
4473                              <1>       ; 29/07/2016
4474                              <1>       ; length of regen buffer in bytes
4475 00001750 668B4B03            <1>       mov    cx, [ebx+3] ; 'slength_l'
4476 00001754 66890D[BC5F0100]    <1>       mov    [CRT_LEN], cx
4477                              <1>       ;
4478                              <1>       ; 27/07/2016
4479 0000175B 30E4                <1>       xor    ah, ah
4480 0000175D A0[4E520100]        <1>       mov    al, [ACTIVE_PAGE] ; may be > 0 for mode 3
4481                              <1>       ;mul   word [CRT_LEN] ; 4096 for mode 3
4482 00001762 66F7E1              <1>       mul    cx ; 29/07/2016
4483 00001765 66A3[3C520100]      <1>       mov    [CRT_START], ax
4484                              <1>       ;
4485 0000176B B060                <1>       mov    al, 60h
4486 0000176D 803D[BB5F0100]00    <1>       cmp    byte [noclearmem], 0
4487 00001774 7602                <1>       jna    short _sm_16
4488 00001776 0480                <1>       add    al, 80h
4489                              <1> _sm_16:
4490 00001778 A2[C75E0000]        <1>       mov    [VGA_VIDEO_CTL], al
4491 0000177D C605[C85E0000]F9    <1>       mov    byte [VGA_SWITCHES], 0F9h
4492 00001784 8025[C95E0000]7F    <1>       and    byte [VGA_MODESET_CTL], 7Fh
4493                              <1>
4494 0000178B 5E                  <1>       pop    esi ; *
4495                              <1>
4496                              <1>       ; 26/07/2016
4497                              <1>       ; 07/07/2016
4498 0000178C 668B0D[DB5E0000]    <1>       mov    cx, [CURSOR_MODE] ; restore cursor mode (initial value)
4499 00001793 66870E              <1>       xchg   cx, [esi] ; cl = start line, ch = end line
4500                              <1>                        ; reset to initial value
4501 00001796 86E9                <1>       xchg   ch, cl  ; ch = start line, cl = end line
4502 00001798 66890D[DB5E0000]    <1>       mov    [CURSOR_MODE], cx ; save (fixed) cursor mode
4503                              <1>
4504                              <1>       ; 27/07/2016
4505 0000179F 803D[D25F0100]02    <1>       cmp    byte [VGA_MTYPE], 2 ; CTEXT, MTEXT
4506 000017A6 7317                <1>       jnb    short _sm_17
4507                              <1>
4508                              <1>       ; Set cursor shape
4509                              <1>       ;mov   cx, 0607h
4510                              <1>       ;call  _set_ctype
4511                              <1>
4512                              <1>       ; 29/07/2016
4513 000017A8 B40A                <1>       mov    ah, 10 ; 6845 register for cursor set
4514 000017AA E8C4050000          <1>       call   m16    ; output cx register
4515                              <1>
4516                              <1>       ; 25/07/2016
4517 000017AF 803D[C25E0000]03    <1>         cmp     byte [CRT_MODE], 03h
4518 000017B6 7507                <1>       jne    short _sm_17
4519                              <1>       ; 26/07/2016
4520                              <1>
4521 000017B8 A0[4E520100]        <1>       mov    al, [ACTIVE_PAGE]
4522 000017BD EB0C                <1>       jmp    short _sm_18
4523                              <1> _sm_17:
4524                              <1>       ; Set cursor pos for page 0..7
4525 000017BF 6629C0              <1>       sub    ax, ax ; eax = 0
4526 000017C2 BF[3E520100]        <1>       mov    edi, CURSOR_POSN
4527 000017C7 AB                  <1>       stosd
4528 000017C8 AB                  <1>       stosd
4529 000017C9 AB                  <1>       stosd
4530 000017CA AB                  <1>       stosd
```

```
4531                                <1>       ;; Set active page 0
4532                                <1>       ;mov   [ACTIVE_PAGE], al ; 0
4533                                <1> _sm_18:
4534                                <1>       ; 29/07/2016
4535 000017CB 803D[D25F0100]02     <1>       cmp    byte [VGA_MTYPE], 2 ; CTEXT, MTEXT
4536 000017D2 0F8386000000         <1>         jnb  _sm_23
4537                                <1>
4538                                <1>       ;cmp   byte [CHAR_HEIGHT], 16
4539                                <1>       ;je    short _sm_19
4540                                <1>
4541                                <1>       ;; copy and activate 8x16 font
4542                                <1>
4543                                <1>       ; 26/07/2016
4544 000017D8 B004                 <1>       mov    al, 04h
4545                                <1>       ;sub   bl, bl
4546                                <1>       ; AX = 1104H ; Load ROM 8x16 Character Set
4547                                <1>       ; (BL = font block to load (EGA: 0-3; VGA: 0-7))
4548 000017DA E83A150000           <1>       call   load_text_8_16_pat
4549                                <1>
4550                                <1>       ; video_func_1103h:
4551                                <1>       ; biosfn_set_text_block_specifier:
4552                                <1>       ; BL = font block selector code
4553                                <1>       ; NOTE: TRDOS 386 only uses and sets font block 0
4554                                <1>       ; (It is as BL = 0 for TRDOS 386)
4555 000017DF 66BAC403             <1>       mov    dx, 3C4h ; VGAREG_SEQU_ADDRESS
4556                                <1>       ;mov   ah, bl
4557 000017E3 28E4                 <1>       sub    ah, ah ; 0
4558 000017E5 B003                 <1>       mov    al, 03h
4559 000017E7 66EF                 <1>       out    dx, ax
4560                                <1> _sm_19:
4561                                <1>       ; 29/07/2016
4562                                <1>       ; 26/07/2016
4563                                <1>       ; 24/06/2016
4564                                <1>       ;mov   edi, 0B8000h
4565                                <1>       ;mov   cx, 4000h ; 16K words (32K)
4566                                <1>       ;
4567 000017E9 30C0                 <1>       xor    al, al
4568 000017EB 3805[B95F0100]       <1>         cmp     byte [p_crt_mode], al ; 0
4569 000017F1 7707                 <1>         ja      short _sm_20 ; 3h, 80h or 83h
4570                                <1>
4571                                <1>       ; 30/07/2016
4572                                <1>       ; 24/06/2016
4573                                <1>       ; TRDOS 386 (TRDOS v2) 'set mode' modification
4574                                <1>       ; (for multiscreen feature):
4575                                <1>       ; If '_set_mode' procedure is called for video mode 3
4576                                <1>       ;     while video mode is 3, video page will be cleared
4577                                <1>       ;     and cursor position of video page will be reset.
4578                                <1>       ; If '_set_mode' procedure is called for a video mode
4579                                <1>       ;     except video mode 3, while current video mode
4580                                <1>       ;     is 3; all video pages of mode 3 will be copied
4581                                <1>       ;     to 98000h address as backup, before mode change.
4582                                <1>       ; If '_set_mode' procedure is called for video mode 3
4583                                <1>       ;     while video mode is not 3 and if there is video
4584                                <1>       ;     page backup for video mode 3, all (of 8) mode 3
4585                                <1>       ;     video pages will be restored from 98000h.
4586                                <1>
4587 000017F3 A2[4E520100]         <1>       mov    [ACTIVE_PAGE], al ; 0
4588                                <1>       ;mov   ax, 0720h
4589                                <1>       ;;mov  cx, 4000h ; 16K words (32K)
4590                                <1>       ;;mov  edi, 0B8000h
4591                                <1>       ;rep   stosw
4592                                <1>       ;sub   al, al
4593 000017F8 EB64                 <1>       jmp    short _sm_23
4594                                <1> _sm_20:
4595                                <1>       ; Previous video mode is 3
4596                                <1>       ; New video mode is 3 while current video mode is not 3
4597                                <1>       ; (multi screen) video pages will be restored from 0B0000h
4598                                <1>
4599 000017FA 0FB61D[4E520100]     <1>       movzx  ebx, byte [ACTIVE_PAGE]
4600 00001801 D0E3                 <1>       shl    bl, 1 ; * 2
4601 00001803 81C3[3E520100]       <1>       add    ebx, CURSOR_POSN
4602                                <1>
4603                                <1>       ; 29/07/2016
4604 00001809 F605[B95F0100]7F     <1>       test    byte [p_crt_mode], 7Fh ; 83h or 3h
4605 00001810 7427                 <1>       jz     short _sm_21 ; do not restore video pages
4606                                <1>
4607                                <1>       ;; restore video pages
4608 00001812 BE00800900           <1>       mov    esi, 98000h ; 30/07/2016
4609 00001817 BF00800B00           <1>       mov    edi, 0B8000h
4610 0000181C 66B90020             <1>       mov    cx, 2000h ; 8K dwords (32K)
4611 00001820 F3A5                 <1>       rep    movsd
4612                                <1>
4613                                <1>       ; restore cursor positions
4614 00001822 BE[BE5F0100]         <1>       mov    esi, cursor_pposn
4615 00001827 BF[3E520100]         <1>       mov    edi, CURSOR_POSN
4616                                <1>       ;mov   ecx, 4 ; restore all cursor positions (16 bytes)
4617 0000182C B104                 <1>       mov    cl, 4
4618 0000182E F3A5                 <1>       rep    movsd
4619                                <1>
4620 00001830 F605[B95F0100]80     <1>         test    byte [p_crt_mode], 80h
4621 00001837 7420                 <1>       jz     short _sm_22 ; do not clear current video pages
4622                                <1> _sm_21:
4623                                <1>       ; clear video page
4624 00001839 668B0D[BC5F0100]     <1>       mov    cx, [CRT_LEN] ; 4096
4625 00001840 66D1E9               <1>       shr    cx, 1 ; 2072
4626 00001843 66B82007             <1>       mov    ax, 0720h
4627 00001847 BF00800B00           <1>       mov    edi, 0B8000h ; [crt_base]
4628 0000184C 66033D[3C520100]     <1>       add    di, [CRT_START]
4629 00001853 F366AB               <1>       rep    stosw  ; FILL THE REGEN BUFFER WITH BLANKS
4630                                <1>       ;
4631 00001856 66890B               <1>       mov    [ebx], cx ; reset cursor position
4632                                <1> _sm_22:
4633 00001859 A2[B95F0100]         <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)
                                <1> csf_2:
0000189B 880424                 <1>       mov    [esp], al
0000189E 8A642401               <1>       mov    ah, [esp+1]
                                <1>       ; CL = ((CL+1) * cheight / 8) - 1;
000018A2 FEC4                   <1>       inc    ah
000018A4 A0[C65E0000]           <1>       mov    al, [CHAR_HEIGHT]
000018A9 F6E4                   <1>       mul    ah
000018AB C0E803                 <1>       shr    al, 3 ; / 8
000018AE FEC8                   <1>       dec    al ; - 1
000018B0 88442401               <1>       mov    [esp+1], al
                                <1>       ; ah = 15 (if [CHAR_HEIGHT] = 16)
                                <1>       ;
000018B4 6658                   <1>       pop    ax
                                <1> csf_3:
000018B6 C3                     <1>       retn
                                <1>
                                <1> SET_CTYPE:
                                <1>       ; 12/09/2016
                                <1>       ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
000018B7 803D[C25E0000]07       <1>       cmp    byte [CRT_MODE], 7
000018BE 0F8790FCFFFF           <1>       ja     VIDEO_RETURN ; 12/09/2016
000018C4 E805000000             <1>       call   _set_ctype
000018C9 E986FCFFFF             <1>         jmp     VIDEO_RETURN
                                <1>
                                <1> _set_ctype:
                                <1>       ; 02/09/2014 (Retro UNIX 386 v1)
                                <1>       ;
                                <1>       ; VIDEO.ASM - 06/10/85  VIDEO DISPLAY BIOS
                                <1>
                                <1>       ; (CH) = BITS 4-0 = START LINE FOR CURSOR
                                <1>       ;  ** HARDWARE WILL ALWAYS CAUSE BLINK
                                <1>       ;  ** SETTING BIT 5 OR 6 WILL CAUSE ERRATIC BLINKING
                                <1>       ;     OR NO CURSOR AT ALL
                                <1>       ; (CL) = BITS 4-0 = END LINE FOR CURSOR
                                <1>
                                <1> ;--------------------------------------------------
                                <1> ; SET_CTYPE
                                <1> ;     THIS ROUTINE SETS THE CURSOR VALUE
                                <1> ; INPUT
                                <1> ;     (CX) HAS CURSOR VALUE CH-START LINE, CL-STOP LINE
                                <1> ; OUTPUT
                                <1> ;     NONE
                                <1> ;--------------------------------------------------
                                <1>
```

```
                                <1>       ; 07/07/2016
                                <1>       ; Fixing cursor start and stop line depending on
                                <1>       ; current character height (=16)
                                <1>       ; (Note: Default/initial values are 6 and 7.
                                <1>       ; If set values are 6 (start) & 7 (stop) and
                                <1>       ; [CHAR_HEIGHT] = 16 :
                                <1>       ; After fixing, start line will be 14, stop line
                                <1>       ; will be 15.)
000018CE 6689C8                 <1>       mov   ax, cx
000018D1 86C4                   <1>       xchg  al, ah
                                <1>       ; AL = start line, AH = stop line
000018D3 E88CFFFFFF             <1>       call  cursor_shape_fix
                                <1>       ; AL = start line (fixed), AH = stop line (fixed)
000018D8 6689C1                 <1>       mov   cx, ax
000018DB 86E9                   <1>       xchg  ch, cl
                                <1>       ; CH = start line (fixed), CL = stop line (fixed)
                                <1>       ;
000018DD B40A                   <1>       mov   ah, 10 ; 6845 register for cursor set
000018DF 66890D[DB5E0000]       <1>       mov   [CURSOR_MODE], cx ; save in data area
                                <1>       ;call  m16    ; output cx register
                                <1>       ;retn
000018E6 E988040000             <1>         jmp     m16
                                <1>
                                <1> SET_CPOS:
                                <1>       ; 12/09/2016
                                <1>       ; 07/07/2016
                                <1>       ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
000018EB 80FF07                 <1>       cmp   bh, 7 ; video page > 7 ; 07/07/2016
000018EE 0F8760FCFFFF           <1>       ja    VIDEO_RETURN
                                <1>       ;
000018F4 803D[C25E0000]07       <1>       cmp   byte [CRT_MODE], 7
000018FB 770A                   <1>       ja    short vga_set_cpos ; 12/09/2016
000018FD E846040000             <1>       call  _set_cpos
00001902 E94DFCFFFF             <1>         jmp     VIDEO_RETURN
                                <1>
                                <1> vga_set_cpos:
                                <1>       ; 12/09/2016
                                <1>       ; 09/07/2016
                                <1>       ; set cursor position
                                <1>       ; NOTE: Hardware cursor position will not be set
                                <1>       ;   in any VGA modes (>7)
                                <1>       ;   But, cursor position will be saved into
                                <1>       ;   [CURSOR_POSN].
                                <1>       ;   TRDOS 386 (TRDOS v2.0) uses only one page
                                <1>       ;   (page 0) for all graphics modes.
                                <1>
00001907 668915[3E520100]       <1>       mov   [CURSOR_POSN], dx ; save cursor pos for pg 0
                                <1>       ; 04/08/2016
                                <1>       ;mov   bh, [ACTIVE_PAGE] ; = 0
                                <1>       ;call  _set_cpos
0000190E E941FCFFFF             <1>       jmp     VIDEO_RETURN
                                <1>
                                <1> READ_CURSOR:
                                <1>       ; 12/09/2016
                                <1>       ; 07/07/2016
                                <1>       ; 12/05/2016
                                <1>       ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>       ;
                                <1>       ; VIDEO.ASM - 06/10/85  VIDEO DISPLAY BIOS
                                <1>
                                <1> ;--------------------------------------------------
                                <1> ; READ_CURSOR
                                <1> ;     THIS ROUTINE READS THE CURRENT CURSOR VALUE FROM THE
                                <1> ;     845, FORMATS IT, AND SENDS IT BACK TO THE CALLER
                                <1> ; INPUT
                                <1> ;     BH - PAGE OF CURSOR
                                <1> ; OUTPUT
                                <1> ;     DX - ROW, COLUMN OF THE CURRENT CURSOR POSITION
                                <1> ;     CX - CURRENT CURSOR MODE
                                <1> ;--------------------------------------------------
                                <1>
                                <1>       ; BH = Video page number (0 to 7)
                                <1>
                                <1>       ; 07/07/2016
00001913 80FF07                 <1>       cmp   bh, 7 ; video page > 7 (invalid)
00001916 7606                   <1>       jna   short read_cursor_1
                                <1>       ; invalid video page (input)
00001918 31C9                   <1>       xor   ecx, ecx ; 0
0000191A 31D2                   <1>       xor   edx, edx ; 0
0000191C EB15                   <1>       jmp   short read_cursor_2
                                <1> read_cursor_1:
                                <1>       ; 12/09/2016
0000191E 803D[C25E0000]07       <1>       cmp   byte [CRT_MODE], 7 ; vga mode
00001925 7727                   <1>       ja    short vga_get_cpos
                                <1>       ;
00001927 E815000000             <1>       call  get_cpos
0000192C 0FB70D[DB5E0000]       <1>       movzx  ecx, word [CURSOR_MODE]
                                <1> read_cursor_2:
00001933 5D                     <1>       pop   ebp
00001934 5F                     <1>       pop   edi
00001935 5E                     <1>       pop   esi
00001936 5B                     <1>       pop   ebx
00001937 58                     <1>       pop   eax  ; DISCARD SAVED CX AND DX
00001938 58                     <1>       pop   eax
00001939 A1[AC5F0100]           <1>       mov   eax, [video_eax] ; 12/05/2016
                                <1>       ;;15/01/2017
                                <1>       ;;mov  byte [intflg], 0 ; 07/01/2017
0000193E 1F                     <1>       pop   ds
0000193F 07                     <1>       pop   es
00001940 CF                     <1>       iretd
                                <1>
                                <1> get_cpos:
                                <1>       ; 12/05/2016
```

```
                                <1>       ; 16/01/2016
                                <1>       ; BH = Video page number (0 to 7)
                                <1>       ;
00001941 D0E7                   <1>       shl    bh, 1 ; WORD OFFSET
00001943 0FB6F7                 <1>       movzx  esi, bh
00001946 0FB796[3E520100]       <1>       movzx  edx, word [esi+CURSOR_POSN]
0000194D C3                     <1>       retn
                                <1>
                                <1> vga_get_cpos:
                                <1>       ; 12/09/2016
                                <1>       ; get cursor position (vga)
0000194E 0FB715[3E520100]       <1>       movzx  edx, word [CURSOR_POSN] ; cursor pos for pg 0
00001955 31C9                   <1>       xor    ecx, ecx ; Cursor Mode = 0 (invalid)
00001957 EBDA                   <1>       jmp     short read_cursor_2
                                <1>
                                <1> ACT_DISP_PAGE:
                                <1>       ; 07/07/2016
                                <1>       ; 26/06/2016
                                <1>       ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>       ;
                                <1>       ; VIDEO.ASM - 06/10/85  VIDEO DISPLAY BIOS
                                <1>       ;
                                <1> ;--------------------------------------------------
                                <1> ; ACT_DISP_PAGE
                                <1> ;     THIS ROUTINE SETS THE ACTIVE DISPLAY PAGE, ALLOWING
                                <1> ;     THE FULL USE OF THE MEMORY SET ASIDE FOR THE VIDEO ATTACHMENT
                                <1> ; INPUT
                                <1> ;     AL HAS THE NEW ACTIVE DISPLAY PAGE
                                <1> ; OUTPUT
                                <1> ;     THE 6845 IS RESET TO DISPLAY THAT PAGE
                                <1> ;--------------------------------------------------
                                <1>       ; 07/07/2016
00001959 3C07                   <1>       cmp    al, 7  ; > 7 = invalid video page number
0000195B 0F87F3FBFFFF           <1>       ja     VIDEO_RETURN
00001961 803D[C25E0000]03       <1>         cmp     byte [CRT_MODE], 3
00001968 7408                   <1>       je     short adp_1
0000196A 20C0                   <1>       and    al, al
0000196C 0F85E2FBFFFF           <1>         jnz     VIDEO_RETURN
                                <1>       ;sub   al, al ; 0 ; force to page 0
                                <1> adp_1:
00001972 E805000000             <1>       call   set_active_page
00001977 E9D8FBFFFF             <1>         jmp     VIDEO_RETURN
                                <1>
                                <1> set_active_page:   ; tty_sw
                                <1>       ; 09/12/2017
                                <1>       ; 26/07/2016
                                <1>       ; 26/06/2016
                                <1>       ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>       ; 30/06/2015
                                <1>       ; 04/03/2014  (act_disp_page --> tty_sw)
                                <1>       ; 10/12/2013
                                <1>       ; 04/12/2013
                                <1>       ;
0000197C A2[4E520100]           <1>       mov    [ACTIVE_PAGE], al ; save active page value ; [ptty]
                                <1> _set_active_page:
                                <1>       ; 27/06/2015
00001981 0FB6D8                 <1>       movzx  ebx, al
                                <1>       ;
                                <1>       ;cbw   ; 07/09/2014 (ah=0)
00001984 28E4                   <1>       sub    ah, ah ; 09/12/2017
00001986 66F725[BC5F0100]       <1>       mul    word [CRT_LEN]  ; get saved length of regen buffer
                                <1>                              ; display page times regen length
                                <1>       ; 10/12/2013
0000198D 66A3[3C520100]         <1>       mov    [CRT_START], ax ; save start address for later
00001993 6689C1                 <1>       mov    cx, ax ; start address to cx
                                <1> _M16:
                                <1>       ;sar   cx, 1
00001996 66D1E9                 <1>       shr    cx, 1  ; divide by 2 for 6845 handling
00001999 B40C                   <1>       mov    ah, 12 ; 6845 register for start address
0000199B E8D3030000             <1>       call   m16
                                <1>       ;sal   bx, 1
                                <1>       ; 01/09/2014
000019A0 D0E3                   <1>       shl    bl, 1  ; *2 for word offset
000019A2 81C3[3E520100]         <1>       add    ebx, CURSOR_POSN
000019A8 668B13                 <1>       mov    dx, [ebx] ; get cursor for this page
                                <1>       ; 16/01/2016
                                <1>       ;call  m18
                                <1>       ;retn
000019AB E9AF030000             <1>       jmp    m18
                                <1>
                                <1> position:
                                <1>       ; 24/06/2016
                                <1>       ; 12/05/2016 - TRDOS 386 (TRDOS v2.0)
                                <1>       ; 27/06/2015
                                <1>       ; 02/09/2014
                                <1>       ; 30/08/2014 (Retro UNIX 386 v1)
                                <1>       ; 04/12/2013 (Retro UNIX 8086 v1)
                                <1>       ;
                                <1>       ; VIDEO.ASM - 06/10/85  VIDEO DISPLAY BIOS
                                <1>       ;
                                <1> ;--------------------------------------
                                <1> ; POSITION
                                <1> ;     THIS SERVICE ROUTINE CALCULATES THE REGEN BUFFER ADDRESS
                                <1> ;     OF A CHARACTER IN THE ALPHA MODE
                                <1> ; INPUT
                                <1> ;     AX = ROW, COLUMN POSITION
                                <1> ; OUTPUT
                                <1> ;     AX = OFFSET OF CHAR POSITION IN REGEN BUFFER
                                <1> ;--------------------------------------
                                <1>
                                <1>       ; DX = ROW, COLUMN POSITION
000019B0 0FB605[C45E0000]       <1>       movzx  eax, byte [CRT_COLS] ; 24/06/2016
000019B7 F6E6                   <1>       mul    dh      ; row value
```

```
4943 000019B9 30F6                <1>       xor    dh, dh   ; 0
4944 000019BB 6601D0              <1>       add    ax, dx  ; add column value to the result
4945 000019BE 66D1E0              <1>       shl    ax, 1  ; * 2 for attribute bytes
4946                              <1>              ; EAX = AX = OFFSET OF CHAR POSITION IN REGEN BUFFER
4947 000019C1 C3                  <1>       retn
4948                              <1>
4949                              <1> find_position:
4950                              <1>       ; 24/06/2016
4951                              <1>       ; 12/05/2016 - TRDOS 386 (TRDOS v2.0)
4952                              <1>       ; 27/06/2015
4953                              <1>       ; 07/09/2014
4954                              <1>       ; 02/09/2014
4955                              <1>       ; 30/08/2014 (Retro UNIX 386 v1)
4956                              <1>       ; VIDEO.ASM - 06/10/85  VIDEO DISPLAY BIOS
4957                              <1>
4958 000019C2 0FB6CF              <1>       movzx  ecx, bh ; video page number
4959 000019C5 89CE                <1>       mov    esi, ecx
4960 000019C7 66D1E6              <1>       shl    si, 1
4961 000019CA 668B96[3E520100]    <1>       mov    dx, [esi+CURSOR_POSN]
4962 000019D1 740C                <1>       jz     short p21
4963 000019D3 6631F6              <1>       xor    si, si
4964                              <1> p20:
4965 000019D6 660335[BC5F0100]    <1>       add    si, [CRT_LEN] ; 24/06/2016
4966                              <1>       ;add   si, 80*25*2 ; add length of buffer for one page
4967 000019DD E2F7                <1>       loop   p20
4968                              <1> p21:
4969 000019DF 6621D2              <1>       and    dx, dx
4970 000019E2 7407                <1>       jz     short p22
4971 000019E4 E8C7FFFFFF          <1>       call   position ; determine location in regen in page
4972 000019E9 01C6                <1>       add    esi, eax ; add location to start of regen page
4973                              <1> p22:
4974                              <1>       ;mov   dx, [addr_6845] ; get base address of active display
4975                              <1>       ;mov   dx, 03D4h ; I/O address of color card
4976                              <1>       ;add   dx, 6  ; point at status port
4977 000019EB 66BADA03            <1>       mov    dx, 03DAh ; status port
4978                              <1>       ; cx = 0
4979 000019EF C3                  <1>       retn
4980                              <1>
4981                              <1> SCROLL_UP:
4982                              <1>       ; 07/07/2016
4983                              <1>       ; 26/06/2016
4984                              <1>       ; 12/05/2016
4985                              <1>       ; 30/01/2016
4986                              <1>       ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
4987                              <1>       ; 07/09/2014
4988                              <1>       ; 02/09/2014
4989                              <1>       ; 01/09/2014 (Retro UNIX 386 v1 - beginning)
4990                              <1>       ; 04/04/2014
4991                              <1>       ; 04/12/2013
4992                              <1>       ;
4993                              <1>       ; VIDEO.ASM - 06/10/85  VIDEO DISPLAY BIOS
4994                              <1>       ;
4995                              <1> ;--------------------------------------------
4996                              <1> ; SCROLL UP
4997                              <1> ;     THIS ROUTINE MOVES A BLOCK OF CHARACTERS UP
4998                              <1> ;     ON THE SCREEN
4999                              <1> ; INPUT
5000                              <1> ;     (AH) = CURRENT CRT MODE
5001                              <1> ;     (AL) = NUMBER OF ROWS TO SCROLL
5002                              <1> ;     (CX) = ROW/COLUMN OF UPPER LEFT CORNER
5003                              <1> ;     (DX) = ROW/COLUMN OF LOWER RIGHT CORNER
5004                              <1> ;     (BH) = ATTRIBUTE TO BE USED ON BLANKED LINE
5005                              <1> ;     (DS) = DATA SEGMENT
5006                              <1> ;     (ES) = REGEN BUFFER SEGMENT
5007                              <1> ; OUTPUT
5008                              <1> ;     NONE -- THE REGEN BUFFER IS MODIFIED
5009                              <1> ;--------------------------------------------
5010                              <1>
5011                              <1>       ; 07/07/2016
5012 000019F0 38F5                <1>       cmp    ch, dh
5013 000019F2 0F875CFBFFFF        <1>       ja     VIDEO_RETURN
5014 000019F8 38D1                <1>       cmp    cl, dl
5015 000019FA 0F8754FBFFFF        <1>       ja     VIDEO_RETURN
5016                              <1>       ;
5017 00001A00 E805000000          <1>       call   _scroll_up
5018 00001A05 E94AFBFFFF          <1>         jmp     VIDEO_RETURN
5019                              <1>
5020                              <1> _scroll_up:  ; from 'write_tty'
5021                              <1>       ;
5022                              <1>       ; cl = left upper column
5023                              <1>       ; ch = left upper row
5024                              <1>       ; dl = right lower column
5025                              <1>       ; dh = right lower row
5026                              <1>       ;
5027                              <1>       ; al = line count
5028                              <1>       ; bl = attribute to be used on blanked line
5029                              <1>       ; bh = video page number (0 to 7)
5030                              <1>
5031 00001A0A E896000000          <1>       call   test_line_count ; 16/01/2016
5032                              <1>
5033 00001A0F 8A25[C25E0000]      <1>       mov    ah, [CRT_MODE] ; current video mode
5034                              <1>       ;cmp   ah, 4
5035                              <1>       ;jb    short n0
5036                              <1>       ;cmp   byte [CRT_MODE], 4
5037 00001A15 80FC04              <1>         cmp  ah, 4 ; 07/07/2016
5038 00001A18 0F8320050000        <1>       jnb     GRAPHICS_UP ; 26/06/2016
5039                              <1>
5040                              <1>       ;cmp   ah, 7 ; TEST FOR BW CARD
5041                              <1>       ;jne   GRAPHICS_UP
5042                              <1> n0:
5043                              <1>       ; 07/07/2016
5044 00001A1E 80FF07              <1>       cmp    bh, 7 ; video page number
5045 00001A21 7606                <1>       jna    short n1
```

```
5046 00001A23 8A3D[4E520100]      <1>       mov   bh, [ACTIVE_PAGE]
5047                              <1> n1:
5048 00001A29 88DC                <1>       mov   ah, bl ; attribute
5049 00001A2B 6650                <1>       push  ax ; *
5050                              <1>       ;mov  esi, [CRT_BASE]
5051 00001A2D BE00800B00          <1>         mov    esi, 0B8000h
5052 00001A32 3A3D[4E520100]      <1>         cmp    bh, [ACTIVE_PAGE]
5053 00001A38 750B                <1>       jne   short n2
5054                              <1>       ;
5055 00001A3A 66A1[3C520100]      <1>         mov    ax, [CRT_START]
5056 00001A40 6601C6              <1>         add    si, ax
5057 00001A43 EB11                <1>         jmp    short n4
5058                              <1> n2:
5059 00001A45 20FF                <1>         and    bh, bh
5060 00001A47 740D                <1>       jz    short n4
5061 00001A49 88F8                <1>       mov   al, bh
5062                              <1> n3:
5063 00001A4B 660335[BC5F0100]    <1>         add  si, [CRT_LEN]
5064 00001A52 FEC8                <1>         dec  al
5065 00001A54 75F5                <1>       jnz   short n3
5066                              <1> n4:
5067 00001A56 E85D000000          <1>       call  scroll_position ; 16/01/2016
5068 00001A5B 7420                <1>         jz     short n6
5069                              <1>
5070 00001A5D 01CE                <1>         add    esi, ecx ; from address for scroll
5071 00001A5F 88F5                <1>       mov   ch, dh  ; #rows in block
5072 00001A61 28C5                <1>       sub   ch, al ; #rows to be moved
5073                              <1> n5:
5074 00001A63 E894000000          <1>       call  n10 ; 16/01/2016
5075                              <1>
5076 00001A68 51                  <1>         push ecx
5077 00001A69 0FB60D[C45E0000]    <1>       movzx  ecx, byte [CRT_COLS]
5078 00001A70 00C9                <1>       add   cl, cl
5079 00001A72 01CE                <1>         add  esi, ecx  ; next line
5080 00001A74 01CF                <1>         add  edi, ecx
5081 00001A76 59                  <1>       pop   ecx
5082                              <1>
5083 00001A77 FECD                <1>       dec   ch      ; count of lines to move
5084 00001A79 75E8                <1>       jnz   short n5 ; row loop
5085                              <1>       ; ch = 0
5086 00001A7B 88C6                <1>       mov   dh, al  ; #rows
5087                              <1> n6:
5088                              <1>       ; attribute in ah
5089 00001A7D B020                <1>       mov   al, ' '       ; fill with blanks
5090                              <1> n7:
5091 00001A7F E885000000          <1>       call  n11 ; 16/01/2016
5092                              <1>
5093 00001A84 8A0D[C45E0000]      <1>       mov   cl, [CRT_COLS]
5094 00001A8A 00C9                <1>       add   cl, cl
5095 00001A8C 01CF                <1>         add  edi, ecx
5096                              <1>
5097 00001A8E FECE                <1>       dec   dh
5098 00001A90 75ED                <1>       jnz   short n7
5099                              <1> n16:
5100 00001A92 3A3D[4E520100]      <1>       cmp   bh, [ACTIVE_PAGE]
5101 00001A98 750A                <1>       jne   short n8
5102                              <1>
5103                              <1>       ;cmp  byte [CRT_MODE], 7  ; is this the black and white card
5104                              <1>       ;je   short n8            ; if so, skip the mode reset
5105                              <1>
5106 00001A9A A0[C35E0000]        <1>       mov   al, [CRT_MODE_SET] ; get the value of mode set
5107 00001A9F 66BAD803            <1>       mov   dx, 03D8h ; always set color card port
5108 00001AA3 EE                  <1>       out   dx, al
5109                              <1> n8:
5110 00001AA4 C3                  <1>       retn
5111                              <1>
5112                              <1> test_line_count:
5113                              <1>       ; 12/05/2016
5114                              <1>       ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
5115                              <1>       ; 07/09/2014 (scroll_up)
5116 00001AA5 08C0                <1>       or    al, al
5117 00001AA7 740E                <1>       jz    short al_set2
5118 00001AA9 6652                <1>       push  dx
5119 00001AAB 28EE                <1>       sub   dh, ch  ; subtract upper row from lower row number
5120 00001AAD FEC6                <1>       inc   dh     ; adjust difference by 1
5121 00001AAF 38C6                <1>       cmp   dh, al       ; line count = amount of rows in window?
5122 00001AB1 7502                <1>       jne   short al_set1 ; if not the we're all set
5123 00001AB3 30C0                <1>       xor   al, al ; otherwise set al to zero
5124                              <1> al_set1:
5125 00001AB5 665A                <1>       pop   dx
5126                              <1> al_set2:
5127 00001AB7 C3                  <1>       retn
5128                              <1>
5129                              <1> scroll_position:
5130                              <1>       ; 26/06/2016
5131                              <1>       ; 30/01/2016
5132                              <1>         ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
5133                              <1>       ; 07/09/2014 (scroll_up)
5134                              <1>
5135 00001AB8 6652                <1>       push  dx
5136 00001ABA 6689CA              <1>       mov   dx, cx ; now, upper left position in DX
5137 00001ABD E8EEFEFFFF          <1>       call  position
5138 00001AC2 01C6                <1>       add   esi, eax
5139 00001AC4 89F7                <1>       mov   edi, esi
5140 00001AC6 665A                <1>       pop   dx     ; lower right position in DX
5141 00001AC8 6629CA              <1>       sub   dx, cx
5142 00001ACB FEC6                <1>       inc   dh     ; dh = #rows
5143 00001ACD FEC2                <1>       inc   dl     ; dl = #cols in block
5144 00001ACF 59                  <1>       pop   ecx    ; return address
5145 00001AD0 6658                <1>       pop   ax     ; * ; al = line count, ah = attribute
5146 00001AD2 51                  <1>       push  ecx    ; return address
5147 00001AD3 0FB7C8              <1>       movzx ecx, ax
5148 00001AD6 8A25[C45E0000]      <1>       mov   ah, [CRT_COLS]
```

```
5149 00001ADC F6E4              <1>       mul    ah     ; determine offset to from address
5150 00001ADE 6601C0            <1>       add    ax, ax  ; *2 for attribute byte
5151                            <1>       ;
5152 00001AE1 6650              <1>       push   ax     ; offset
5153 00001AE3 6652              <1>       push   dx
5154                            <1>       ;
5155                            <1>       ; 04/04/2014
5156 00001AE5 66BADA03          <1>       mov    dx, 3DAh ; guaranteed to be color card here
5157                            <1> n9:                   ; wait_display_enable
5158 00001AE9 EC                <1>         in      al, dx   ; get port
5159 00001AEA A808              <1>       test   al, RVRT ; wait for vertical retrace
5160 00001AEC 74FB              <1>       jz     short n9 ; wait_display_enable
5161 00001AEE B025              <1>       mov    al, 25h
5162 00001AF0 B2D8              <1>       mov    dl, 0D8h ; address control port
5163 00001AF2 EE                <1>       out    dx, al ; turn off video during vertical retrace
5164 00001AF3 665A              <1>       pop    dx     ; #rows, #cols
5165 00001AF5 6658              <1>              pop    ax     ; offset
5166 00001AF7 6691              <1>       xchg   ax, cx ;
5167                            <1>       ; ecx = offset, al = line count, ah = attribute
5168                            <1>       ;
5169 00001AF9 08C0              <1>       or     al, al
5170 00001AFB C3                <1>       retn
5171                            <1> n10:
5172                            <1>       ; Move rows
5173 00001AFC 88D1              <1>       mov    cl, dl ; get # of cols to move
5174 00001AFE 56                <1>       push   esi
5175 00001AFF 57                <1>       push   edi    ; save start address
5176                            <1> n10r:
5177 00001B00 66A5              <1>       movsw         ; move that line on screen
5178 00001B02 FEC9              <1>       dec    cl
5179 00001B04 75FA              <1>         jnz     short n10r
5180 00001B06 5F                <1>       pop    edi
5181 00001B07 5E                <1>       pop    esi    ; recover addresses
5182 00001B08 C3                <1>       retn
5183                            <1> n11:
5184                            <1>       ; Clear rows
5185                            <1>                    ; dh =  #rows
5186 00001B09 88D1              <1>         mov  cl, dl ; get # of cols to clear
5187 00001B0B 57                <1>         push    edi     ; save address
5188                            <1> n11r:
5189 00001B0C 66AB              <1>         stosw           ; store fill character
5190 00001B0E FEC9              <1>       dec    cl
5191 00001B10 75FA              <1>         jnz     short n11r
5192 00001B12 5F                <1>         pop     edi     ; recover address
5193 00001B13 C3                <1>       retn
5194                            <1>
5195                            <1> SCROLL_DOWN:
5196                            <1>       ; 07/07/2016
5197                            <1>       ; 27/06/2016
5198                            <1>       ; 26/06/2016
5199                            <1>       ; 12/05/2016
5200                            <1>       ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
5201                            <1>       ;
5202                            <1>       ; VIDEO.ASM - 06/10/85  VIDEO DISPLAY BIOS
5203                            <1>
5204                            <1> ;-----------------------------------------
5205                            <1> ; SCROLL DOWN
5206                            <1> ;     THIS ROUTINE MOVES THE CHARACTERS WITHIN A DEFINED
5207                            <1> ;     BLOCK DOWN ON THE SCREEN, FILLING THE TOP LINES
5208                            <1> ;     WITH A DEFINED CHARACTER
5209                            <1> ; INPUT
5210                            <1> ;     (AH) = CURRENT CRT MODE
5211                            <1> ;     (AL) = NUMBER OF LINES TO SCROLL
5212                            <1> ;     (CX) = UPPER LEFT CORNER OF RECION
5213                            <1> ;     (DX) = LOWER RIGHT CORNER OF REGION
5214                            <1> ;     (BH) = FILL CHARACTER
5215                            <1> ;     (DS) = DATA SEGMENT
5216                            <1> ;     (ES) = REGEN SEGMENT
5217                            <1> ; OUTPUT
5218                            <1> ;     NONE -- SCREEN IS SCROLLED
5219                            <1> ;-----------------------------------------
5220                            <1>
5221                            <1>       ; 07/07/2016
5222 00001B14 38F5              <1>       cmp    ch, dh
5223 00001B16 0F8738FAFFFF      <1>       ja     VIDEO_RETURN
5224 00001B1C 38D1              <1>       cmp    cl, dl
5225 00001B1E 0F8730FAFFFF      <1>       ja     VIDEO_RETURN
5226                            <1>       ;
5227 00001B24 E805000000        <1>       call   _scroll_down
5228 00001B29 E926FAFFFF        <1>         jmp     VIDEO_RETURN
5229                            <1>
5230                            <1> _scroll_down: ; 27/06/2016
5231                            <1>
5232                            <1>       ; cl = left upper column
5233                            <1>       ; ch = left upper row
5234                            <1>       ; dl = right lower column
5235                            <1>       ; dh = right lower row
5236                            <1>       ;
5237                            <1>       ; al = line count
5238                            <1>       ; bl = attribute to be used on blanked line
5239                            <1>       ; bh = video page number (0 to 7)
5240                            <1>
5241                            <1>       ; !!!!
5242 00001B2E FD                <1>       std           ; DIRECTION FOR SCROLL DOWN
5243                            <1>       ; !!!!
5244 00001B2F E871FFFFFF        <1>       call   test_line_count ; 16/01/2016
5245                            <1>
5246 00001B34 8A25[C25E0000]    <1>       mov    ah, [CRT_MODE] ; current video mode
5247                            <1>       ;cmp   ah, 4
5248                            <1>       ;jb    short _n0
5249                            <1>       ;cmp   byte [CRT_MODE], 4
5250 00001B3A 80FC04            <1>         cmp  ah, 4 ; 07/07/2016
5251 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[4E520100]     <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[4E520100]         <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
```

```
5458                                    <1>
5459 00001C1C E8CC0A0000                <1>       call   vga_write_char_only
5460 00001C21 E92EF9FFFF                <1>       jmp     VIDEO_RETURN
5461                                    <1>
5462                                    <1> write_c_c:
5463                                    <1>       ;and   bh, 7 ; video page number (<= 7)
5464 00001C26 0FB6F7                    <1>       movzx  esi, bh
5465 00001C29 8A9E[CB5E0000]            <1>       mov    bl, [esi+chr_attrib]
5466                                    <1>
5467 00001C2F E805000000                <1>       call   _write_c_current
5468 00001C34 E91BF9FFFF                <1>         jmp     VIDEO_RETURN
5469                                    <1>
5470                                    <1> _write_c_current:  ; from 'write_tty'
5471                                    <1>       ; 26/06/2016
5472                                    <1>       ; 24/06/2016
5473                                    <1>       ; 12/05/2016
5474                                    <1>       ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
5475                                    <1>       ; 30/08/2014 (Retro UNIX 386 v1)
5476                                    <1>       ; 18/01/2014
5477                                    <1>       ; 04/12/2013
5478                                    <1>       ;
5479                                    <1>       ; VIDEO.ASM - 06/10/85  VIDEO DISPLAY BIOS
5480                                    <1>
5481                                    <1>       ;mov   ah, [CRT_MODE] ; current video mode
5482                                    <1>       ;cmp   ah, 4
5483                                    <1>       ;jb    short p40
5484 00001C39 803D[C25E0000]04          <1>       cmp    byte [CRT_MODE], 4
5485 00001C40 0F8381070000              <1>         jnb     GRAPHICS_WRITE ; 26/06/2016
5486                                    <1>
5487                                    <1>       ;cmp   ah, 7 ; TEST FOR BW CARD
5488                                    <1>       ;jne   GRAPHICS_WRITE
5489                                    <1> p40:
5490                                    <1>       ; al = character
5491                                    <1>       ; bl = color/attribute
5492                                    <1>       ; bh = video page
5493                                    <1>       ; cx = count of characters to write
5494 00001C46 6652                      <1>       push   dx
5495 00001C48 88DC                      <1>       mov    ah, bl  ; color/attribute (12/05/2016)
5496 00001C4A 6650                      <1>       push   ax     ; save character & attribute/color
5497 00001C4C 6651                      <1>       push   cx
5498 00001C4E E86FFDFFFF                <1>       call   find_position  ; get regen location and port address
5499 00001C53 6659                      <1>       pop    cx
5500                                    <1>       ; esi = regen location
5501                                    <1>       ; dx = status port
5502                                    <1>       ;
5503 00001C55 81C600800B00              <1>       add    esi, 0B8000h ; 30/08/2014 (crt_base)
5504                                    <1>       ;
5505 00001C5B 8A25[C25E0000]            <1>       mov    ah, [CRT_MODE]
5506 00001C61 80EC02                    <1>       sub    ah, 2
5507 00001C64 D0EC                      <1>       shr    ah, 1
5508 00001C66 7519                      <1>       jnz    short p44    ; 26/06/2016
5509                                    <1>
5510                                    <1>       ; WAIT FOR HORIZONTAL RETRACE OR VERTICAL RETRACE IF COLOR 80
5511                                    <1> p41:
5512 00001C68 FB                        <1>       sti          ; enable interrupts first
5513 00001C69 3A3D[4E520100]            <1>       cmp     bh, [ACTIVE_PAGE]
5514 00001C6F 7510                      <1>       jne    short p44
5515 00001C71 FA                        <1>       cli          ; block interrupts for single loop
5516 00001C72 EC                        <1>       in     al, dx ; get status from the adapter
5517 00001C73 A808                      <1>       test   al, RVRT ; check for vertical retrace first
5518 00001C75 7509                      <1>       jnz    short p43 ; Do fast write now if vertical retrace
5519 00001C77 A801                      <1>       test   al, RHRZ ; is horizontal retrace low
5520 00001C79 75ED                      <1>       jnz    short p41 ; wait until it is
5521                                    <1> p42:             ;  wait for either retrace high
5522 00001C7B EC                        <1>       in     al, dx ; get status again
5523 00001C7C A809                      <1>       test   al, RVRT+RHRZ ; is horizontal or vertical retrace high
5524 00001C7E 74FB                      <1>       jz     short p42 ; wait until either retrace active
5525                                    <1> p43:
5526 00001C80 FB                        <1>       sti
5527                                    <1> p44:
5528 00001C81 668B0424                  <1>       mov    ax, [esp] ; restore the character (al) & attribute (ah)
5529 00001C85 668906                    <1>       mov    [esi], ax
5530                                    <1>
5531 00001C88 6649                      <1>       dec    cx
5532 00001C8A 7404                      <1>       jz     short p45
5533                                    <1>
5534 00001C8C 46                        <1>       inc    esi
5535 00001C8D 46                        <1>       inc    esi
5536 00001C8E EBD8                      <1>       jmp    short p41
5537                                    <1> p45:
5538 00001C90 6658                      <1>       pop    ax
5539 00001C92 665A                      <1>       pop    dx
5540 00001C94 C3                        <1>       retn
5541                                    <1>
5542                                    <1> ; 09/07/2016
5543                                    <1> ; 26/06/2016
5544                                    <1> ; 24/06/2016
5545                                    <1> ; 12/05/2016
5546                                    <1> ; 18/01/2016
5547                                    <1> ; 16/01/2016 - TRDOS 386 (TRDOS v2.0)
5548                                    <1> ; 30/06/2015
5549                                    <1> ; 27/06/2015
5550                                    <1> ; 11/03/2015
5551                                    <1> ; 02/09/2014
5552                                    <1> ; 30/08/2014
5553                                    <1> ; VIDEO FUNCTIONS
5554                                    <1> ; (write_tty - Retro UNIX 8086 v1 - U9.ASM, 01/02/2014)
5555                                    <1>
5556                                    <1> WRITE_TTY:
5557                                    <1>       ; 09/12/2017
5558                                    <1>       ; 09/07/2016
5559                                    <1>       ; 01/07/2016
5560                                    <1>       ; 26/06/2016
```

```
                                <1>       ; 24/06/2016
                                <1>       ; 13/05/2016
                                <1>       ; 12/05/2016
                                <1>       ; 30/01/2016
                                <1>       ; 18/01/2016
                                <1>       ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>       ; 13/08/2015
                                <1>       ; 02/09/2014
                                <1>       ; 30/08/2014 (Retro UNIX 386 v1 - beginning)
                                <1>       ; 01/02/2014 (Retro UNIX 8086 v1 - last update)
                                <1>       ; 03/12/2013 (Retro UNIX 8086 v1 - beginning)
                                <1>       ; (Modified registers: EAX, EBX, ECX, EDX, ESI, EDI)
                                <1>       ;
                                <1>       ; INPUT -> AL = Character to be written
                                <1>       ;        BL = Color (Forecolor, Backcolor)
                                <1>       ;        BH = Video Page (0 to 7)
                                <1>
                                <1>       ; 09/07/2016
00001C95 803D[C25E0000]07       <1>         cmp     byte [CRT_MODE], 7
00001C9C 760A                   <1>       jna    short write_tty_cga
                                <1>
00001C9E E8290D0000             <1>       call   vga_write_teletype
00001CA3 E9ACF8FFFF             <1>       jmp    VIDEO_RETURN
                                <1>
                                <1> write_tty_cga:
                                <1>       ; 13/05/2016
                                <1>       ;call  _write_tty
                                <1>       ; 01/07/2016
00001CA8 E818000000             <1>       call   _write_tty_m3
00001CAD E9A2F8FFFF             <1>       jmp    VIDEO_RETURN
                                <1>
                                <1> RVRT  equ    00001000b    ; VIDEO VERTICAL RETRACE BIT
                                <1> RHRZ  equ    00000001b    ; VIDEO HORIZONTAL RETRACE BIT
                                <1>
                                <1> ; Derived from "WRITE_TTY" procedure of IBM "pc-at" rombios source code
                                <1> ; (06/10/1985), 'video.asm', INT 10H, VIDEO_IO
                                <1> ;
                                <1> ; 06/10/85  VIDEO DISPLAY BIOS
                                <1> ;
                                <1> ;--- WRITE_TTY ----------------------------------------------------------------
                                <1> ;                                                                           :
                                <1> ;   THIS INTERFACE PROVIDES A TELETYPE LIKE INTERFACE TO THE                :
                                <1> ;   VIDEO CARDS. THE INPUT CHARACTER IS WRITTEN TO THE CURRENT               :
                                <1> ;   CURSOR POSITION, AND THE CURSOR IS MOVED TO THE NEXT POSITION.          :
                                <1> ;   IF THE CURSOR LEAVES THE LAST COLUMN OF THE FIELD, THE COLUMN           :
                                <1> ;   IS SET TO ZERO, AND THE ROW VALUE IS INCREMENTED. IF THE ROW       :
                                <1> ;   ROW VALUE LEAVES THE FIELD, THE CURSOR IS PLACED ON THE LAST ROW,        :
                                <1> ;   FIRST COLUMN, AND THE ENTIRE SCREEN IS SCROLLED UP ONE LINE.       :
                                <1> ;   WHEN THE SCREEN IS SCROLLED UP, THE ATTRIBUTE FOR FILLING THE            :
                                <1> ;   NEWLY BLANKED LINE IS READ FROM THE CURSOR POSITION ON THE PREVIOUS      :
                                <1> ;   LINE BEFORE THE SCROLL, IN CHARACTER MODE. IN GRAPHICS MODE,       :
                                <1> ;   THE 0 COLOR IS USED.                                                    :
                                <1> ;   ENTRY --                                                                :
                                <1> ;     (AH) = CURRENT CRT MODE                                               :
                                <1> ;     (AL) = CHARACTER TO BE WRITTEN                                        :
                                <1> ;         NOTE THAT BACK SPACE, CARRIAGE RETURN, BELL AND LINE FEED ARE :
                                <1> ;         HANDLED AS COMMANDS RATHER THAN AS DISPLAY GRAPHICS CHARACTERS     :
                                <1> ;     (BL) = FOREGROUND COLOR FOR CHAR WRITE IF CURRENTLY IN A GRAPHICS MODE  :
                                <1> ;   EXIT --                                                                 :
                                <1> ;     ALL REGISTERS SAVED                                                   :
                                <1> ;-----------------------------------------------------------------------------
                                <1>
                                <1> ; 09/12/2017
                                <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[3E520100]         <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
```

```
5664 00001CE1 FEC2                <1>       inc    dl           ; next column
5665 00001CE3 3A15[C45E0000]      <1>       cmp    dl, [CRT_COLS]  ; test for column overflow
5666 00001CE9 755D                <1>         jne     _set_cpos
5667 00001CEB B200                <1>       mov    dl, 0        ; column = 0
5668                              <1> u10:                      ; (line feed found)
5669 00001CED 80FE18              <1>       cmp    dh, 25-1     ; check for last row
5670 00001CF0 7218                <1>       jb     short u6
5671                              <1>       ;
5672                              <1>       ; scroll required
5673                              <1> u1:
5674                              <1>       ; SET CURSOR POSITION (04/12/2013)
5675 00001CF2 E851000000          <1>       call   _set_cpos
5676                              <1>       ;
5677                              <1>       ; determine value to fill with during scroll
5678                              <1> u2:
5679                              <1>       ; bh = video page number
5680                              <1>       ;
5681 00001CF7 E8B3FEFFFF          <1>       call   _read_ac_current ; 18/01/2016
5682                              <1>       ;
5683                              <1>       ; al = character, ah = attribute
5684                              <1>       ; bh = video page number
5685                              <1> u3:
5686                              <1>       ;;mov  ax, 0601h     ; scroll one line
5687                              <1>       ;;sub  cx, cx        ; upper left corner
5688                              <1>       ;;mov  dh, 25-1      ; lower right row
5689                              <1>       ;;;mov dl, [CRT_COLS]
5690                              <1>       ;mov   dl, 80        ; lower right column
5691                              <1>       ;;dec  dl
5692                              <1>       ;;mov  dl, 79
5693                              <1>
5694                              <1>       ;;call scroll_up     ; 04/12/2013
5695                              <1>       ;;; 11/03/2015
5696                              <1>       ; 02/09/2014
5697                              <1>       ;;;mov cx, [crt_ulc] ; Upper left corner  (0000h)
5698                              <1>       ;;;mov dx, [crt_lrc] ; Lower right corner (184Fh)
5699                              <1>       ; 11/03/2015
5700 00001CFC 6629C9              <1>       sub    cx, cx
5701 00001CFF 66BA4F18            <1>       mov    dx, 184Fh ; dl = 79 (column), dh = 24 (row)
5702                              <1>       ;
5703 00001D03 B001                <1>       mov    al, 1        ; scroll 1 line up
5704                              <1>              ; ah = attribute
5705                              <1>       ;mov   bl, al ; 12/05/2016
5706 00001D05 E900FDFFFF          <1>       jmp    _scroll_up   ; 16/01/2016
5707                              <1> ;u4:
5708                              <1>       ;;int  10h          ; video-call return
5709                              <1>                           ; scroll up the screen
5710                              <1>                           ; tty return
5711                              <1> ;u5:
5712                              <1>       ;retn               ; return to the caller
5713                              <1>
5714                              <1> u6:                       ; set-cursor-inc
5715 00001D0A FEC6                <1>       inc    dh           ; next row
5716                              <1>                           ; set cursor
5717                              <1> ;u7:
5718                              <1>       ;;mov  ah, 02h
5719                              <1>       ;;jmp  short u4      ; establish the new cursor
5720                              <1>       ;call  _set_cpos
5721                              <1>       ;jmp   short u5
5722 00001D0C EB3A                <1>       jmp     _set_cpos
5723                              <1>
5724                              <1>       ; check for control characters
5725                              <1> u8:
5726 00001D0E 7436                <1>       je     short u9
5727 00001D10 3C0A                <1>       cmp    al, 0Ah             ; is it a line feed (0Ah)
5728 00001D12 74D9                <1>       je     short u10
5729 00001D14 3C07                <1>       cmp    al, 07h      ; is it a bell
5730 00001D16 747A                <1>       je     short u11
5731 00001D18 3C08                <1>       cmp    al, 08h             ; is it a backspace
5732                              <1>       ;jne   short u0
5733 00001D1A 7422                <1>       je     short bs     ; 12/12/2013
5734                              <1>       ; 12/12/2013 (tab stop)
5735 00001D1C 3C09                <1>       cmp    al, 09h             ; is it a tab stop
5736 00001D1E 75B8                <1>       jne    short u0
5737 00001D20 88D0                <1>       mov    al, dl
5738                              <1>       ;cbw
5739 00001D22 30E4                <1>       xor    ah, ah ; 09/12/2017
5740 00001D24 B108                <1>       mov    cl, 8
5741 00001D26 F6F1                <1>       div    cl
5742 00001D28 28E1                <1>       sub    cl, ah
5743                              <1> ts:
5744                              <1>       ; 02/09/2014
5745                              <1>       ; 01/09/2014
5746 00001D2A B020                <1>       mov    al, 20h
5747                              <1> tsloop:
5748 00001D2C 6651                <1>       push   cx
5749 00001D2E 6650                <1>       push   ax
5750                              <1>       ;mov   bh, [ACTIVE_PAGE]
5751 00001D30 E890FFFFFF          <1>       call   _write_tty_m3 ; 24/06/2016 (m3 -> _write_tty_m3)
5752 00001D35 6658                <1>       pop    ax  ; ah = attribute/color
5753 00001D37 6659                <1>       pop    cx
5754 00001D39 FEC9                <1>       dec    cl
5755 00001D3B 75EF                <1>       jnz    short tsloop
5756 00001D3D C3                  <1>       retn
5757                              <1> bs:
5758                              <1>       ; back space found
5759                              <1>
5760 00001D3E 08D2                <1>       or     dl, dl              ; is it already at start of line
5761                              <1>       ;je    short u7     ; set_cursor
5762 00001D40 7406                <1>       jz     short _set_cpos
5763 00001D42 664A                <1>       dec    dx                  ; no -- just move it back
5764                              <1>       ;jmp   short u7
5765 00001D44 EB02                <1>       jmp    short _set_cpos
5766                              <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[3E520100]           <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[4E520100]         <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[3C520100]       <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
00001D6E 66D1E9                 <1>       shr    cx, 1  ; divide by 2 for char only count
00001D71 B40E                   <1>       mov    ah, 14 ; register number for cursor
                                <1>       ;call  m16    ; output value to the 6845
                                <1>       ;retn
                                <1>
                                <1>       ;-----THIS ROUTINE OUTPUTS THE CX REGISTER
                                <1>       ;     TO THE 6845 REGISTERS NAMED IN (AH)
                                <1> m16:
00001D73 FA                     <1>       cli
                                <1>       ;mov   dx, [addr_6845] ; address register
00001D74 66BAD403               <1>       mov    dx, 03D4h ; I/O address of color card
00001D78 88E0                   <1>       mov    al, ah ; get value
00001D7A EE                     <1>       out    dx, al ; register set
00001D7B 6642                   <1>       inc    dx     ; data register
00001D7D EB00                   <1>       jmp    $+2    ; i/o delay
00001D7F 88E8                   <1>       mov    al, ch ; data
00001D81 EE                     <1>       out    dx, al
00001D82 664A                   <1>       dec    dx
00001D84 88E0                   <1>       mov    al, ah
00001D86 FEC0                   <1>       inc    al     ; point to other data register
00001D88 EE                     <1>       out    dx, al ; set for second register
00001D89 6642                   <1>       inc    dx
00001D8B EB00                   <1>       jmp    $+2    ; i/o delay
00001D8D 88C8                   <1>       mov    al, cl ; second data value
00001D8F EE                     <1>       out    dx, al
00001D90 FB                     <1>       sti
                                <1> m17:
00001D91 C3                     <1>       retn
                                <1>
                                <1> beeper:
                                <1>       ; 04/08/2016
                                <1>       ; 12/05/2016 - TRDOS 386 (TRDOS v2.0)
                                <1>       ; 30/08/2014 (Retro UNIX 386 v1)
                                <1>       ; 18/01/2014
                                <1>       ; 03/12/2013
                                <1>       ; bell found
                                <1> u11:
00001D92 FB                     <1>       sti
00001D93 3A3D[4E520100]         <1>       cmp    bh, [ACTIVE_PAGE]
00001D99 7551                   <1>       jne    short u12    ; Do not sound the beep
                                <1>                           ; if it is not written on the active page
                                <1> beeper_gfx: ; 04/08/2016
00001D9B 66B93305               <1>       mov    cx, 1331     ; divisor for 896 hz tone
00001D9F B31F                   <1>       mov    bl, 31       ; set count for 31/64 second for beep
                                <1>       ;call  beep         ; sound the pod bell
                                <1>       ;jmp   short u5     ; tty_return
                                <1>       ;retn
                                <1>
                                <1> TIMER equ    040h               ; 8254 TIMER - BASE ADDRESS
                                <1> PORT_B      equ    061h         ; PORT B READ/WRITE DIAGNOSTIC REGISTER
```

```
                        <1> GATE2 equ    00000001b    ; TIMER 2 INPUT CATE CLOCK BIT
                        <1> SPK2  equ    00000010b    ; SPEAKER OUTPUT DATA ENABLE BIT
                        <1>
                        <1> beep:
                        <1>       ; 07/02/2015
                        <1>       ; 30/08/2014 (Retro UNIX 386 v1)
                        <1>       ; 18/01/2014
                        <1>       ; 03/12/2013
                        <1>       ;
                        <1>       ; TEST4.ASM - 06/10/85  POST AND BIOS UTILITY ROUTINES
                        <1>       ;
                        <1>       ; ROUTINE TO SOUND THE BEEPER USING TIMER 2 FOR TONE
                        <1>       ;
                        <1>       ; ENTRY:
                        <1>       ;    (BL) = DURATION COUNTER ( 1 FOR 1/64 SECOND )
                        <1>       ;    (CX) = FREQUENCY DIVISOR (1193180/FREQUENCY) (1331 FOR 886 HZ)
                        <1>       ; EXIT:                          :
                        <1>       ;    (AX),(BL),(CX) MODIFIED.
                        <1>
00001DA1 9C             <1>       pushf  ; 18/01/2014 ; save interrupt status
00001DA2 FA             <1>       cli                 ; block interrupts during update
00001DA3 B0B6           <1>       mov    al, 10110110b; select timer 2, lsb, msb binary
00001DA5 E643           <1>       out    TIMER+3, al  ; write timer mode register
00001DA7 EB00           <1>       jmp    $+2          ; I/O delay
00001DA9 88C8           <1>       mov    al, cl       ; divisor for hz (low)
00001DAB E642           <1>       out    TIMER+2,AL   ; write timer 2 count - lsb
00001DAD EB00           <1>       jmp    $+2          ; I/O delay
00001DAF 88E8           <1>       mov    al, ch       ; divisor for hz (high)
00001DB1 E642           <1>       out    TIMER+2, al  ; write timer 2 count - msb
00001DB3 E461           <1>       in     al, PORT_B   ; get current setting of port
00001DB5 88C4           <1>       mov    ah, al       ; save that setting
00001DB7 0C03           <1>       or     al, GATE2+SPK2      ; gate timer 2 and turn speaker on
00001DB9 E661           <1>       out    PORT_B, al   ; and restore interrupt status
                        <1>       ;popf  ; 18/01/2014
00001DBB FB             <1>       sti
                        <1> g7:                       ; 1/64 second per count (bl)
00001DBC B90B040000     <1>       mov    ecx, 1035    ; delay count for 1/64 of a second
00001DC1 E827000000     <1>       call   waitf        ; go to beep delay 1/64 count
00001DC6 FECB           <1>       dec    bl           ; (bl) length count expired?
00001DC8 75F2           <1>       jnz    short g7     ; no - continue beeping speaker
                        <1>       ;
                        <1>       ;pushf              ; save interrupt status
00001DCA FA             <1>       cli    ; 18/01/2014 ; block interrupts during update
00001DCB E461           <1>       in     al, PORT_B   ; get current port value
                        <1>         ;or      al, not (GATE2+SPK2) ; isolate current speaker bits in case
00001DCD 0CFC           <1>         or      al, ~(GATE2+SPK2)
00001DCF 20C4           <1>         and  ah, al       ; someone turned them off during beep
00001DD1 88E0           <1>       mov    al, ah       ; recover value of port
                        <1>         ;or      al, not (GATE2+SPK2) ; force speaker data off
00001DD3 0CFC           <1>       or     al, ~(GATE2+SPK2) ; isolate current speaker bits in case
00001DD5 E661           <1>       out    PORT_B, al   ; and stop speaker timer
                        <1>       ;popf               ; restore interrupt flag state
00001DD7 FB             <1>       sti
00001DD8 B90B040000     <1>       mov    ecx, 1035    ; force 1/64 second delay (short)
00001DDD E80B000000     <1>       call   waitf        ; minimum delay between all beeps
                        <1>       ;pushf              ; save interrupt status
00001DE2 FA             <1>       cli                 ; block interrupts during update
00001DE3 E461           <1>       in     al, PORT_B   ; get current port value in case
00001DE5 2403           <1>       and    al, GATE2+SPK2      ; someone turned them on
00001DE7 08E0           <1>       or     al, ah       ; recover value of port_b
00001DE9 E661           <1>       out    PORT_B, al   ; restore speaker status
00001DEB 9D             <1>       popf                ; restore interrupt flag state
                        <1> u12:
00001DEC C3             <1>       retn
                        <1>
                        <1> REFRESH_BIT equ    00010000b    ; REFRESH TEST BIT
                        <1>
                        <1> WAITF:
                        <1> waitf:
                        <1>       ; 30/08/2014 (Retro UNIX 386 v1)
                        <1>       ; 03/12/2013
                        <1>       ;
                        <1> ;     push ax                   ; save work register (ah)
                        <1> ;waitf1:
                        <1>                           ; use timer 1 output bits
                        <1> ;     in     al, PORT_B   ; read current counter output status
                        <1> ;     and    al, REFRESH_BIT     ; mask for refresh determine bit
                        <1> ;     cmp    al, ah       ; did it just change
                        <1> ;     je     short waitf1 ; wait for a change in output line
                        <1> ;     ;
                        <1> ;     mov    ah, al       ; save new lflag state
                        <1> ;     loop   waitf1       ; decrement half cycles till count end
                        <1> ;     ;
                        <1> ;     pop    ax           ; restore (ah)
                        <1> ;     retn                ; return (cx)=0
                        <1>
                        <1> ; 06/02/2015 (unix386.s <-- dsectrm2.s)
                        <1> ; 17/12/2014 (dsectrm2.s)
                        <1> ; WAITF
                        <1> ; /// IBM PC-XT Model 286 System BIOS Source Code - Test 4 - 06/10/85 ///
                        <1> ;
                        <1> ;---WAITF----------------------------------------------------------------------
                        <1> ;     FIXED TIME WAIT ROUTINE (HARDWARE CONTROLLED - NOT PROCESSOR)
                        <1> ; ENTRY:
                        <1> ;     (CX) = COUNT OF 15.085737 MICROSECOND INTERVALS TO WAIT
                        <1> ;            MEMORY REFRESH TIMER 1 OUTPUT USED AS REFERENCE
                        <1> ; EXIT:
                        <1> ;                  AFTER (CX) TIME COUNT (PLUS OR MINUS 16 MICROSECONDS)
                        <1> ;     (CX) = 0
                        <1> ;-----------------------------------------------------------------------------
                        <1>
                        <1> ; Refresh period: 30 micro seconds (15-80 us)
                        <1> ; (16/12/2014 - AWARDBIOS 1999 - ATORGS.ASM, WAIT_REFRESH)
```

```
                        <1>
                        <1> ;WAITF:                             ; DELAY FOR (CX)*15.085737 US
00001DED 6650           <1>       PUSH   AX                   ; SAVE WORK REGISTER (AH)
                        <1>       ; 16/12/2014
                        <1>       ;shr   cx, 1                ; convert to count of 30 micro seconds
00001DEF D1E9           <1>       shr    ecx, 1 ; 21/02/2015
                        <1> ;17/12/2014
                        <1> ;WAITF1:
                        <1> ;     IN     AL, PORT_B   ;061h  ; READ CURRENT COUNTER OUTPUT STATUS
                        <1> ;     AND    AL, REFRESH_BIT     ;00010000b ; MASK FOR REFRESH DETERMINE BIT
                        <1> ;     CMP    AL, AH              ; DID IT JUST CHANGE
                        <1> ;     JE     short WAITF1        ; WAIT FOR A CHANGE IN OUTPUT LINE
                        <1> ;     MOV    AH, AL              ; SAVE NEW FLAG STATE
                        <1> ;     LOOP   WAITF1              ; DECREMENT HALF CYCLES TILL COUNT END
                        <1>       ;
                        <1>       ; 17/12/2014
                        <1>       ;
                        <1>       ; Modification from 'WAIT_REFRESH' procedure of AWARD BIOS - 1999
                        <1>       ;
                        <1> ;WAIT_REFRESH:  Uses port 61, bit 4 to have CPU speed independent waiting.
                        <1> ;     INPUT:  CX = number of refresh periods to wait
                        <1> ;                  (refresh periods = 1 per 30 microseconds on most machines)
                        <1> WR_STATE_0:
00001DF1 E461           <1>       IN     AL,PORT_B           ; IN AL,SYS1
00001DF3 A810           <1>       TEST   AL,010H
00001DF5 74FA           <1>       JZ     SHORT WR_STATE_0
                        <1> WR_STATE_1:
00001DF7 E461           <1>       IN     AL,PORT_B           ; IN AL,SYS1
00001DF9 A810           <1>       TEST   AL,010H
00001DFB 75FA           <1>       JNZ    SHORT WR_STATE_1
00001DFD E2F2           <1>         LOOP    WR_STATE_0
                        <1>       ;
00001DFF 6658           <1>       POP    AX                  ; RESTORE (AH)
00001E01 C3             <1>       RETn                       ; (CX) = 0
                        <1>
                        <1> ; 09/07/2016
                        <1> ; 01/07/2016
                        <1> ; 24/06/2016
                        <1> ; 23/06/2016 - TRDOS 386 (TRDOS v2.0)
                        <1> ; VIDEO1.ASM - 24/03/1985 (IBM PC-AT BIOS source code)
                        <1> ;--------------------------------------------------------------------------
                        <1> ; WRITE_STRING                                                          :
                        <1> ;     THIS ROUTINE WRITES A STRING OF CHARACTERS TO THE CRT.                  :
                        <1> ; INPUT                                                         :
                        <1> ;     (AL) = WRITE STRING COMMAND  0 - 3                                :
                        <1> ;     (BH) = DISPLAY PAGE (ACTIVE PAGE)                                 :
                        <1> ;     (CX) = COUNT OF CHARACTERS TO WRITE, IF (CX) = 0 THEN RETURN             :
                        <1> ;     (DX) = CURSOR POSITION FOR START OF STRING WRITE                   :
                        <1> ;     (BL) = ATTRIBUTE OF CHARACTER TO WRITE IF (AL) = 0  OR     (AL) = 1       :
                        <1> ;     (eBP) = SOURCE STRING OFFSET                                      :
                        <1> ; OUTPUT                                                        :
                        <1> ;     NONE                                                      :
                        <1> ;--------------------------------------------------------------------------
                        <1>
                        <1> ; AL = 00h: Assign all characters the attribute in BL; do not update cursor
                        <1> ; AL = 01h: Assign all characters the attribute in BL; update cursor
                        <1> ; AL = 02h: Use attributes in string; do not update cursor
                        <1> ; AL = 03h: Use attributes in string; update cursor
                        <1>
                        <1> WRITE_STRING:
                        <1>       ; 12/09/2016
                        <1>       ; 09/07/2016
                        <1>       ;cmp   byte [CRT_MODE], 7 ; 6?!
                        <1>       ;ja    VIDEO_RETURN        ; not a valid function for VGA modes
                        <1>       ;
00001E02 A2[B85F0100]   <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 E8D9CA0000     <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[3E520100] <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:
```

```
6076 00001E4C 8A4500              <1>       MOV    AL, [eBP]           ; GET CHARACTER FROM INPUT STRING
6077 00001E4F 45                  <1>       INC    eBP                 ; BUMP POINTER TO CHARACTER
6078                              <1>
6079                              <1> ;-----      TEST FOR SPECIAL CHARACTER'S
6080                              <1>
6081 00001E50 3C08                <1>       CMP    AL, 08H                  ; IS IT A BACKSPACE
6082 00001E52 740C                <1>       JE     short P51           ; BACK_SPACE
6083 00001E54 3C0D                <1>       CMP    AL, 0Dh ; CR        ; IS IT CARRIAGE RETURN
6084 00001E56 7408                <1>       JE     short P51           ; CAR_RET
6085 00001E58 3C0A                <1>       CMP    AL, 0Ah ; LF        ; IS IT A LINE FEED
6086 00001E5A 7404                <1>       JE     short P51           ; LINE_FEED
6087 00001E5C 3C07                <1>       CMP    AL, 07h                  ; IS IT A BELL
6088 00001E5E 7515                <1>       JNE    short P52           ; IF NOT THEN DO WRITE CHARACTER
6089                              <1> P51:
6090                              <1>       ;MOV   AH,0EH              ; TTY_CHARACTER_WRITE
6091                              <1>       ;INT   10H                 ; WRITE TTY CHARACTER TO THE CRT
6092                              <1>
6093 00001E60 E860FEFFFF          <1>       call   _write_tty_m3
6094                              <1>
6095 00001E65 5A                  <1>       pop    edx ; *
6096 00001E66 5E                  <1>       pop    esi ; **
6097                              <1>
6098 00001E67 668B96[3E520100]    <1>       MOV    DX, [eSI+CURSOR_POSN]     ; GET CURRENT CURSOR POSITION
6099 00001E6E EB46                <1>       JMP    SHORT P54           ; SET CURSOR POSITION AND CONTINUE
6100                              <1> P55:
6101 00001E70 E9DFF6FFFF          <1>       JMP    VIDEO_RETURN
6102                              <1> P52:
6103 00001E75 66B90100            <1>       MOV    CX, 1               ; SET CHARACTER WRITE AMOUNT TO ONE
6104 00001E79 803D[B85F0100]02    <1>       CMP    byte [w_str_cmd], 2 ; IS THE ATTRIBUTE IN THE STRING
6105 00001E80 7204                <1>       JB     short P53           ; IF NOT THEN SKIP
6106 00001E82 8A5D00              <1>       MOV    BL, [eBP]           ; ELSE GET NEW ATTRIBUTE
6107 00001E85 45                  <1>       INC    eBP                 ; BUMP STRING POINTER
6108                              <1> P53:
6109                              <1>       ;MOV   AH,09H              ; GOT_CHARACTER
6110                              <1>       ;INT   10H                 ; WRITE CHARACTER TO THE CRT
6111                              <1>
6112 00001E86 E8AEFDFFFF          <1>       call   _write_c_current
6113                              <1>
6114 00001E8B 5A                  <1>       pop    edx ; *
6115                              <1>
6116 00001E8C 0FB6F7              <1>       movzx  esi, bh ; video page number (0 to 7)
6117 00001E8F 889E[CB5E0000]      <1>       mov    [esi+chr_attrib], bl ; color/attribute
6118                              <1>
6119 00001E95 FEC2                <1>       INC    DL                  ; INCREMENT COLUMN COUNTER
6120 00001E97 3A15[C45E0000]      <1>       CMP    DL, [CRT_COLS]           ; IF COLS ARE WITHIN RANGE FOR THIS MODE
6121 00001E9D 7217                <1>       JB     short P54           ;    THEN GO TO COLUMNS SET
6122 00001E9F FEC6                <1>       INC    DH                  ; BUMP ROW COUNTER BY ONE
6123 00001EA1 28D2                <1>       SUB    DL, DL              ; SET COLUMN COUNTER TO ZERO
6124 00001EA3 80FE19              <1>       CMP    DH, 25              ; IF ROWS ARE LESS THAN 25 THEN
6125 00001EA6 720E                <1>       JB     short P54           ; GO TO ROWS_COLUMNS_SET
6126                              <1>
6127 00001EA8 66B80A0E            <1>       MOV    AX,0E0AH            ; ELSE SCROLL SCREEN
6128                              <1>       ;INT   10H                 ; RESET ROW COUNTER TO 24
6129                              <1>
6130 00001EAC E814FEFFFF          <1>       call   _write_tty_m3
6131                              <1>
6132 00001EB1 66BA0018            <1>       mov    dx, 1800h           ; Column = 0, Row = 24
6133 00001EB5 5E                  <1>       pop    esi ; **
6134                              <1> P54:
6135                              <1>                                   ; ROW_COLUMNS_SET
6136                              <1>       ;MOV   AX,0200H            ; SET NEW CURSOR POSITION COMMAND
6137                              <1>       ;INT   10H                 ; ESTABLISH NEW CURSOR POSITION
6138                              <1>
6139 00001EB6 59                  <1>       pop    ecx ; ***
6140 00001EB7 5B                  <1>       pop    ebx ; ****
6141                              <1>
6142                              <1>       ;LOOP  P50                 ; DO IT ONCE MORE UNTIL (CX) = ZERO
6143 00001EB8 6649                <1>       dec    cx
6144 00001EBA 7587                <1>       jnz    short P50next
6145                              <1>
6146 00001EBC 665A                <1>       POP    DX  ; *****         ; RESTORE OLD CURSOR COORDINATES
6147                              <1>
6148 00001EBE F605[B85F0100]01    <1>       test   byte [w_str_cmd], 1 ; IF CURSOR WAS NOT TO BE MOVED
6149 00001EC5 0F8589F6FFFF        <1>       JNZ    VIDEO_RETURN        ; THEN EXIT WITHOUT RESETTING OLD VALUE
6150                              <1>
6151                              <1>       ;MOV   AX,0200H            ; ELSE RESTORE OLD CURSOR POSITION
6152                              <1>       ;INT   10H
6153                              <1>                                   ; DONE - EXIT WRITE STRING
6154 00001ECB E878FEFFFF          <1>       call   _set_cpos
6155 00001ED0 E97FF6FFFF          <1>       JMP    VIDEO_RETURN        ; RETURN TO CALLER
6156                              <1>
6157                              <1> vga_write_string:
6158                              <1>       ; 12/09/2016 - TRDOS 386 (TRDOS v2.0)
6159                              <1>       ;
6160                              <1>       ; derived from 'Plex86/Bochs VGABios' source code
6161                              <1>       ; vgabios-0.7a (2011)
6162                              <1>       ; by the LGPL VGABios developers Team (2001-2008)
6163                              <1>       ; 'vgabios.c', ' biosfn_write_string'
6164                              <1>
6165                              <1>       ; INPUT                                                 :
6166                              <1>       ;      (AL) = WRITE STRING COMMAND  0 - 3                       :
6167                              <1>       ;      (BH) = DISPLAY PAGE (ACTIVE PAGE)                        :
6168                              <1>       ;      (CX) = COUNT OF CHARACTERS TO WRITE, IF (CX) = 0 THEN RETURN       :
6169                              <1>       ;      (DX) = CURSOR POSITION FOR START OF STRING WRITE          :
6170                              <1>       ;      (BL) = ATTRIBUTE OF CHARACTER TO WRITE IF (AL) = 0  OR     (AL) = 1  :
6171                              <1>       ;      (eBP) = SOURCE STRING OFFSET                              :
6172                              <1>       ; OUTPUT                                                :
6173                              <1>       ;      NONE                                              :
6174                              <1>       ;-------------------------------------------------------------------------;
6175                              <1>
6176                              <1>       ; AL = 00h: Assign all characters the attribute in BL; do not update cursor
6177                              <1>       ; AL = 01h: Assign all characters the attribute in BL; update cursor
6178                              <1>       ; AL = 02h: Use attributes in string; do not update cursor
```

```
6179                                 <1>       ; AL = 03h: Use attributes in string; update cursor
6180                                 <1>
6181                                 <1>       ; biosfn_write_string(GET_AL(),GET_BH(),GET_BL(),CX,GET_DH(),GET_DL(),ES,BP);
6182                                 <1>       ; static void biosfn_write_string (flag,page,attr,count,row,col,seg,offset)
6183                                 <1>
6184                                 <1>       ; // Read curs info for the page
6185                                 <1>       ; biosfn_get_cursor_pos(page,&dummy,&oldcurs);
6186                                 <1>       ; bh = video page = 0
6187                                 <1>       ;movzx esi, word [CURSOR_POSN] ; current cursor position for video page 0
6188                                 <1>
6189                                 <1>       ; // if row=0xff special case : use current cursor position
6190                                 <1>       ; if(row==0xff)
6191                                 <1>       ;  {col=oldcurs&0x00ff;
6192                                 <1>       ;   row=(oldcurs&0xff00)>>8;
6193                                 <1>       ;  }
6194                                 <1>
6195                                 <1>       ;mov   al, [w_str_cmd]
6196                                 <1>
6197 00001ED5 80FEFF                <1>       cmp    dh, 0FFh
6198 00001ED8 7407                  <1>       je     short vga_wstr_1 ; user current cursor position
6199                                 <1> vga_wstr_0:
6200                                 <1>       ; set cursor position
6201 00001EDA 668915[3E520100]      <1>       mov    [CURSOR_POSN], dx ; save cursor pos for pg 0
6202                                 <1> vga_wstr_1:
6203 00001EE1 66FF35[3E520100]      <1>       push   word [CURSOR_POSN] ; *
6204                                 <1>
6205                                 <1>       ; ebp = string offset in system buffer (user buffer was copied to)
6206                                 <1>
6207                                 <1>       ; while(count--!=0)
6208                                 <1>       ;  {
6209                                 <1>       ;   car=read_byte(seg,offset++);
6210                                 <1>       ;   if((flag&0x02)!=0)
6211                                 <1>       ;    attr=read_byte(seg,offset++);
6212                                 <1>       ;    biosfn_write_teletype(car,page,attr,WITH_ATTR);
6213                                 <1>       ;  }
6214                                 <1>
6215                                 <1>       ;push  eax ; **
6216                                 <1>       ;test  al, 2
6217 00001EE8 F605[B85F0100]02      <1>       test   byte [w_str_cmd], 2
6218 00001EEF 751D                  <1>       jnz    short vga_wstr_3
6219 00001EF1 881D[4F520100]        <1>       mov    [ccolor], bl
6220                                 <1> vga_wstr_2:
6221 00001EF7 51                    <1>       push   ecx
6222 00001EF8 8A4500                <1>       mov    al, [ebp]
6223 00001EFB E8CC0A0000            <1>       call   vga_write_teletype
6224 00001F00 59                    <1>       pop    ecx
6225 00001F01 6649                  <1>       dec    cx
6226 00001F03 741E                  <1>       jz     short vga_wstr_4
6227 00001F05 45                    <1>       inc    ebp
6228 00001F06 8A1D[4F520100]        <1>       mov    bl, [ccolor]
6229 00001F0C EBE9                  <1>       jmp    short vga_wstr_2
6230                                 <1> vga_wstr_3:
6231 00001F0E 51                    <1>       push   ecx
6232 00001F0F 8A4500                <1>       mov    al, [ebp]
6233 00001F12 45                    <1>       inc    ebp
6234 00001F13 8A5D00                <1>       mov    bl, [ebp]
6235 00001F16 E8B10A0000            <1>       call   vga_write_teletype
6236 00001F1B 59                    <1>       pop    ecx
6237 00001F1C 6649                  <1>       dec    cx
6238 00001F1E 7403                  <1>       jz     short vga_wstr_4
6239 00001F20 45                    <1>       inc    ebp
6240 00001F21 EBEB                  <1>       jmp    short vga_wstr_3
6241                                 <1> vga_wstr_4:
6242                                 <1>       ; // Set back curs pos
6243                                 <1>       ; if((flag&0x01)==0)
6244                                 <1>       ;  biosfn_set_cursor_pos(page,oldcurs);
6245                                 <1>       ; }
6246                                 <1>       ;pop   eax ; **
6247 00001F23 665A                  <1>       pop    dx ; word [CURSOR_POSN] ; *
6248                                 <1>       ;test  al, 1
6249 00001F25 F605[B85F0100]01      <1>       test   byte [w_str_cmd], 1
6250 00001F2C 0F8522F6FFFF          <1>       jnz    VIDEO_RETURN
6251 00001F32 668915[3E520100]      <1>       mov    [CURSOR_POSN], dx
6252 00001F39 E916F6FFFF            <1>       JMP    VIDEO_RETURN
6253                                 <1>
6254                                 <1> ; 07/07/2016
6255                                 <1> ; 27/06/2016 - TRDOS 386 (TRDOS v2.0)
6256                                 <1> ; VIDEO1.ASM - 24/03/1985 (IBM PC-AT BIOS source code)
6257                                 <1> ;--------------------------------------------------
6258                                 <1> ;  SCROLL UP
6259                                 <1> ;   THIS ROUTINE SCROLLS UP THE INFORMATION ON THE CRT
6260                                 <1> ; ENTRY ---
6261                                 <1> ;  CH,CL = UPPER LEFT CORNER OF REGION TO SCROLL
6262                                 <1> ;  DH,DL = LOWER RIGHT CORNER OF REGION TO SCROLL
6263                                 <1> ;   BOTH OF THE ABOVE ARE IN CHARACTER POSITIONS
6264                                 <1> ;  BH = FILL VALUE FOR BLANKED LINES
6265                                 <1> ;  AL = # LINES TO SCROLL (AL=0 MEANS BLANK THE ENTIRE FIELD)
6266                                 <1> ;  DS = DATA SEGMENT
6267                                 <1> ;  ES = REGEN SEGMENT
6268                                 <1> ; EXIT --
6269                                 <1> ;  NOTHING, THE SCREEN IS SCROLLED
6270                                 <1> ;--------------------------------------------------
6271                                 <1>
6272                                 <1>       ; cl = upper left column
6273                                 <1>       ; ch = upper left row
6274                                 <1>       ; dl = lower rigth column
6275                                 <1>       ; dh = lower right row
6276                                 <1>       ;
6277                                 <1>       ; al = line count (AL=0 means blank entire fields)
6278                                 <1>       ; bl = fill value for blanked lines
6279                                 <1>       ; bh = unused
6280                                 <1>
6281                                 <1> GRAPHICS_UP:
```

```
                        <1>       ; 07/07/2016
                        <1>       ;AH = Current video mode, [CRT_MODE]
00001F3E 80FC07         <1>       cmp   ah, 7
00001F41 7766           <1>       ja    short vga_graphics_up
                        <1>       ;je   n0
                        <1>
00001F43 88C7           <1>       MOV   bh, al            ; save line count in BH
00001F45 6689C8         <1>       MOV   AX, CX            ; GET UPPER LEFT POSITION INTO AX REG
                        <1>
                        <1> ;-----      USE CHARACTER SUBROUTINE FOR POSITIONING
                        <1> ;-----      ADDRESS RETURNED IS MULTIPLIED BY 2 FROM CORRECT VALUE
                        <1>
00001F48 E8D9050000     <1>       CALL  GRAPH_POSN
00001F4D 0FB7F8         <1>       MOVzx eDI, AX                 ; SAVE RESULT AS DESTINATION ADDRESS
                        <1>
                        <1> ;-----      DETERMINE SIZE OF WINDOW
                        <1>
00001F50 6629CA         <1>       SUB   DX, CX
00001F53 6681C20101     <1>         ADD     DX, 101h              ; ADJUST VALUES
00001F58 C0E602         <1>       SAL   DH, 2             ; MULTIPLY ROWS BY 4 AT 8 VERT DOTS/CHAR
                        <1>                               ; AND EVEN/ODD ROWS
                        <1> ;-----      DETERMINE CRT MODE
                        <1>
00001F5B 803D[C25E0000]06 <1>       CMP   byte [CRT_MODE], 6  ; TEST FOR MEDIUM RES
00001F62 7305           <1>         JNC     short _R7_            ; FIND_SOURCE
                        <1>
                        <1> ;-----      MEDIUM RES UP
00001F64 D0E2           <1>       SAL   DL, 1             ; # COLUMNS * 2, SINCE 2 BYTES/CHAR
00001F66 66D1E7         <1>       SAL   DI, 1             ; OFFSET *2 SINCE 2 BYTES/CHAR
                        <1>
                        <1> ;-----      DETERMINE THE SOURCE ADDRESS IN THE BUFFER
                        <1> _R7_:                               ; FIND_SOURCE
00001F69 81C700800B00   <1>       add   edi, 0B8000h
00001F6F C0E702         <1>       sal   bh, 2             ; multiply number of lines by 4
00001F72 7431           <1>         JZ      short _R11            ; IF ZERO, THEN BLANK ENTIRE FIELD
00001F74 B050           <1>       MOV   AL, 80            ; 80 BYTES/ROW
00001F76 F6E7           <1>       mul   bh                ; determine offset to source
00001F78 0FB7F0         <1>       movzx esi, ax                 ; offset to source
00001F7B 01FE           <1>       add   eSI, eDI          ; SET UP SOURCE
00001F7D 88F4           <1>       MOV   AH, DH            ; NUMBER OF ROWS IN FIELD
00001F7F 28FC           <1>       sub   ah, bh            ; determine number to move
                        <1>
                        <1> ;-----      LOOP THROUGH, MOVING ONE ROW AT A TIME, BOTH EVEN AND ODD FIELDS
                        <1> _R8:                                ; ROW_LOOP
00001F81 E812040000     <1>         CALL    _R17                  ; MOVE ONE ROW
00001F86 6681EEB01F     <1>       SUB   SI, 2000h-80      ; MOVE TO NEXT ROW
00001F8B 6681EFB01F     <1>       SUB   DI, 2000h-80
00001F90 FECC           <1>       DEC   AH                ; NUMBER OF ROWS TO MOVE
00001F92 75ED           <1>         JNZ     short _R8             ; CONTINUE TILL ALL MOVED
                        <1>
                        <1> ;-----      FILL IN THE VACATED LINE(S)
                        <1> _R9:                                ; CLEAR ENTRY
00001F94 88D8           <1>       mov   al, bl            ; attribute to fill with
                        <1> _R10_:
00001F96 E819040000     <1>         CALL    _R18                  ; CLEAR THAT ROW
00001F9B 6681EFB01F     <1>       SUB   DI, 2000h-80      ; POINT TO NEXT LINE
00001FA0 FECF           <1>       dec   bh                ; number of lines to fill
00001FA2 75F2           <1>         JNZ     short _R10_           ; CLEAR LOOP
00001FA4 C3             <1>       retn                    ; EVERYYHING DONE
                        <1>
                        <1> _R11:                               ; BLANK_FIELD
00001FA5 88F7           <1>       mov   bh, dh            ; set blank count to everything in field
00001FA7 EBEB           <1>         JMP     short _R9             ; CLEAR THE FIELD
                        <1>
                        <1> vga_graphics_up:
                        <1>       ; 08/08/2016
                        <1>       ; 07/08/2016
                        <1>       ; 04/08/2016
                        <1>       ; 01/08/2016
                        <1>       ; 31/07/2016
                        <1>       ; 07/07/2016 - TRDOS 386 (TRDOS v2.0)
                        <1>       ;
                        <1>       ; derived from 'Plex86/Bochs VGABios' source code
                        <1>       ; vgabios-0.7a (2011)
                        <1>       ; by the LGPL VGABios developers Team (2001-2008)
                        <1>       ; 'vgabios.c', 'biosfn_scroll'
                        <1>       ;
                        <1>
                        <1>       ; cl = upper left column
                        <1>       ; ch = upper left row
                        <1>       ; dl = lower rigth column
                        <1>       ; dh = lower right row
                        <1>       ;
                        <1>       ; al = line count (AL=0 means blank entire fields)
                        <1>       ; bl = fill value for blanked lines
                        <1>       ; bh = unused
                        <1>       ;
                        <1>       ; ah = [CRT_MODE], current video mode
                        <1>
00001FA9 88C7           <1>       mov   bh, al ; 31/07/2016
00001FAB BE[E65E0000]   <1>       mov   esi, vga_g_modes
00001FB0 89F7           <1>       mov   edi, esi
00001FB2 83C708         <1>       add   edi, vga_g_mode_count
                        <1> vga_g_up_0:
00001FB5 AC             <1>       lodsb
00001FB6 38E0           <1>       cmp   al, ah ; [CRT_MODE]
00001FB8 7405           <1>       je    short vga_g_up_1
00001FBA 39FE           <1>       cmp   esi, edi
00001FBC 72F7           <1>       jb    short vga_g_up_0
                        <1>       ;xor  bh, bh ; 31/07/2016)
00001FBE C3             <1>       retn  ; nothing to do
                        <1> vga_g_up_1:
00001FBF 88F8           <1>       mov   al, bh ; 31/07/2016
```

```
6385 00001FC1 83C64F              <1>       add    esi, vga_g_memmodel - (vga_g_modes + 1)
6386                              <1>       ; [ESI] = VGA memory model number (LINEAR8, PLANAR4, PLANAR1)
6387                              <1>
6388                              <1>       ; if(rlr>=nbrows)rlr=nbrows-1;
6389                              <1>       ; if(clr>=nbcols)clr=nbcols-1;
6390                              <1>       ; if(nblines>nbrows)nblines=0;
6391                              <1>       ; cols=clr-cul+1;
6392                              <1>
6393 00001FC4 3A35[CA5E0000]      <1>       cmp    dh, [VGA_ROWS]
6394 00001FCA 7208                <1>       jb     short vga_g_up_2
6395 00001FCC 8A35[CA5E0000]      <1>       mov    dh, [VGA_ROWS]
6396 00001FD2 FECE                <1>       dec    dh
6397                              <1> vga_g_up_2:
6398 00001FD4 3A15[C45E0000]      <1>       cmp    dl, [CRT_COLS]  ; = [VGA_COLS]
6399 00001FDA 7208                <1>       jb     short vga_g_up_3
6400 00001FDC 8A15[C45E0000]      <1>       mov    dl, [CRT_COLS]
6401 00001FE2 FECA                <1>       dec    dl
6402                              <1> vga_g_up_3:
6403 00001FE4 3A05[CA5E0000]      <1>       cmp    al, [VGA_ROWS]
6404 00001FEA 7602                <1>       jna    short vga_g_up_4
6405 00001FEC 28C0                <1>       sub    al, al ; 0
6406                              <1> vga_g_up_4:
6407 00001FEE 88D7                <1>       mov    bh, dl ; clr
6408 00001FF0 28CF                <1>       sub    bh, cl ; cul
6409 00001FF2 FEC7                <1>       inc    bh ; cols = clr-cul+1
6410                              <1>
6411 00001FF4 20C0                <1>       and    al, al ; nblines = 0
6412 00001FF6 755D                <1>       jnz    short vga_g_up_6
6413 00001FF8 20ED                <1>       and    ch, ch ; rul = 0
6414 00001FFA 7559                <1>       jnz    short vga_g_up_6
6415 00001FFC 20C9                <1>       and    cl, cl ; cul = 0
6416 00001FFE 7555                <1>       jnz    short vga_g_up_6
6417                              <1>
6418 00002000 6650                <1>       push   ax
6419 00002002 A0[CA5E0000]        <1>       mov    al, [VGA_ROWS]
6420 00002007 FEC8                <1>       dec    al
6421 00002009 38C6                <1>       cmp    dh, al ; rlr = nbrows-1
6422 0000200B 7546                <1>       jne    short vga_g_up_5
6423 0000200D A0[C45E0000]        <1>         mov     al, [CRT_COLS]  ; = VGA_COLS
6424 00002012 FEC8                <1>       dec    al
6425 00002014 38C2                <1>       cmp    dl, al ; clr = nbcols-1
6426 00002016 753B                <1>       jne    short vga_g_up_5
6427 00002018 6658                <1>       pop    ax
6428                              <1>
6429 0000201A 66B80502            <1>       mov    ax, 0205h
6430 0000201E 66BACE03            <1>       mov    dx, 3CEh ; VGAREG_GRDC_ADDRESS
6431 00002022 66EF                <1>       out    dx, ax
6432 00002024 A0[CA5E0000]        <1>       mov    al, [VGA_ROWS]
6433 00002029 8A25[C45E0000]      <1>       mov    ah, [CRT_COLS] ; = [VGA_COLS]
6434 0000202F F6E4                <1>       mul    ah
6435 00002031 0FB7D0              <1>       movzx  edx, ax
6436                              <1>       ; 08/08/2016
6437 00002034 0FB605[C65E0000]    <1>       movzx  eax, byte [CHAR_HEIGHT]
6438 0000203B F7E2                <1>       mul    edx
6439                              <1>       ; eax = byte count
6440 0000203D 89C1                <1>       mov    ecx, eax
6441                              <1>       ;; 07/08/2016
6442                              <1>       ;shl   dx, 3 ; * 8 ; * [CHAR_HEIGHT]
6443                              <1>       ;mov   ecx, edx
6444 0000203F 88D8                <1>       mov    al, bl ; fill value for blanked lines
6445 00002041 BF00000A00          <1>       mov    edi, 0A0000h
6446 00002046 F3AA                <1>       rep    stosb
6447                              <1>
6448 00002048 66B80500            <1>       mov    ax, 5
6449 0000204C 66BACE03            <1>       mov    dx, 3CEh ; VGAREG_GRDC_ADDRESS
6450 00002050 66EF                <1>       out    dx, ax ; 0005h
6451                              <1>
6452 00002052 C3                  <1>       retn
6453                              <1>
6454                              <1> vga_g_up_5:
6455 00002053 6658                <1>       pop    ax
6456                              <1>
6457                              <1> vga_g_up_6:
6458                              <1>       ; [ESI] = VGA memory model number for current video mode
6459                              <1>         ;
6460                              <1>         ; LINEAR8 equ 5
6461                              <1>         ; PLANAR4 equ 4
6462                              <1>         ; PLANAR1 equ 3
6463                              <1>
6464 00002055 803E04              <1>       cmp    byte [esi], PLANAR4
6465 00002058 7424                <1>       je     short vga_g_up_planar
6466 0000205A 803E03              <1>       cmp    byte [esi], PLANAR1
6467 0000205D 741F                <1>       je     short vga_g_up_planar
6468                              <1> vga_g_up_linear8:
6469                              <1>       ; 07/07/2016 (TEMPORARY)
6470                              <1>       ;
6471                              <1>       ; cl = upper left column ; cul
6472                              <1>       ; ch = upper left row ; rul
6473                              <1>       ; dl = lower rigth column ; clr
6474                              <1>       ; dh = lower right row ; rlr
6475                              <1>
6476                              <1> vga_g_up_l0:
6477                              <1>       ;{for(i=rul;i<=rlr;i++)
6478                              <1>       ; if((i+nblines>rlr)||(nblines==0))
6479 0000205F 08C0                <1>       or     al, al
6480 00002061 7414                <1>       jz     short vga_g_up_l2
6481 00002063 88C4                <1>       mov    ah, al
6482 00002065 00EC                <1>       add    ah, ch ; i+nblines
6483                              <1>       ;jc    short vga_g_up_l2
6484 00002067 38F4                <1>       cmp    ah, dh
6485 00002069 770C                <1>       ja     short vga_g_up_l2
6486                              <1>       ; else
6487                              <1>       ;  vgamem_copy_pl4(cul,i+nblines,i,cols,nbcols,cheight);
```

```
6488 0000206B E8F2000000          <1>       call   vgamem_copy_l8
6489                              <1> vga_g_up_l1:
6490 00002070 FEC5                <1>       inc    ch
6491 00002072 38F5                <1>       cmp    ch, dh
6492 00002074 76E9                <1>       jna    short vga_g_up_l0
6493 00002076 C3                  <1>       retn
6494                              <1> vga_g_up_l2:
6495                              <1>       ; vgamem_fill_pl4(cul,i,cols,nbcols,cheight,attr);
6496 00002077 E850010000          <1>       call   vgamem_fill_l8
6497 0000207C EBF2                <1>       jmp    short vga_g_up_l1
6498                              <1>
6499                              <1> vga_g_up_planar:
6500                              <1>       ; cl = upper left column ; cul
6501                              <1>       ; ch = upper left row ; rul
6502                              <1>       ; dl = lower rigth column ; clr
6503                              <1>       ; dh = lower right row ; rlr
6504                              <1> vga_g_up_pl0:
6505                              <1>       ;{for(i=rul;i<=rlr;i++)
6506                              <1>       ; if((i+nblines>rlr)||(nblines==0))
6507 0000207E 20C0                <1>       and    al, al
6508 00002080 7414                <1>       jz     short vga_g_up_pl2
6509 00002082 88C4                <1>       mov    ah, al
6510 00002084 00EC                <1>       add    ah, ch ; i+nblines
6511                              <1>       ;jc    short vga_g_up_pl2
6512 00002086 38F4                <1>       cmp    ah, dh
6513 00002088 770C                <1>       ja     short vga_g_up_pl2
6514                              <1>       ; else
6515                              <1>       ;  vgamem_copy_pl4(cul,i+nblines,i,cols,nbcols,cheight);
6516 0000208A E80E000000          <1>       call   vgamem_copy_pl4
6517                              <1> vga_g_up_pl1:
6518 0000208F FEC5                <1>       inc    ch
6519 00002091 38F5                <1>       cmp    ch, dh
6520 00002093 76E9                <1>       jna    short vga_g_up_pl0
6521 00002095 C3                  <1>       retn
6522                              <1> vga_g_up_pl2:
6523                              <1>       ; vgamem_fill_pl4(cul,i,cols,nbcols,cheight,attr);
6524 00002096 E870000000          <1>       call   vgamem_fill_pl4
6525 0000209B EBF2                <1>       jmp    short vga_g_up_pl1
6526                              <1>
6527                              <1> vgamem_copy_pl4:
6528                              <1>       ; 08/08/2016
6529                              <1>       ; 07/08/2016
6530                              <1>       ; 07/07/2016 - TRDOS 386 (TRDOS v2.0)
6531                              <1>       ;
6532                              <1>       ; derived from 'Plex86/Bochs VGABios' source code
6533                              <1>       ; vgabios-0.7a (2011)
6534                              <1>       ; by the LGPL VGABios developers Team (2001-2008)
6535                              <1>       ; 'vgabios.c', 'vgamem_copy_pl4'
6536                              <1>       ;
6537                              <1>       ; vgamem_copy_pl4(xstart,ysrc,ydest,cols,nbcols,cheight)
6538                              <1>       ; cl = xstart, ah = ysrc (i+nblines), ch = ydest (i),
6539                              <1>       ; bh = cols, [CRT_COLS] = nbcols, [CHAR_HEIGHT] = cheight
6540                              <1>
6541                              <1>       ; src=ysrc*cheight*nbcols+xstart;
6542                              <1>       ; dest=ydest*cheight*nbcols+xstart;
6543                              <1>
6544 0000209D 52                  <1>       push   edx
6545 0000209E 50                  <1>       push   eax
6546                              <1>
6547                              <1>       ; outw(VGAREG_GRDC_ADDRESS, 0x0105)
6548 0000209F 66B80501            <1>       mov    ax, 0105h
6549 000020A3 66BACE03            <1>       mov    dx, 3CEh ; VGAREG_GRDC_ADDRESS
6550 000020A7 66EF                <1>       out    dx, ax
6551                              <1>
6552                              <1>       ; 07/08/2016
6553                              <1>       ;mov     ah, [esp+1]
6554                              <1>       ;movzx edx, ah ; ysrc
6555 000020A9 0FB6542401          <1>       movzx  edx, byte [esp+1]
6556                              <1>       ; 08/08/2016
6557 000020AE 0FB605[C65E0000]    <1>       movzx  eax, byte [CHAR_HEIGHT]
6558 000020B5 8A25[C45E0000]      <1>       mov    ah, [CRT_COLS] ; nbcols
6559 000020BB F6E4                <1>       mul    ah
6560                              <1>       ;; 07/08/2016
6561                              <1>       ;movzx eax, byte [CRT_COLS]
6562                              <1>       ;shl   ax, 3 ; * 8 ; * [CHAR_HEIGHT]
6563 000020BD 50                  <1>       push   eax ; cheight * nbcols
6564 000020BE F7E2                <1>       mul    edx ; * ysrc
6565                              <1>       ; eax = ysrc * cheight * nbcols
6566                              <1>       ; edx = 0
6567 000020C0 88CA                <1>       mov    dl, cl ; edx = xstart
6568 000020C2 01D0                <1>       add    eax, edx
6569 000020C4 89C6                <1>       mov    esi, eax ; src
6570 000020C6 88EA                <1>       mov    dl, ch ; ydest
6571 000020C8 58                  <1>       pop    eax ; cheight * nbcols
6572 000020C9 F7E2                <1>       mul    edx
6573                              <1>       ; eax = ydest * cheight * nbcols
6574 000020CB 88CA                <1>       mov    dl, cl ; edx = xstart
6575 000020CD 01D0                <1>       add    eax, edx
6576 000020CF 89C7                <1>       mov    edi, eax ; dest
6577                              <1>       ; esi = src
6578                              <1>       ; edi = dest
6579                              <1>       ; for(i=0;i<cheight;i++)
6580                              <1>       ; {
6581                              <1>       ;  memcpyb(0xa000,dest+i*nbcols,0xa000,src+i*nbcols,cols);
6582                              <1>       ; }
6583 000020D1 51                  <1>       push   ecx
6584 000020D2 B900000A00          <1>       mov    ecx, 0A0000h
6585 000020D7 01CE                <1>       add    esi, ecx
6586 000020D9 01CF                <1>       add    edi, ecx
6587                              <1>       ; 08/08/2016
6588 000020DB 8A35[C65E0000]      <1>       mov    dh, [CHAR_HEIGHT]
6589                              <1>       ;; 07/08/2016
6590                              <1>       ;mov   dh, 8 ; 07/08/2016
```

```
6591 000020E1 28D2                <1>       sub    dl, dl ; i
6592                              <1> vgamem_copy_pl4_0:
6593 000020E3 56                  <1>       push   esi
6594 000020E4 57                  <1>       push   edi
6595 000020E5 0FB605[C45E0000]    <1>       movzx  eax, byte [CRT_COLS]
6596 000020EC F6E2                <1>       mul    dl
6597                              <1>       ; eax = i * nbcols
6598 000020EE 01C7                <1>       add    edi, eax ; dest+i*nbcols
6599 000020F0 01C6                <1>       add    esi, eax
6600 000020F2 0FB6CF              <1>       movzx  ecx, bh ; cols
6601 000020F5 F3A4                <1>       rep    movsb
6602 000020F7 5F                  <1>       pop    edi
6603 000020F8 5E                  <1>       pop    esi
6604 000020F9 FECE                <1>       dec    dh
6605 000020FB 75E6                <1>       jnz    short vgamem_copy_pl4_0
6606                              <1> vgamem_copy_pl4_1:
6607 000020FD 59                  <1>       pop    ecx
6608                              <1>
6609                              <1>       ; outw(VGAREG_GRDC_ADDRESS, 0x0005);
6610 000020FE 66B80500            <1>       mov    ax, 0005h
6611 00002102 66BACE03            <1>       mov    dx, 3CEh ; VGAREG_GRDC_ADDRESS
6612 00002106 66EF                <1>       out    dx, ax
6613                              <1>
6614 00002108 58                  <1>       pop    eax
6615 00002109 5A                  <1>       pop    edx
6616                              <1>
6617 0000210A C3                  <1>       retn
6618                              <1>
6619                              <1> vgamem_fill_pl4:
6620                              <1>       ; 08/08/2016
6621                              <1>       ; 07/08/2016
6622                              <1>       ; 04/08/2016
6623                              <1>       ; 07/07/2016 - TRDOS 386 (TRDOS v2.0)
6624                              <1>       ;
6625                              <1>       ; derived from 'Plex86/Bochs VGABios' source code
6626                              <1>       ; vgabios-0.7a (2011)
6627                              <1>       ; by the LGPL VGABios developers Team (2001-2008)
6628                              <1>       ; 'vgabios.c', 'vgamem_fill_pl4'
6629                              <1>       ;
6630                              <1>       ; vgamem_fill_pl4(xstart,ystart,cols,nbcols,cheight,attr)
6631                              <1>       ; cl = xstart, edi = ch = ystart, bh = cols,
6632                              <1>       ; [CRT_COLS] = nbcols, [CHAR_HEIGHT] = cheight, attr = 0
6633                              <1>
6634                              <1>       ; dest=ystart*cheight*nbcols+xstart;
6635 0000210B 52                  <1>       push   edx
6636 0000210C 50                  <1>       push   eax
6637                              <1>
6638                              <1>       ; outw(VGAREG_GRDC_ADDRESS, 0x0205)
6639 0000210D 66B80502            <1>       mov    ax, 0205h
6640 00002111 66BACE03            <1>       mov    dx, 3CEh ; VGAREG_GRDC_ADDRESS
6641 00002115 66EF                <1>       out    dx, ax
6642                              <1>
6643                              <1>              ; 08/08/2016
6644 00002117 0FB605[C65E0000]    <1>       movzx   eax, byte [CHAR_HEIGHT]
6645 0000211E F6E5                <1>       mul    ch
6646                              <1>       ;; 07/08/2016
6647                              <1>       ;movzx eax, ch
6648                              <1>       ;shl   ax, 3 ; * 8 ; * [CHAR_HEIGHT]
6649 00002120 0FB615[C45E0000]    <1>       movzx  edx, byte [CRT_COLS] ; = [VGA_COLS]
6650 00002127 F7E2                <1>       mul    edx
6651                              <1>       ; edx  = 0
6652 00002129 88CA                <1>       mov    dl, cl
6653 0000212B 01D0                <1>       add    eax, edx
6654 0000212D 89C7                <1>       mov    edi, eax
6655                              <1>       ; edi = dest
6656                              <1>       ; for(i=0;i<cheight;i++)
6657                              <1>       ; {
6658                              <1>       ;  memsetb(0xa000,dest+i*nbcols,attr,cols);
6659                              <1>       ; }
6660 0000212F 81C700000A00        <1>       add    edi, 0A0000h
6661 00002135 51                  <1>       push   ecx
6662                              <1>       ; 08/08/2016
6663 00002136 8A35[C65E0000]      <1>       mov    dh, [CHAR_HEIGHT]
6664                              <1>       ;; 07/08/2016
6665                              <1>       ;mov   dh, 8 ; 07/08/2016
6666 0000213C 28D2                <1>       sub    dl, dl ; i
6667                              <1> vgamem_fill_pl4_0:
6668 0000213E 57                  <1>       push   edi
6669 0000213F 0FB605[C45E0000]    <1>       movzx  eax, byte [CRT_COLS]
6670 00002146 F6E2                <1>       mul    dl
6671                              <1>       ; eax = i * nbcols
6672 00002148 01C7                <1>       add    edi, eax ; dest+i*nbcols
6673 0000214A 88D8                <1>       mov    al, bl ; attr ; 04/08/2016
6674 0000214C 0FB6CF              <1>       movzx  ecx, bh ; cols
6675 0000214F F3AA                <1>       rep    stosb
6676 00002151 5F                  <1>       pop    edi
6677 00002152 75EA                <1>       jnz    short vgamem_fill_pl4_0
6678                              <1> vgamem_fill_pl4_1:
6679 00002154 59                  <1>       pop    ecx
6680                              <1>
6681                              <1>       ; outw(VGAREG_GRDC_ADDRESS, 0x0005);
6682 00002155 66B80500            <1>       mov    ax, 0005h
6683 00002159 66BACE03            <1>       mov    dx, 3CEh ; VGAREG_GRDC_ADDRESS
6684 0000215D 66EF                <1>       out    dx, ax
6685                              <1>
6686 0000215F 58                  <1>       pop    eax
6687 00002160 5A                  <1>       pop    edx
6688                              <1>
6689 00002161 C3                  <1>       retn
6690                              <1>
6691                              <1> vgamem_copy_l8:
6692                              <1>       ; 08/08/2016
6693                              <1>       ; 07/08/2016
```

```
                                <1>       ; 06/08/2016
                                <1>       ; 07/07/2016 - TRDOS 386 (TRDOS v2.0)
                                <1>       ;
                                <1>       ; TEMPORARY
                                <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', 'vgamem_copy_pl4'
                                <1>       ;
                                <1>       ; vgamem_copy_pl4(xstart,ysrc,ydest,cols,nbcols,cheight)
                                <1>       ; cl = xstart, ah = ysrc (i+nblines), ch = ydest (i),
                                <1>       ; bh = cols, [CRT_COLS] = nbcols, [CHAR_HEIGHT] = cheight
                                <1>
                                <1>       ; src=ysrc*cheight*nbcols+xstart;
                                <1>       ; dest=ydest*cheight*nbcols+xstart;
                                <1>
00002162 52                     <1>       push   edx
00002163 50                     <1>       push   eax
                                <1>
                                <1>       ; outw(VGAREG_GRDC_ADDRESS, 0x0105)
                                <1>       ;mov   ax, 0105h
                                <1>       ;mov   dx, 3CEh ; VGAREG_GRDC_ADDRESS
                                <1>       ;out   dx, ax
                                <1>
                                <1>       ;mov   ah, [esp+1]
                                <1>
00002164 0FB6D4                 <1>       movzx  edx, ah ; ysrc
                                <1>       ; 08/08/2016
00002167 0FB605[C65E0000]       <1>       movzx  eax, byte [CHAR_HEIGHT]
0000216E 8A25[C45E0000]         <1>       mov    ah, [CRT_COLS] ; nbcols
00002174 F6E4                   <1>       mul    ah
                                <1>       ;; 07/08/2016
                                <1>       ;movzx eax, byte [CRT_COLS]
                                <1>       ;shl   ax, 3 ; * 8 ; * [CHAR_HEIGHT]
00002176 50                     <1>       push   eax ; cheight * nbcols
00002177 F7E2                   <1>       mul    edx ; * ysrc
                                <1>       ; eax = ysrc * cheight * nbcols
                                <1>       ; edx = 0
00002179 88CA                   <1>       mov    dl, cl ; edx = xstart
0000217B 01D0                   <1>       add    eax, edx
0000217D 89C6                   <1>       mov    esi, eax ; src
0000217F 66C1E603               <1>       shl    si, 3 ; * 8 ; 06/08/2016
00002183 88EA                   <1>       mov    dl, ch ; ydest
00002185 58                     <1>       pop    eax ; cheight * nbcols
00002186 F7E2                   <1>       mul    edx
                                <1>       ; eax = ydest * cheight * nbcols
00002188 88CA                   <1>       mov    dl, cl ; edx = xstart
0000218A 01D0                   <1>       add    eax, edx
0000218C 89C7                   <1>       mov    edi, eax ; dest
0000218E 66C1E703               <1>       shl    di, 3 ; * 8 ; 06/08/2016
                                <1>       ; esi = src
                                <1>       ; edi = dest
                                <1>       ; for(i=0;i<cheight;i++)
                                <1>       ; {
                                <1>       ;  memcpyb(0xa000,dest+i*nbcols,0xa000,src+i*nbcols,cols);
                                <1>       ; }
00002192 51                     <1>       push   ecx
00002193 B900000A00             <1>       mov    ecx, 0A0000h
00002198 01CE                   <1>       add    esi, ecx
0000219A 01CF                   <1>       add    edi, ecx
                                <1>       ; 08/08/2016
0000219C 8A35[C65E0000]         <1>       mov    dh, [CHAR_HEIGHT]
                                <1>       ;; 07/08/2016
                                <1>       ;mov   dh, 8 ; 07/08/2016
000021A2 28D2                   <1>       sub    dl, dl ; i
                                <1> vgamem_copy_l8_0:
000021A4 56                     <1>       push   esi
000021A5 57                     <1>       push   edi
000021A6 0FB605[C45E0000]       <1>       movzx  eax, byte [CRT_COLS]
000021AD F6E2                   <1>       mul    dl
                                <1>       ; eax = i * nbcols
000021AF 66C1E003               <1>       shl    ax, 3 ; * 8 ; 06/08/2016
000021B3 01C7                   <1>       add    edi, eax ; dest+i*nbcols
000021B5 01C6                   <1>       add    esi, eax
000021B7 0FB6CF                 <1>       movzx  ecx, bh ; cols
000021BA 66C1E103               <1>       shl    cx, 3 ; * 8 ; 06/08/2016
000021BE F3A4                   <1>       rep    movsb
000021C0 5F                     <1>       pop    edi
000021C1 5E                     <1>       pop    esi
000021C2 FEC2                   <1>       inc    dl ; 06/08/2016
000021C4 FECE                   <1>       dec    dh
000021C6 75DC                   <1>       jnz    short vgamem_copy_l8_0
                                <1> vgamem_copy_l8_1:
000021C8 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>
000021C9 58                     <1>       pop    eax
000021CA 5A                     <1>       pop    edx
                                <1>
000021CB C3                     <1>       retn
                                <1>
                                <1> vgamem_fill_l8:
                                <1>       ; 08/08/2016
                                <1>       ; 07/08/2016
                                <1>       ; 06/08/2016
                                <1>       ; 04/08/2016
                                <1>       ; 07/07/2016 - TRDOS 386 (TRDOS v2.0)
                                <1>       ;
```

```
                                <1>       ; TEMPORARY
                                <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', 'vgamem_fill_pl4'
                                <1>       ;
                                <1>       ; vgamem_fill_pl4(xstart,ystart,cols,nbcols,cheight,attr)
                                <1>       ; cl = xstart, edi = ch = ystart, bh = cols,
                                <1>       ; [CRT_COLS] = nbcols, [CHAR_HEIGHT] = cheight, attr = 0
                                <1>
                                <1>       ; dest=ystart*cheight*nbcols+xstart;
000021CC 52                     <1>       push   edx
000021CD 50                     <1>       push   eax
                                <1>
                                <1>       ;; outw(VGAREG_GRDC_ADDRESS, 0x0205)
                                <1>       ;mov   ax, 0205h
                                <1>       ;mov   dx, 3CEh ; VGAREG_GRDC_ADDRESS
                                <1>       ;out   dx, ax
                                <1>
                                <1>         ; 08/08/2016
000021CE 0FB605[C65E0000]       <1>       movzx   eax, byte [CHAR_HEIGHT]
000021D5 F6E5                   <1>       mul    ch
                                <1>       ;; 07/08/2016
                                <1>       ;movzx eax, ch
                                <1>       ;shl   ax, 3 ; * 8 ; * [CHAR_HEIGHT]
000021D7 0FB615[C45E0000]       <1>       movzx  edx, byte [CRT_COLS] ; = [VGA_COLS]
000021DE F7E2                   <1>       mul    edx
                                <1>       ; edx  = 0
000021E0 88CA                   <1>       mov    dl, cl
000021E2 01D0                   <1>       add    eax, edx
000021E4 89C7                   <1>       mov    edi, eax
000021E6 66C1E703               <1>       shl    di, 3 ; * 8 ; 06/08/2016
                                <1>       ; edi = dest
                                <1>       ; for(i=0;i<cheight;i++)
                                <1>       ; {
                                <1>       ;  memsetb(0xa000,dest+i*nbcols,attr,cols);
                                <1>       ; }
000021EA 81C700000A00           <1>       add    edi, 0A0000h
000021F0 51                     <1>       push   ecx
                                <1>       ; 08/08/2016
000021F1 8A35[C65E0000]         <1>       mov    dh, [CHAR_HEIGHT]
                                <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
```

```
6900                                    <1> vga_g_down_0:
6901 0000222C AC                        <1>         lodsb
6902 0000222D 38E0                      <1>         cmp    al, ah ; [CRT_MODE]
6903 0000222F 7405                      <1>         je     short vga_g_down_1
6904 00002231 39FE                      <1>         cmp    esi, edi
6905 00002233 72F7                      <1>         jb     short vga_g_down_0
6906                                    <1>         ; xor  bh, bh ; 31/07/2016
6907 00002235 C3                        <1>         retn   ; nothing to do
6908                                    <1> vga_g_down_1:
6909 00002236 88F8                      <1>         mov    al, bh ; 31/07/2016
6910 00002238 83C64F                    <1>         add    esi, vga_memmodel - (vga_modes + 1)
6911                                    <1>         ; [ESI] = VGA memory model number (LINEAR8, PLANAR4, PLANAR1)
6912                                    <1>
6913                                    <1>         ; if(rlr>=nbrows)rlr=nbrows-1;
6914                                    <1>         ; if(clr>=nbcols)clr=nbcols-1;
6915                                    <1>         ; if(nblines>nbrows)nblines=0;
6916                                    <1>         ; cols=clr-cul+1;
6917                                    <1>
6918 0000223B 3A35[CA5E0000]            <1>         cmp    dh, [VGA_ROWS]
6919 00002241 7208                      <1>         jb     short vga_g_down_2
6920 00002243 8A35[CA5E0000]            <1>         mov    dh, [VGA_ROWS]
6921 00002249 FECE                      <1>         dec    dh
6922                                    <1> vga_g_down_2:
6923 0000224B 3A15[C45E0000]            <1>         cmp    dl, [CRT_COLS]  ; = [VGA_COLS]
6924 00002251 7208                      <1>         jb     short vga_g_down_3
6925 00002253 8A15[C45E0000]            <1>         mov    dl, [CRT_COLS]
6926 00002259 FECA                      <1>         dec    dl
6927                                    <1> vga_g_down_3:
6928 0000225B 3A05[CA5E0000]            <1>         cmp    al, [VGA_ROWS]
6929 00002261 7602                      <1>         jna    short vga_g_down_4
6930 00002263 28C0                      <1>         sub    al, al ; 0
6931                                    <1> vga_g_down_4:
6932 00002265 88F7                      <1>         mov    bh, dh ; clr
6933 00002267 28CF                      <1>         sub    bh, cl ; cul
6934 00002269 FEC7                      <1>         inc    bh ; cols = clr-cul+1
6935                                    <1>
6936 0000226B 20C0                      <1>         and    al, al ; nblines = 0
6937 0000226D 755B                      <1>         jnz    short vga_g_down_6
6938 0000226F 20ED                      <1>         and    ch, ch ; rul = 0
6939 00002271 7557                      <1>         jnz    short vga_g_down_6
6940 00002273 20C9                      <1>         and    cl, cl ; cul = 0
6941 00002275 7553                      <1>         jnz    short vga_g_down_6
6942                                    <1>
6943 00002277 6650                      <1>         push   ax
6944 00002279 A0[CA5E0000]              <1>         mov    al, [VGA_ROWS]
6945 0000227E FEC8                      <1>         dec    al
6946 00002280 38C6                      <1>         cmp    dh, al ; rlr = nbrows-1
6947 00002282 7544                      <1>         jne    short vga_g_down_5
6948 00002284 A0[C45E0000]              <1>           mov     al, [CRT_COLS]  ; = VGA_COLS
6949 00002289 FEC8                      <1>         dec    al
6950 0000228B 38C2                      <1>         cmp    dl, al ; clr = nbcols-1
6951 0000228D 7539                      <1>         jne    short vga_g_down_5
6952 0000228F 6658                      <1>         pop    ax
6953                                    <1>
6954 00002291 66B80502                  <1>         mov    ax, 0205h
6955 00002295 66BACE03                  <1>         mov    dx, 3CEh ; VGAREG_GRDC_ADDRESS
6956 00002299 66EF                      <1>         out    dx, ax
6957 0000229B A0[CA5E0000]              <1>         mov    al, [VGA_ROWS]
6958 000022A0 8A25[C45E0000]            <1>         mov    ah, [CRT_COLS] ; = [VGA_COLS]
6959 000022A6 F6E4                      <1>         mul    ah
6960 000022A8 0FB7D0                    <1>         movzx  edx, ax
6961                                    <1>         ; 08/08/2016
6962 000022AB 0FB605[C65E0000]          <1>         movzx  eax, byte [CHAR_HEIGHT]
6963 000022B2 F7E2                      <1>         mul    edx
6964                                    <1>         ; eax = byte count
6965 000022B4 89C1                      <1>         mov    ecx, eax
6966                                    <1>         ;; 07/08/2016
6967                                    <1>         ;shl   dx, 3 ; * 8 ; * [CHAR_HEIGHT]
6968                                    <1>         ;mov   ecx, edx
6969 000022B6 88D8                      <1>         mov    al, bl ; fill value for blanked lines
6970 000022B8 BF00000A00                <1>         mov    edi, 0A0000h
6971 000022BD F3AA                      <1>         rep    stosb
6972                                    <1>
6973 000022BF B005                      <1>         mov    al, 5
6974 000022C1 66BACE03                  <1>         mov    dx, 3CEh ; VGAREG_GRDC_ADDRESS
6975 000022C5 66EF                      <1>         out    dx, ax ; 0005h
6976                                    <1>
6977 000022C7 C3                        <1>         retn
6978                                    <1>
6979                                    <1> vga_g_down_5:
6980 000022C8 6658                      <1>         pop    ax
6981                                    <1>
6982                                    <1> vga_g_down_6:
6983                                    <1>         ; [ESI] = VGA memory model number for current video mode
6984                                    <1>           ;
6985                                    <1>           ; LINEAR8 equ 5
6986                                    <1>           ; PLANAR4 equ 4
6987                                    <1>           ; PLANAR1 equ 3
6988                                    <1>
6989 000022CA 803E04                   <1>         cmp    byte [esi], PLANAR4
6990 000022CD 742C                      <1>         je     short vga_g_down_planar
6991 000022CF 803E03                    <1>         cmp    byte [esi], PLANAR1
6992 000022D2 7427                      <1>         je     short vga_g_down_planar
6993                                    <1> vga_g_down_linear8:
6994                                    <1>         ; 07/07/2016 (TEMPORARY)
6995                                    <1>         ;
6996                                    <1>         ; cl = upper left column ; cul
6997                                    <1>         ; ch = upper left row ; rul
6998                                    <1>         ; dl = lower rigth column ; clr
6999                                    <1>         ; dh = lower right row ; rlr
7000                                    <1>
7001                                    <1> vga_g_down_l0:
7002                                    <1>         ;{for(i=rlr;i>=rul;i--)
```

```
7003                                    <1>         ; if((i<rul+nblines)||(nblines==0))
7004 000022D4 08C0                      <1>       or    al, al
7005 000022D6 741C                      <1>       jz    short vga_g_down_l2
7006 000022D8 88C4                      <1>       mov   ah, al
7007 000022DA 00EC                      <1>       add   ah, ch
7008                                    <1>       ;jc   short vga_g_down_l2
7009 000022DC 86EE                      <1>       xchg  ch, dh
7010 000022DE 38E5                      <1>       cmp   ch, ah
7011 000022E0 7212                      <1>       jb    short vga_g_down_l2
7012 000022E2 88EC                      <1>       mov   ah, ch
7013 000022E4 28C4                      <1>       sub   ah, al ; ah = i - nblines
7014                                    <1>       ; else
7015                                    <1>       ; vgamem_copy_pl4(cul,i,i-nblines,cols,nbcols,cheight);
7016 000022E6 E877FEFFFF                <1>       call  vgamem_copy_l8
7017                                    <1> vga_g_down_l1:
7018 000022EB 86F5                      <1>       xchg  dh, ch
7019 000022ED FECE                      <1>       dec   dh
7020 000022EF 38EE                      <1>       cmp   dh, ch
7021 000022F1 73E1                      <1>       jnb   short vga_g_down_l0
7022 000022F3 C3                        <1>       retn
7023                                    <1>
7024                                    <1> vga_g_down_l2:
7025                                    <1>       ; vgamem_fill_pl4(cul,i,cols,nbcols,cheight,attr);
7026 000022F4 E8D3FEFFFF                <1>       call  vgamem_fill_l8
7027 000022F9 EBF0                      <1>       jmp   short vga_g_down_l1
7028                                    <1>
7029                                    <1> vga_g_down_planar:
7030                                    <1>       ; cl = upper left column ; cul
7031                                    <1>       ; ch = upper left row ; rul
7032                                    <1>       ; dl = lower rigth column ; clr
7033                                    <1>       ; dh = lower right row ; rlr
7034                                    <1> vga_g_down_pl0:
7035                                    <1>       ;{for(i=rlr;i>=rul;i--)
7036                                    <1>         ; if((i<rul+nblines)||(nblines==0))
7037 000022FB 08C0                      <1>       or    al, al
7038 000022FD 741C                      <1>       jz    short vga_g_down_pl2
7039 000022FF 88C4                      <1>       mov   ah, al
7040 00002301 00EC                      <1>       add   ah, ch
7041                                    <1>       ;jc   short vga_g_down_pl2
7042 00002303 86EE                      <1>       xchg  ch, dh
7043 00002305 38E5                      <1>       cmp   ch, ah
7044 00002307 7212                      <1>       jb    short vga_g_down_pl2
7045 00002309 88EC                      <1>       mov   ah, ch
7046 0000230B 28C4                      <1>       sub   ah, al ; ah = i - nblines
7047                                    <1>       ; else
7048                                    <1>       ; vgamem_copy_pl4(cul,i,i-nblines,cols,nbcols,cheight);
7049 0000230D E88BFDFFFF                <1>       call  vgamem_copy_pl4
7050                                    <1> vga_g_down_pl1:
7051 00002312 86F5                      <1>       xchg  dh, ch
7052 00002314 FECE                      <1>       dec   dh
7053 00002316 38EE                      <1>       cmp   dh, ch
7054 00002318 73E1                      <1>       jnb   short vga_g_down_pl0
7055 0000231A C3                        <1>       retn
7056                                    <1>
7057                                    <1> vga_g_down_pl2:
7058                                    <1>       ; vgamem_fill_pl4(cul,i,cols,nbcols,cheight,attr);
7059 0000231B E8EBFDFFFF                <1>       call  vgamem_fill_pl4
7060 00002320 EBF0                      <1>       jmp   short vga_g_down_pl1
7061                                    <1>
7062                                    <1> ; 07/07/2016
7063                                    <1> ; 27/06/2016 - TRDOS 386 (TRDOS v2.0)
7064                                    <1> ; VIDEO1.ASM - 24/03/1985 (IBM PC-AT BIOS source code)
7065                                    <1> ;--------------------------------------------------
7066                                    <1> ; SCROLL DOWN
7067                                    <1> ;  THIS ROUTINE SCROLLS DOWN THE INFORMATION ON THE CRT
7068                                    <1> ; ENTRY --
7069                                    <1> ;  CH,CL = UPPER LEFT CORNER OF REGION TO SCROLL
7070                                    <1> ;  DH,DL = LOWER RIGHT CORNER OF REGION TO SCROLL
7071                                    <1> ;   BOTH OF THE ABOVE ARE IN CHARACTER POSITIONS
7072                                    <1> ;  BH = FILL VALUE FOR BLANKED LINES
7073                                    <1> ;  AL = # LINES TO SCROLL (AL=0 MEANS BLANK THE ENTIRE FIELD)
7074                                    <1> ;  DS = DATA SEGMENT
7075                                    <1> ;  ES = REGEN SEGMENT
7076                                    <1> ; EXIT --
7077                                    <1> ;  NOTHING, THE SCREEN IS SCROLLED
7078                                    <1> ;--------------------------------------------------
7079                                    <1>
7080                                    <1>       ; cl = upper left column
7081                                    <1>       ; ch = upper left row
7082                                    <1>       ; dl = lower rigth column
7083                                    <1>       ; dh = lower right row
7084                                    <1>       ;
7085                                    <1>       ; al = line count (AL=0 means blank entire fields)
7086                                    <1>       ; bl = fill value for blanked lines
7087                                    <1>       ; bh = unused
7088                                    <1>
7089                                    <1> GRAPHICS_DOWN:
7090                                    <1>       ; 07/07/2016
7091                                    <1>       ;AH = Current video mode, [CRT_MODE]
7092                                    <1>       ;STD                    ; SET DIRECTION
7093 00002322 80FC07                    <1>       cmp   ah, 7
7094 00002325 0F87F4FEFFFF              <1>         ja      vga_graphics_down
7095                                    <1>       ;je   _n0
7096                                    <1>
7097 0000232B 88C7                      <1>       MOV   bh, al            ; save line count in BH
7098 0000232D 6689D0                    <1>       MOV   AX, DX            ; GET LOWER RIGHT POSITION INTO AX REG
7099                                    <1>
7100                                    <1> ;-----      USE CHARACTER SUBROUTINE FOR POSITIONING
7101                                    <1> ;-----      ADDRESS RETURNED IS MULTIPLIED BY 2 FROM CORRECT VALUE
7102                                    <1>
7103 00002330 E8F1010000                <1>       CALL  GRAPH_POSN
7104 00002335 0FB7F8                    <1>       MOVzx eDI, AX                   ; SAVE RESULT AS DESTINATION ADDRESS
7105                                    <1>
```

```
                            <1> ;-----      DETERMINE SIZE OF WINDOW
                            <1>
00002338 6629CA             <1>       SUB    DX, CX
0000233B 6681C20101         <1>         ADD     DX, 101h                ; ADJUST VALUES
00002340 C0E602             <1>       SAL    DH, 2                ; MULTIPLY ROWS BY 4 AT 8 VERT DOTS/CHAR
                            <1>                                    ; AND EVEN/ODD ROWS
                            <1>
                            <1> ;-----      DETERMINE CRT MODE
                            <1>
00002343 803D[C25E0000]06   <1>       CMP    byte [CRT_MODE], 6  ; TEST FOR MEDIUM RES
0000234A 7307               <1>         JNC     short _R12              ; FIND_SOURCE_DOWN
                            <1>
                            <1> ;-----      MEDIUM RES DOWN
0000234C D0E2               <1>       SAL    DL, 1                ; # COLUMNS * 2, SINCE 2 BYTES/CHAR
0000234E 66D1E7             <1>       SAL    DI, 1                ; OFFSET *2 SINCE 2 BYTES/CHAR
00002351 6647               <1>       INC    DI                   ; POINT TO LAST BYTE
                            <1>
                            <1> ;-----      DETERMINE THE SOURCE ADDRESS IN THE BUFFER
                            <1>
                            <1> _R12:                                      ; FIND_SOURCE_DOWN
00002353 81C700800B00       <1>       add    edi, 0B8000h
00002359 6681C7F000         <1>       ADD    DI, 240                   ; POINT TO LAST ROW OF PIXELS
0000235E C0E702             <1>       sal    bh, 2                ; multiply number of lines by 4
00002361 74(06)             <1>       JZ     short 6                   ; IF ZERO, THEN BLANK ENTIRE FIELD
00002363 B050               <1>       MOV    AL, 80               ; 80 BYTES/ROW
00002365 F6E7               <1>       mul    bh                   ; determine offset to source
00002367 89FE               <1>       MOV    eSI, eDI             ; SET UP SOURCE
00002369 6629C6             <1>       SUB    SI, AX               ; SUBTRACT THE OFFSET
0000236C 88F4               <1>       MOV    AH, DH               ; NUMBER OF ROWS IN FIELD
0000236E 28FC               <1>       sub    ah, bh               ; determine number to move
                            <1>
                            <1> ;-----      LOOP THROUGH, MOVING ONE ROW AT A TIME, BOTH EVEN AND ODD FIELDS
                            <1>
                            <1> _R13:                                      ; ROW_LOOP_DOWN
00002370 E823000000         <1>         CALL    _R17                   ; MOVE ONE ROW
00002375 6681EE5020         <1>       SUB    SI, 2000h+80         ; MOVE TO NEXT ROW
0000237A 6681EF5020         <1>       SUB    DI, 2000h+80
0000237F FECC               <1>       DEC    AH                   ; NUMBER OF ROWS TO MOVE
00002381 75ED               <1>         JNZ     short _R13             ; CONTINUE TILL ALL MOVED
                            <1>
                            <1> ;-----      FILL IN THE VACATED LINE(S)
                            <1> _R14:                                      ; CLEAR_ENTRY_DOWN
00002383 88D8               <1>       mov    al, bl               ; attribute to fill with
                            <1> _R15_:                                     ; CLEAR_LOOP_DOWN
00002385 E82A000000         <1>       CALL   _R18                 ; CLEAR A ROW
0000238A 6681EF5020         <1>       SUB    DI, 2000h+80         ; POINT TO NEXT LINE
0000238F FECF               <1>       dec    bh                   ; number of lines to fill
00002391 75F2               <1>         JNZ     short _R15_            ; CLEAR_LOOP_DOWN
                            <1>
00002393 C3                 <1>       retn                        ; EVERYYHING DONE
                            <1>
                            <1> _R16:                                      ; BLANK_FIELD_DOWN
00002394 88F7               <1>       mov    bh, dh               ; set blank count to everything in field
00002396 EBEB               <1>         JMP     short _R14             ; CLEAR THE FIELD
                            <1>
                            <1> ; 27/06/2016 - TRDOS 386 (TRDOS v2.0)
                            <1> ; VIDEO1.ASM - 24/03/1985 (IBM PC-AT BIOS source code)
                            <1>
                            <1> ;-----      ROUTINE TO MOVE ONE ROW OF INFORMATION
                            <1>
                            <1> _R17:
00002398 0FB6CA             <1>       MOVzx  ecx, DL                   ; NUMBER OF BYTES IN THE ROW
0000239B 56                 <1>       PUSH   eSI
0000239C 57                 <1>       PUSH   eDI                  ; SAVE POINTERS
0000239D F3A4               <1>       REP    MOVSB                ; MOVE THE EVEN FIELD
0000239F 5F                 <1>       POP    eDI
000023A0 5E                 <1>       POP    eSI
000023A1 6681C60020         <1>       ADD    SI, 2000h
000023A6 6681C70020         <1>       ADD    DI, 2000h            ; POINT TO THE ODD FIELD
000023AB 56                 <1>       PUSH   eSI
000023AC 57                 <1>       PUSH   eDI                  ; SAVE THE POINTERS
000023AD 88D1               <1>       MOV    CL, DL               ; COUNT BACK
000023AF F3A4               <1>       REP    MOVSB                ; MOVE THE ODD FIELD
000023B1 5F                 <1>       POP    eDI
000023B2 5E                 <1>       POP    eSI                  ; POINTERS BACK
000023B3 C3                 <1>       RETn                        ; RETURN TO CALLER
                            <1>
                            <1> ;-----      CLEAR A SINGLE ROW
                            <1>
                            <1> _R18:
000023B4 0FB6CA             <1>       MOVzx  ecx, DL                   ; NUMBER OF BYTES IN FIELD
000023B7 57                 <1>       PUSH   eDI                  ; SAVE POINTER
000023B8 F3AA               <1>       REP    STOSB                ; STORE THE NEW VALUE
000023BA 5F                 <1>       POP    eDI                  ; POINTER BACK
000023BB 6681C70020         <1>       ADD    DI, 2000h            ; POINT TO ODD FIELD
000023C0 57                 <1>       PUSH   eDI
000023C1 88D1               <1>       MOV    CL, DL
000023C3 F3AA               <1>       REP    STOSB                ; FILL THE ODD FIELD
000023C5 5F                 <1>       POP    eDI
000023C6 C3                 <1>       RETn                        ; RETURN TO CALLER
                            <1>
                            <1> ; 04/07/2016
                            <1> ; 01/07/2016
                            <1> ; 30/06/2016 - TRDOS 386 (TRDOS v2.0)
                            <1> ; VIDEO1.ASM - 24/03/1985 (IBM PC-AT BIOS source code)
                            <1> ;-----------------------------------------------
                            <1> ; GRAPHICS WRITE
                            <1> ;  THIS ROUTINE WRITES THE ASCII CHARACTER TO THE CURRENT
                            <1> ;  POSITION ON THE SCREEN.
                            <1> ; ENTRY --
                            <1> ;  AL = CHARACTER TO WRITE
                            <1> ;  BL = COLOR ATTRIBUTE TO BE USED FOR FOREGROUND COLOR
                            <1> ;     IF BIT 7 IS SET, THE CHAR IS XOR'D INTO THE REGEN BUFFER
```

```
                        <1> ;     (0 IS USED FOR THE BACKGROUND COLOR)
                        <1> ;  CX = NUMBER OF CHARS TO WRITE
                        <1> ;  DS = DATA SEGMENT
                        <1> ;  ES = REGEN SEGMENT
                        <1> ; EXIT --
                        <1> ;  NOTHING IS RETURNED
                        <1> ;
                        <1> ; GRAPHICS READ
                        <1> ;  THIS ROUTINE READS THE ASCII CHARACTER AT THE CURRENT CURSOR
                        <1> ;  POSITION ON THE SCREEN BY MATCHING THE DOTS ON THE SCREEN TO THE
                        <1> ;  CHARACTER GENERATOR CODE POINTS
                        <1> ; ENTRY --
                        <1> ;  NONE (0 IS ASSUMED AS THE BACKGROUND COLOR)
                        <1> ; EXIT --
                        <1> ;  AL = CHARACTER READ AT THAT POSITION (0 RETURNED IF NONE FOUND)
                        <1> ;
                        <1> ; FOR BOTH ROUTINES, THE IMAGES USED TO FORM CHARS ARE CONTAINED IN ROM
                        <1> ;  FOR THE 1ST 128 CHARS.  TO ACCESS CHARS IN THE SECOND HALF, THE USER
                        <1> ;  MUST INITIALIZE THE VECTOR AT INTERRUPT 1FH (LOCATION 0007CH) TO
                        <1> ;  POINT TO THE USER SUPPLIED TABLE OF GRAPHIC IMAGES (8X8 BOXES).
                        <1> ;  FAILURE TO DO SO WILL CAUSE IN STRANGE RESULTS
                        <1> ;-------------------------------------------------
                        <1>
                        <1> GRAPHICS_WRITE:
000023C7 25FF000000     <1>       and    eax, 0FFh          ; ZERO TO HIGH OF CODE POINT
000023CC 50             <1>       PUSH   eAX                ; SAVE CODE POINT VALUE
                        <1>
                        <1> ;-----      DETERMINE POSITION IN REGEN BUFFER TO PUT CODE POINTS
                        <1>
000023CD E84D010000     <1>       CALL   S26                ; FIND LOCATION IN REGEN BUFFER
000023D2 89C7           <1>       MOV    eDI, eAX           ; REGEN POINTER IN DI
                        <1>
                        <1> ;-----      DETERMINE REGION TO GET CODE POINTS FROM
                        <1>
000023D4 58             <1>       POP    eAX                ; RECOVER CODE POINT
                        <1>
000023D5 BE[A0260100]   <1>       MOV    eSI, CRT_CHAR_GEN  ; OFFSET OF IMAGES
                        <1>
                        <1> ;-----      DETERMINE GRAPHICS MODE IN OPERATION
                        <1>                                ; DETERMINE_MODE
000023DA 66C1E003       <1>       SAL    AX, 3              ; MULTIPLY CODE POINT VALUE BY 8
000023DE 01C6           <1>       ADD    eSI, eAX           ; SI HAS OFFSET OF DESIRED CODES
                        <1>
000023E0 803D[C25E0000]06 <1>     CMP    byte [CRT_MODE], 6
000023E7 7231           <1>       JC     short S6           ; TEST FOR MEDIUM RESOLUTION MODE
                        <1>
                        <1> ;-----      HIGH RESOLUTION MODE
                        <1>
000023E9 81C700800B00   <1>       add    edi, 0B8000h
                        <1> S1:                            ; HIGH_CHAR
000023EF 57             <1>       PUSH   eDI                ; SAVE REGEN POINTER
000023F0 56             <1>       PUSH   eSI                ; SAVE CODE POINTER
000023F1 B604           <1>       MOV    DH, 4              ; NUMBER OF TIMES THROUGH LOOP
                        <1> S2:
000023F3 AC             <1>       LODSB                     ; GET BYTE FROM CODE POINTS
000023F4 F6C380         <1>       TEST   BL, 80H                  ; SHOULD WE USE THE FUNCTION
000023F7 7515           <1>       JNZ    short S5           ; TO PUT CHAR IN
000023F9 AA             <1>       STOSB                     ; STORE IN REGEN BUFFER
000023FA AC             <1>       LODSB
                        <1> S4:
000023FB 8887FF1F0000   <1>       MOV    [eDI+2000H-1], AL ; STORE IN SECOND HALF
00002401 83C74F         <1>       ADD    eDI, 79                  ; MOVE TO NEXT ROW IN REGEN
00002404 FECE           <1>       DEC    DH                 ; DONE WITH LOOP
00002406 75EB           <1>       JNZ    short S2
00002408 5E             <1>       POP    eSI
00002409 5F             <1>       POP    eDI                ; RECOVER REGEN POINTER
0000240A 47             <1>       INC    eDI                ; POINT TO NEXT CHAR POSITION
0000240B E2E2           <1>       LOOP   S1                 ; MORE CHARS TO WRITE
0000240D C3             <1>       retn
                        <1>
                        <1> S5:
0000240E 3207           <1>       XOR    AL, [eDI]          ; EXCLUSIVE OR WITH CURRENT
00002410 AA             <1>       STOSB                     ; STORE THE CODE POINT
00002411 AC             <1>       LODSB                     ; AGAIN FOR ODD FIELD
00002412 3287FF1F0000   <1>       XOR    AL, [eDI+2000H-1]
00002418 EBE1           <1>       JMP    short S4           ; BACK TO MAINSTREAM
                        <1>
                        <1> ;-----      MEDIUM RESOLUTION WRITE
                        <1> S6:                            ; MED_RES_WRITE
0000241A 88DA           <1>       MOV    DL, BL             ; SAVE HIGH COLOR BIT
0000241C 66D1E7         <1>       SAL    DI, 1              ; OFFSET*2 SINCE 2 BYTES/CHAR
                        <1>                                ; EXPAND BL TO FULL WORD OF COLOR
0000241F 80E303         <1>       AND    BL, 3              ; ISOLATE THE COLOR BITS ( LOW 2 BITS )
00002422 B055           <1>       MOV    AL, 055H           ; GET BIT CONVERSION MULTIPLIER
00002424 F6E3           <1>       MUL    BL                 ; EXPAND 2 COLOR BITS TO 4 REPLICATIONS
00002426 88C3           <1>       MOV    BL, AL             ; PLACE BACK IN WORK REGISTER
00002428 88C7           <1>       MOV    BH, AL             ; EXPAND TO 8 REPLICATIONS OF COLOR BITS
0000242A 81C700800B00   <1>       add    edi, 0B8000h
                        <1> S7:                                  ; MED_CHAR
00002430 57             <1>       PUSH   eDI                ; SAVE REGEN POINTER
00002431 56             <1>       PUSH   eSI                ; SAVE THE CODE POINTER
00002432 B604           <1>       MOV    DH, 4              ; NUMBER OF LOOPS
                        <1> S8:
00002434 AC             <1>       LODSB                     ; GET CODE POINT
00002435 E8B3000000     <1>       CALL   S21                ; DOUBLE UP ALL THE BITS
0000243A 6621D8         <1>       AND    AX, BX             ; CONVERT TO FOREGROUND COLOR ( 0 BACK )
0000243D 86E0           <1>       XCHG   AH, AL             ; SWAP HIGH/LOW BYTES FOR WORD MOVE
0000243F F6C280         <1>       TEST   DL, 80H                 ; IS THIS XOR FUNCTION
00002442 7403           <1>       JZ     short S9           ; NO, STORE IT IN AS IS
00002444 663307         <1>       XOR    AX, [eDI]          ; DO FUNCTION WITH LOW/HIGH
                        <1> S9:
00002447 668907         <1>       MOV    [eDI], AX          ; STORE FIRST BYTE HIGH, SECOND LOW
0000244A AC             <1>       LODSB                     ; GET CODE POINT
```

```
7312 0000244B E89D000000          <1>       CALL   S21
7313 00002450 6621D8              <1>       AND    AX, BX              ; CONVERT TO COLOR
7314 00002453 86E0                <1>       XCHG   AH, AL              ; SWAP HIGH/LOW BYTES FOR WORD MOVE
7315 00002455 F6C280              <1>       TEST   DL, 80H                   ; AGAIN, IS THIS XOR FUNCTION
7316 00002458 7407                <1>       JZ     short _S10          ; NO, JUST STORE THE VALUES
7317 0000245A 66338700200000      <1>       XOR    AX, [eDI+2000H]           ; FUNCTION WITH FIRST HALF LOW
7318                              <1> _S10:
7319 00002461 66898700200000      <1>       MOV    [eDI+2000H], AX           ; STORE SECOND PORTION HIGH
7320 00002468 6683C750            <1>       ADD    DI, 80              ; POINT TO NEXT LOCATION
7321 0000246C FECE                <1>       DEC    DH
7322 0000246E 75C4                <1>       JNZ    short S8            ; KEEP GOING
7323 00002470 5E                  <1>       POP    eSI                 ; RECOVER CODE POINTER
7324 00002471 5F                  <1>       POP    eDI                 ; RECOVER REGEN POINTER
7325 00002472 47                  <1>       INC    eDI                 ; POINT TO NEXT CHAR POSITION
7326 00002473 47                  <1>       INC    eDI
7327 00002474 E2BA                <1>       LOOP   S7                  ; MORE TO WRITE
7328 00002476 C3                  <1>       retn
7329                              <1>
7330                              <1> ; 04/07/2016
7331                              <1> ; 01/07/2016
7332                              <1> ; 30/06/2016 - TRDOS 386 (TRDOS v2.0)
7333                              <1> ; VIDEO1.ASM - 24/03/1985 (IBM PC-AT BIOS source code)
7334                              <1> ;---------------------------------------
7335                              <1> ; GRAPHICS READ
7336                              <1> ;---------------------------------------
7337                              <1> GRAPHICS_READ:
7338 00002477 E8A3000000          <1>       CALL   S26                 ; CONVERTED TO OFFSET IN REGEN
7339 0000247C 89C6                <1>       MOV    eSI, eAX            ; SAVE IN SI
7340 0000247E 81C600800B00        <1>       add    esi, 0B8000h        ; 01/07/2016
7341 00002484 83EC08              <1>       SUB    eSP, 8              ; ALLOCATE SPACE FOR THE READ CODE POINT
7342 00002487 89E5                <1>       MOV    eBP, eSP            ; POINTER TO SAVE AREA
7343                              <1>
7344                              <1> ;-----       DETERMINE GRAPHICS MODES
7345 00002489 B604                <1>       mov    dh, 4               ; number of passes ; 01/07/2016
7346 0000248B 803D[C25E0000]06    <1>       CMP    byte [CRT_MODE], 6
7347 00002492 7219                <1>       JC     short S12           ; MEDIUM RESOLUTION
7348                              <1>
7349                              <1> ;-----       HIGH RESOLUTION READ
7350                              <1> ;-----       GET VALUES FROM REGEN BUFFER AND CONVERT TO CODE POINT
7351                              <1>       ;MOV   DH,4                ; NUMBER OF PASSES
7352                              <1> S11:
7353 00002494 8A06                <1>       MOV    AL, [eSI]           ; GET FIRST BYTE
7354 00002496 884500              <1>       MOV    [eBP], AL           ; SAVE IN STORAGE AREA
7355 00002499 45                  <1>       INC    eBP                 ; NEXT LOCATION
7356 0000249A 8A8600200000        <1>       MOV    AL, [eSI+2000H]           ; GET LOWER REGION BYTE
7357 000024A0 884500              <1>       MOV    [eBP], AL           ; ADJUST AND STORE
7358 000024A3 45                  <1>       INC    eBP
7359 000024A4 83C650              <1>       ADD    eSI, 80                   ; POINTER INTO REGEN
7360 000024A7 FECE                <1>       DEC    DH                  ; LOOP CONTROL
7361 000024A9 75E9                <1>       JNZ    short S11           ; DO IT SOME MORE
7362 000024AB EB1D                <1>       JMP    SHORT S14           ; GO MATCH THE SAVED CODE POINTS
7363                              <1>
7364                              <1> ;-----       MEDIUM RESOLUTION READ
7365                              <1> S12:
7366 000024AD 66D1E6              <1>       SAL    SI, 1               ; OFFSET*2 SINCE 2 BYTES/CHAR
7367                              <1>       ;MOV   DH, 4               ; NUMBER OF PASSES
7368                              <1> S13:
7369 000024B0 E84D000000          <1>       CALL   S23                 ; GET BYTES FROM REGEN INTO SINGLE SAVE
7370 000024B5 81C6FE1F0000        <1>       ADD    eSI, 2000H-2        ; GO TO LOWER REGION
7371 000024BB E842000000          <1>       CALL   S23                 ; GET THIS PAIR INTO SAVE
7372 000024C0 81EEB21F0000        <1>       SUB    eSI, 2000H-80+2           ; ADJUST POINTER BACK INTO UPPER
7373 000024C6 FECE                <1>       DEC    DH
7374 000024C8 75E6                <1>       JNZ    short S13           ; KEEP GOING UNTIL ALL 8 DONE
7375                              <1>
7376                              <1> ;-----       SAVE AREA HAS CHARACTER IN IT, MATCH IT
7377                              <1> S14:                             ; FIND_CHAR
7378 000024CA BF[A0260100]        <1>       MOV    eDI, CRT_CHAR_GEN   ; ESTABLISH ADDRESSING
7379 000024CF 83ED08              <1>       SUB    eBP, 8              ; ADJUST POINTER TO START OF SAVE AREA
7380 000024D2 89EE                <1>       MOV    eSI, eBP
7381                              <1> S15:
7382 000024D4 66B80001            <1>       mov    ax, 256                   ; NUMBER TO TEST AGAINST
7383                              <1> S16:
7384 000024D8 56                  <1>       PUSH   eSI                 ; SAVE SAVE AREA POINTER
7385 000024D9 57                  <1>       PUSH   eDI                 ; SAVE CODE POINTER
7386                              <1>       ;MOV   eCX, 4              ; NUMBER OF WORDS TO MATCH
7387                              <1>       ;REPE  CMPSW               ; COMPARE THE 8 BYTES AS WORDS
7388 000024DA A7                  <1>       cmpsd                      ; compare first 4 bytes
7389 000024DB 7501                <1>       jne    short S17           ;
7390 000024DD A7                  <1>       cmpsd                      ; compare last 4 bytes
7391                              <1> S17:
7392 000024DE 5F                  <1>       POP    eDI                 ; RECOVER THE POINTERS
7393 000024DF 5E                  <1>       POP    eSI
7394                              <1>       ;JZ    short S18           ; IF ZERO FLAG SET, THEN MATCH OCCURRED
7395 000024E0 7407                <1>       je     short S18
7396                              <1>       ;                          ; NO MATCH, MOVE ON TO NEXT
7397 000024E2 83C708              <1>       ADD    eDI, 8              ; NEXT CODE POINT
7398 000024E5 6648                <1>       dec    ax                  ; LOOP CONTROL
7399 000024E7 75EF                <1>       JNZ    short S16           ; DO ALL OF THEM
7400                              <1>
7401                              <1> ;-----       CHARACTER IS FOUND ( AL=0 IF NOT FOUND )
7402                              <1> S18:
7403 000024E9 83C408              <1>       ADD    eSP, 8              ; READJUST THE STACK, THROW AWAY SAVE
7404 000024EC C3                  <1>       retn                       ; ALL DONE
7405                              <1>
7406                              <1> ; 30/06/2016 - TRDOS 386 (TRDOS v2.0)
7407                              <1> ; VIDEO1.ASM - 24/03/1985 (IBM PC-AT BIOS source code)
7408                              <1> ;-----------------------------------------
7409                              <1> ; EXPAND BYTE
7410                              <1> ;  THIS ROUTINE TAKES THE BYTE IN AL AND DOUBLES ALL
7411                              <1> ;  OF THE BITS, TURNING THE 8 BITS INTO 16 BITS.
7412                              <1> ;  THE RESULT IS LEFT IN AX
7413                              <1> ;-----------------------------------------
7414                              <1> S21:
```

```
7415 000024ED 6651              <1>       PUSH   CX                ; SAVE REGISTER
7416                            <1>       ;MOV   CX, 8             ; SHIFT COUNT REGISTER FOR ONE BYTE
7417 000024EF B108              <1>       mov    cl, 8
7418                            <1> S22:
7419 000024F1 D0C8              <1>       ROR    AL,1              ; SHIFT BITS, LOW BIT INTO CARRY FLAG
7420 000024F3 66D1DD            <1>       RCR    BP,1              ; MOVE CARRY FLAG (LOW BIT INTO RESULTS
7421 000024F6 66D1FD            <1>       SAR    BP,1              ; SIGN EXTEND HIGH BIT (DOUBLE IT)
7422                            <1>       ;LOOP  S22               ; REPEAT FOR ALL 8 BITS
7423 000024F9 FEC9              <1>       dec    cl
7424 000024FB 75F4              <1>       jnz    short S22
7425 000024FD 6695              <1>       XCHG   AX, BP            ; MOVE RESULTS TO PARAMETER REGISTER
7426 000024FF 6659              <1>       POP    CX                ; RECOVER REGISTER
7427 00002501 C3                <1>       RETn                     ; ALL DONE
7428                            <1>
7429                            <1> ; 01/07/2016 - TRDOS 386 (TRDOS v2.0)
7430                            <1> ; VIDEO1.ASM - 24/03/1985 (IBM PC-AT BIOS source code)
7431                            <1> ;-----------------------------------------------
7432                            <1> ; MED_READ_BYTE
7433                            <1> ; THIS ROUTINE WILL TAKE 2 BYTES FROM THE REGEN BUFFER,
7434                            <1> ;  COMPARE AGAINST THE CURRENT FOREGROUND COLOR, AND PLACE
7435                            <1> ;  THE CORRESPONDING ON/OFF BIT PATTERN INTO THE CURRENT
7436                            <1> ;  POSITION IN THE SAVE AREA
7437                            <1> ; ENTRY --
7438                            <1> ;  SI,DS = POINTER TO REGEN AREA OF INTEREST
7439                            <1> ;  BX = EXPANDED FOREGROUND COLOR
7440                            <1> ;  BP = POINTER TO SAVE AREA
7441                            <1> ; EXIT --
7442                            <1> ;  SI AND BP ARE INCREMENTED
7443                            <1> ;-----------------------------------------------
7444                            <1> S23:
7445 00002502 66AD              <1>       LODSW                    ; GET FIRST BYTE AND SECOND BYTES
7446 00002504 86C4              <1>       XCHG   AL, AH            ; SWAP FOR COMPARE
7447 00002506 66B900C0          <1>       MOV    CX, 0C000H        ; 2 BIT MASK TO TEST THE ENTRIES
7448 0000250A B200              <1>       MOV    DL, 0             ; RESULT REGISTER
7449                            <1> S24:
7450 0000250C 6685C8            <1>       TEST   AX, CX            ; IS THIS SECTION BACKCROUND?
7451 0000250F 7401              <1>         JZ     short S25              ; IF ZERO, IT IS BACKGROUND (CARRY=0)
7452 00002511 F9                <1>       STC                      ; WASN'T, SO SET CARRY
7453                            <1> S25:
7454 00002512 D0D2              <1>       RCL    DL, 1             ; MOVE THAT BIT INTO THE RESULT
7455 00002514 66C1E902          <1>       SHR    CX, 2             ; MOVE THE MASK TO THE RIGHT BY 2 BITS
7456 00002518 73F2              <1>       JNC    short S24         ; DO IT AGAIN IF MASK DIDN'T FALL OUT
7457 0000251A 885500            <1>       MOV    [eBP], DL         ; STORE RESULT IN SAVE AREA
7458 0000251D 45                <1>       INC    eBP               ; ADJUST POINTER
7459 0000251E C3                <1>       RETn                     ; ALL DONE
7460                            <1>
7461                            <1> ; 30/06/2016 - TRDOS 386 (TRDOS v2.0)
7462                            <1> ; VIDEO1.ASM - 24/03/1985 (IBM PC-AT BIOS source code)
7463                            <1> ;----------------------------------------
7464                            <1> ; V4_POSITION
7465                            <1> ;  THIS ROUTINE TAKES THE CURSOR POSITION CONTAINED IN
7466                            <1> ;  THE MEMORY LOCATION, AND CONVERTS IT INTO AN OFFSET
7467                            <1> ;  INTO THE REGEN BUFFER, ASSUMING ONE BYTE/CHAR.
7468                            <1> ;  FOR MEDIUM RESOLUTION GRAPHICS, THE NUMBER MUST
7469                            <1> ;  BE DOUBLED.
7470                            <1> ; ENTRY -- NO REGISTERS,MEMORY LOCATION @CURSOR_POSN IS USED
7471                            <1> ; EXIT--
7472                            <1> ;  AX CONTAINS OFFSET INTO REGEN BUFFER
7473                            <1> ;----------------------------------------
7474                            <1> S26:
7475 0000251F 0FB705[3E520100]  <1>       movzx  eax, word [CURSOR_POSN]   ; GET CURRENT CURSOR
7476                            <1> GRAPH_POSN:
7477 00002526 53                <1>       PUSH   eBX               ; SAVE REGISTER
7478 00002527 0FB6D8            <1>       movzx  ebx, al                 ; SAVE A COPY OF CURRENT CURSOR
7479 0000252A A0[C45E0000]      <1>       MOV    AL, [CRT_COLS]          ; GET BYTES PER COLUMN
7480 0000252F F6E4              <1>       MUL    AH                ; MULTIPLY BY ROWS
7481 00002531 66C1E002          <1>       SHL    AX, 2             ; MULTIPLY * 4 SINCE 4 ROWS/BYTE
7482 00002535 01D8              <1>       ADD    eAX, eBX          ; DETERMINE OFFSET
7483 00002537 5B                <1>       POP    eBX               ; RECOVER POINTER
7484 00002538 C3                <1>       RETn                     ; ALL DONE
7485                            <1>
7486                            <1> ; 09/07/2016
7487                            <1> ; 01/07/2016 - TRDOS 386 (TRDOS v2.0)
7488                            <1> ; VIDEO1.ASM - 24/03/1985 (IBM PC-AT BIOS source code)
7489                            <1> ;-----------------------------------------
7490                            <1> ; SET_COLOR
7491                            <1> ;     THIS ROUTINE WILL ESTABLISH THE BACKGROUND COLOR, THE OVERSCAN COLOR,
7492                            <1> ;     AND THE FOREGROUND COLOR SET FOR MEDIUM RESOLUTION GRAPHICS
7493                            <1> ; INPUT
7494                            <1> ;     (BH) HAS COLOR ID
7495                            <1> ;           IF BH=0, THE BACKGROUND COLOR VALUE IS SET
7496                            <1> ;                 FROM THE LOW BITS OF BL (0-31)
7497                            <1> ;           IF BH=1, THE PALETTE SELECTION IS MADE
7498                            <1> ;                 BASED ON THE LOW BIT OF BL:
7499                            <1> ;                       0 = GREEN, RED, YELLOW FOR COLORS 1,2,3
7500                            <1> ;                       1 = BLUE, CYAN, MAGENTA FOR COLORS 1,2,3
7501                            <1> ;     (BL) HAS THE COLOR VALUE TO BE USED
7502                            <1> ; OUTPUT
7503                            <1> ;     THE COLOR SELECTION IS UPDATED
7504                            <1> ;-----------------------------------------
7505                            <1> SET_COLOR:
7506 00002539 803D[C25E0000]07  <1>         cmp     byte [CRT_MODE], 7      ; 09/07/2016
7507 00002540 0F870EF0FFFF      <1>       ja     VIDEO_RETURN       ; nothing to do for VGA modes
7508                            <1>
7509                            <1>       ;MOV   DX, [ADDR_6845]           ; I/O PORT FOR PALETTE
7510                            <1>       ;mov   dx, 3D4h
7511                            <1>       ;ADD   DX,5               ; OVERSCAN PORT
7512 00002546 66BAD903          <1>       mov    dx, 3D9h
7513 0000254A A0[C55E0000]      <1>       MOV    AL, [CRT_PALETTE]  ; GET THE CURRENT PALETTE VALUE
7514 0000254F 08FF              <1>       OR     BH, BH             ; IS THIS COLOR 0?
7515 00002551 7512              <1>       JNZ    short M20          ; OUTPUT COLOR 1
7516                            <1>
7517                            <1> ;-----      HANDLE COLOR 0 BY SETTING THE BACKGROUND COLOR
```

```
7518                                    <1>
7519 00002553 24E0                      <1>       AND    AL, 0E0H           ; TURN OFF LOW 5 BITS OF CURRENT
7520 00002555 80E31F                    <1>       AND    BL, 01FH           ; TURN OFF HIGH 3 BITS OF INPUT VALUE
7521 00002558 08D8                      <1>       OR     AL, BL             ; PUT VALUE INTO REGISTER
7522                                    <1> M19:                            ; OUTPUT THE PALETTE
7523 0000255A EE                        <1>       OUT    DX, AL             ; OUTPUT COLOR SELECTION TO 3D9 PORT
7524 0000255B A2[C55E0000]              <1>       MOV    [CRT_PALETTE], AL  ; SAVE THE COLOR VALUE
7525 00002560 E9EFEFFFFF                <1>       JMP    VIDEO_RETURN
7526                                    <1>
7527                                    <1> ;-----      HANDLE COLOR 1 BY SELECTING THE PALETTE TO BE USED
7528                                    <1>
7529                                    <1> M20:
7530 00002565 24DF                      <1>       AND    AL, 0DFH           ; TURN OFF PALETTE SELECT BIT
7531 00002567 D0EB                      <1>       SHR    BL, 1              ; TEST THE LOW ORDER BIT OF BL
7532 00002569 73EF                      <1>       JNC    short M19          ; ALREADY DONE
7533 0000256B 0C20                      <1>       OR     AL, 20H                 ; TURN ON PALETTE SELECT BIT
7534 0000256D EBEB                      <1>       JMP    short M19          ; GO DO IT
7535                                    <1>
7536                                    <1> ; 09/07/2016
7537                                    <1> ; 01/07/2016 - TRDOS 386 (TRDOS v2.0)
7538                                    <1> ; VIDEO1.ASM - 24/03/1985 (IBM PC-AT BIOS source code)
7539                                    <1> ;-----------------------------------------
7540                                    <1> ; READ DOT -- WRITE DOT
7541                                    <1> ; THESE ROUTINES WILL WRITE A DOT, OR READ THE
7542                                    <1> ;  DOT AT THE INDICATED LOCATION
7543                                    <1> ; ENTRY --
7544                                    <1> ;   DX = ROW (0-199)     (THE ACTUAL VALUE DEPENDS ON THE MODE)
7545                                    <1> ;   CX = COLUMN ( 0-639) ( THE VALUES ARE NOT RANGE CHECKED )
7546                                    <1> ;   AL = DOT VALUE TO WRITE (1,2 OR 4 BITS DEPENDING ON MODE,
7547                                    <1> ;     REQUIRED FOR WRITE DOT ONLY, RIGHT JUSTIFIED)
7548                                    <1> ;     BIT 7 OF AL = 1 INDICATES XOR THE VALUE INTO THE LOCATION
7549                                    <1> ;   DS = DATA SEGMENT
7550                                    <1> ;   ES = REGEN SEGMENT
7551                                    <1> ;
7552                                    <1> ; EXIT
7553                                    <1> ;     AL = DOT VALUE READ, RIGHT JUSTIFIED, READ ONLY
7554                                    <1> ;-----------------------------------------
7555                                    <1>
7556                                    <1> READ_DOT:
7557                                    <1>       ; 09/07/2016
7558 0000256F 8A25[C25E0000]            <1>       mov    ah,  [CRT_MODE]
7559 00002575 80FC07                    <1>       cmp    ah, 7 ; 6!?
7560 00002578 760A                      <1>       jna    short read_dot_cga
7561                                    <1>
7562 0000257A E8CB030000                <1>       call   vga_read_pixel
7563                                    <1>       ; al = pixel value
7564 0000257F E9D5EFFFFF                <1>       jmp    _video_return
7565                                    <1>
7566                                    <1> read_dot_cga:
7567                                    <1>       ;je    VIDEO_RETURN ; 7
7568 00002584 80FC04                    <1>       cmp    ah, 4 ; graphics ?
7569 00002587 0F82C7EFFFFF              <1>       jb     VIDEO_RETURN ; no, text mode, nothing to do
7570                                    <1>
7571 0000258D E855000000                <1>       CALL   R3                 ; DETERMINE BYTE POSITION OF DOT
7572 00002592 8A06                      <1>       MOV    AL, [eSI]          ; GET THE BYTE
7573 00002594 20E0                      <1>       AND    AL, AH             ; MASK OFF THE OTHER BITS IN THE BYTE
7574 00002596 D2E0                      <1>       SHL    AL, CL             ; LEFT JUSTIFY THE VALUE
7575 00002598 88F1                      <1>       MOV    CL, DH             ; GET NUMBER OF BITS IN RESULT
7576 0000259A D2C0                      <1>       ROL    AL, CL             ; RIGHT JUSTIFY THE RESULT
7577                                    <1>       ;JMP   VIDEO_RETURN       ; RETURN FROM VIDEO I/O
7578 0000259C 0FB6C0                    <1>       movzx  eax, al
7579 0000259F E9B5EFFFFF                <1>       jmp    _video_return
7580                                    <1>
7581                                    <1> ; 09/07/2016
7582                                    <1> ; 01/07/2016 - TRDOS 386 (TRDOS v2.0)
7583                                    <1> ; VIDEO1.ASM - 24/03/1985 (IBM PC-AT BIOS source code)
7584                                    <1>
7585                                    <1> WRITE_DOT:
7586                                    <1>       ; 09/07/2016
7587 000025A4 8A25[C25E0000]            <1>       mov    ah, [CRT_MODE]
7588 000025AA 80FC07                    <1>       cmp    ah, 7 ; 6!?
7589 000025AD 760A                      <1>       jna    short write_dot_cga
7590                                    <1>
7591 000025AF E805030000                <1>       call   vga_write_pixel
7592 000025B4 E99BEFFFFF                <1>       jmp    VIDEO_RETURN
7593                                    <1>
7594                                    <1> write_dot_cga:
7595                                    <1>       ;je    VIDEO_RETURN ; 7
7596 000025B9 80FC04                    <1>       cmp    ah, 4 ; graphics ?
7597 000025BC 0F8292EFFFFF              <1>       jb     VIDEO_RETURN ; no, text mode, nothing to do
7598                                    <1>
7599                                    <1>       ;PUSH  AX                 ; SAVE DOT VALUE
7600 000025C2 6650                      <1>       PUSH   AX                 ; TWICE
7601 000025C4 E81E000000                <1>       CALL   R3                 ; DETERMINE BYTE POSITION OF THE DOT
7602 000025C9 D2E8                      <1>       SHR    AL, CL             ; SHIFT TO SET UP THE BITS FOR OUTPUT
7603 000025CB 20E0                      <1>       AND    AL, AH             ; STRIP OFF THE OTHER BITS
7604 000025CD 8A0E                      <1>       MOV    CL, [eSI]          ; GET THE CURRENT BYTE
7605 000025CF 665B                      <1>       POP    BX                 ; RECOVER XOR FLAG
7606 000025D1 F6C380                    <1>       TEST   BL, 80H                 ; IS IT ON
7607 000025D4 750D                      <1>       JNZ    short R2           ; YES, XOR THE DOT
7608 000025D6 F6D4                      <1>       NOT    AH                 ; SET MASK TO REMOVE THE INDICATED BITS
7609 000025D8 20E1                      <1>       AND    CL, AH
7610 000025DA 08C8                      <1>       OR     AL, CL             ; OR IN THE NEW VALUE OF THOSE BITS
7611                                    <1> R1:                             ; FINISH_DOT
7612 000025DC 8806                      <1>       MOV    [eSI], AL          ; RESTORE THE BYTE IN MEMORY
7613                                    <1>       ;POP   AX
7614 000025DE E971EFFFFF                <1>       JMP    VIDEO_RETURN       ; RETURN FROM VIDEO I/O
7615                                    <1> R2:                             ; XOR_DOT
7616 000025E3 30C8                      <1>       XOR    AL, CL             ; EXCLUSIVE OR THE DOTS
7617 000025E5 EBF5                      <1>       JMP    short R1           ; FINISH UP THE WRITING
7618                                    <1>
7619                                    <1> ; 01/07/2016 - TRDOS 386 (TRDOS v2.0)
7620                                    <1> ; VIDEO1.ASM - 24/03/1985 (IBM PC-AT BIOS source code)
```

```
                                <1>
                                <1> ;-----------------------------------------------
                                <1> ; THIS SUBROUTINE DETERMINES THE REGEN BYTE LOCATION OF THE
                                <1> ; INDICATED ROW COLUMN VALUE IN GRAPHICS MODE.
                                <1> ; ENTRY --
                                <1> ;  DX = ROW VALUE (0-199)
                                <1> ;  CX = COLUMN VALUE (0-639)
                                <1> ; EXIT --
                                <1> ;  SI = OFFSET INTO REGEN BUFFER FOR BYTE OF INTEREST
                                <1> ;  AH = MASK TO STRIP OFF THE BITS OF INTEREST
                                <1> ;  CL = BITS TO SHIFT TO RIGHT JUSTIFY THE MASK IN AH
                                <1> ;  DH = # BITS IN RESULT
                                <1> ;  BX = MODIFIED
                                <1> ;-----------------------------------------------
                                <1> R3:
                                <1>
                                <1> ;-----      DETERMINE 1ST BYTE IN INDICATED ROW BY MULTIPLYING ROW VALUE BY 40
                                <1> ;-----       ( LOW BIT OF ROW DETERMINES EVEN/ODD, 80 BYTES/ROW )
                                <1>
000025E7 0FB7F0                 <1>       movzx  esi, ax                   ; WILL SAVE AL AND AH DURING OPERATION
000025EA B028                   <1>       MOV    AL, 40
000025EC F6E2                   <1>       MUL    DL                  ; AX= ADDRESS OF START OF INDICATED ROW
000025EE A808                   <1>       TEST   AL, 08H             ; TEST FOR EVEN/ODD ROW CALCULATED
000025F0 7404                   <1>       JZ     short R4            ; JUMP IF EVEN ROW
000025F2 6605D81F               <1>       ADD    AX, 2000H-40        ; OFFSET TO LOCATION OF ODD ROWS ADJUST
                                <1> R4:                             ; EVEN_ROW
000025F6 6696                   <1>       XCHG   SI, AX              ; MOVE POINTER TO (SI) AND RECOVER (AX)
000025F8 81C600800B00           <1>       add    esi, 0B8000h
000025FE 6689CA                 <1>       MOV    DX, CX              ; COLUMN VALUE TO DX
                                <1>
                                <1> ;-----      DETERMINE GRAPHICS MODE CURRENTLY IN EFFECT
                                <1>
                                <1> ; SET UP THE REGISTERS ACCORDING TO THE MODE
                                <1> ; CH = MASK FOR LOW OF COLUMN ADDRESS ( 7/3 FOR HIGH/MED RES )
                                <1> ; CL = # OF ADDRESS BITS IN COLUMN VALUE ( 3/2 FOR H/M )
                                <1> ; BL = MASK TO SELECT BITS FROM POINTED BYTE ( 80H/C0H FOR H/M )
                                <1> ; BH = NUMBER OF VALID BITS IN POINTED BYTE ( 1/2 FOR H/M )
                                <1>
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[60210100]           <1>       mov    esi, palette0
00002654 EB15                   <1>       jmp    short l_dac_p_4
                                <1> l_dac_p_1:
00002656 BE[20220100]           <1>       mov    esi, palette1
0000265B EB0E                   <1>       jmp    short l_dac_p_4
```

```
                                <1> l_dac_p_2:
0000265D BE[E0220100]           <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[A0230100]           <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
```

```
7827 000026E4 EC                <1>       in     al, dx
7828 000026E5 B020              <1>       mov    al, 20h
7829 000026E7 66BAC003          <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
7830 000026EB EE                <1>       out    dx, al ; 20h -> set bit 5
7831                            <1>                       ; (after loading palette regs)
7832 000026EC C3                <1>       retn
7833                            <1>
7834                            <1> vga_write_char_attr:
7835                            <1> vga_write_char_only:
7836                            <1>       ; 08/07/2016 (TRDOS 386 = TRDOS v2.0)
7837                            <1>       ;
7838                            <1>       ; derived from 'Plex86/Bochs VGABios' source code
7839                            <1>       ; vgabios-0.7a (2011)
7840                            <1>       ; by the LGPL VGABios developers Team (2001-2008)
7841                            <1>       ; 'vgabios.c', 'biosfn_write_char_attr'
7842                            <1>       ; 'biosfn_write_char_only'
7843                            <1>
7844                            <1>       ; INPUT ->
7845                            <1>       ; [CRT_MODE] = current video mode (>7)
7846                            <1>       ; CX = Count of characters to write
7847                            <1>       ; AL = Character to write
7848                            <1>       ; BL = Color of character
7849                            <1>       ; OUTPUT ->
7850                            <1>       ; Regen buffer updated
7851                            <1>
7852 000026ED 8A25[C25E0000]    <1>       mov    ah, [CRT_MODE]
7853 000026F3 668B15[3E520100]  <1>       mov    dx, [CURSOR_POSN] ; cursor pos for page 0
7854                            <1>
7855 000026FA BE[DE5E0000]      <1>       mov    esi, vga_modes
7856 000026FF 89F7              <1>       mov    edi, esi
7857 00002701 83C710            <1>       add    edi, vga_mode_count
7858                            <1> vga_wca_0:
7859 00002704 AC                <1>       lodsb
7860 00002705 38E0              <1>       cmp    al, ah ; [CRT_MODE]
7861 00002707 7405              <1>       je     short vga_wca_2
7862 00002709 39FE              <1>       cmp    esi, edi
7863 0000270B 72F7              <1>       jb     short vga_wca_0
7864                            <1> vga_wca_1:
7865 0000270D C3                <1>       retn   ; nothing to do
7866                            <1> vga_wca_2:
7867 0000270E 83C64F            <1>       add    esi, vga_memmodel - (vga_modes + 1)
7868                            <1>       ; [ESI] = VGA memory model number (LINEAR8, PLANAR4, PLANAR1)
7869                            <1>
7870                            <1>       ; biosfn_write_char_attr (car,page,attr,count)
7871                            <1>       ; AL = car, page = 0, BL = attr, CX = count
7872 00002711 803E04            <1>       cmp    byte [esi], PLANAR4
7873 00002714 741D              <1>       je     short vga_wca_planar
7874 00002716 803E03            <1>       cmp    byte [esi], PLANAR1
7875 00002719 7418              <1>       je     short vga_wca_planar
7876                            <1> vga_wca_linear8:
7877                            <1>       ; while((count-->0) && (xcurs<nbcols))
7878                            <1>       ; CX = count
7879 0000271B 6621C9            <1>       and    cx, cx
7880 0000271E 74ED              <1>       jz     short vga_wca_1
7881 00002720 3A15[C45E0000]    <1>       cmp    dl, [CRT_COLS]
7882 00002726 73E5              <1>       jnb    short vga_wca_1
7883                            <1>       ; write_gfx_char_lin(car,attr,xcurs,ycurs,nbcols);
7884                            <1>       ; AL = car, BL = attr, DL = xcurs, DH = ycurs,
7885                            <1>       ; [CRT_COLS] = nbcols
7886 00002728 E81E000000        <1>       call   write_gfx_char_lin
7887 0000272D 6649              <1>       dec    cx ; count
7888 0000272F FEC2              <1>       inc    dl ; xcurs
7889 00002731 EBE8              <1>       jmp    short vga_wca_linear8
7890                            <1> vga_wca_planar:
7891                            <1>       ; while((count-->0) && (xcurs<nbcols))
7892                            <1>       ; CX = count
7893 00002733 6621C9            <1>       and    cx, cx
7894 00002736 74D5              <1>       jz     short vga_wca_1
7895 00002738 3A15[C45E0000]    <1>       cmp    dl, [CRT_COLS]
7896 0000273E 73CD              <1>       jnb    short vga_wca_1
7897                            <1>       ; write_gfx_char_pl4(car,attr,xcurs,ycurs,nbcols,cheight);
7898                            <1>       ; AL = car, BL = attr, DL = xcurs, DH = ycurs,
7899                            <1>       ; [CRT_COLS] = nbcols, [CHAR_HEIGHT] = cheight
7900 00002740 E89D000000        <1>       call   write_gfx_char_pl4
7901 00002745 6649              <1>       dec    cx ; count
7902 00002747 FEC2              <1>       inc    dl ; xcurs
7903 00002749 EBE8              <1>       jmp    short vga_wca_planar
7904                            <1>
7905                            <1> write_gfx_char_lin:
7906                            <1>       ; 08/08/2016
7907                            <1>       ; 31/07/2016
7908                            <1>       ; 08/07/2016 (TRDOS 386 = TRDOS v2.0)
7909                            <1>       ;
7910                            <1>       ; derived from 'Plex86/Bochs VGABios' source code
7911                            <1>       ; vgabios-0.7a (2011)
7912                            <1>       ; by the LGPL VGABios developers Team (2001-2008)
7913                            <1>       ; 'vgabios.c', 'write_gfx_char_lin'
7914                            <1>
7915                            <1>       ; write_gfx_char_lin(car,attr,xcurs,ycurs,nbcols)
7916                            <1>       ; INPUT ->
7917                            <1>       ; AL = car, BL = attr, DL = xcurs, DH = ycurs,
7918                            <1>       ; [CRT_COLS] = nbcols
7919                            <1>       ; OUTPUT ->
7920                            <1>       ; Regen buffer updated
7921                            <1>
7922 0000274B 51                <1>       push   ecx
7923 0000274C 53                <1>       push   ebx
7924 0000274D 52                <1>       push   edx
7925 0000274E 50                <1>       push   eax
7926                            <1>       ; addr=xcurs*8+ycurs*nbcols*64;
7927                            <1>       ; 08/08/2016
7928 0000274F 0FB6F0            <1>       movzx  esi, al ; car
7929 00002752 0FB6C6            <1>       movzx  eax, dh ; ycurs
```

```
7930 00002755 8A25[C45E0000]      <1>       mov    ah, [CRT_COLS] ; nbcols
7931 0000275B F6E4                <1>       mul    ah
7932                              <1>       ;shl   ax, 6 ; * 64
7933 0000275D 66C1E003            <1>       shl    ax, 3 ; * 8
7934                              <1>       ;sub   dh, dh
7935                              <1>       ;shl   dx, 3 ; xcurs * 8
7936                              <1>       ;movzx edi, dx
7937 00002761 0FB6FA              <1>       movzx  edi, dl
7938 00002764 66C1E703            <1>       shl    di, 3 ; xcurs * 8
7939 00002768 30F6                <1>       xor    dh, dh
7940 0000276A 8A15[C65E0000]      <1>       mov    dl, [CHAR_HEIGHT]
7941 00002770 66F7E2              <1>       mul    dx
7942                              <1>       ; eax = ycurs*nbcols*8*[CHAR_HEIGHT]
7943 00002773 01C7                <1>       add    edi, eax ; addr
7944 00002775 81C700000A00        <1>       add    edi, 0A0000h
7945                              <1>       ;shl   si, 3 ; car * 8
7946 0000277B 30E4                <1>       xor    ah, ah
7947 0000277D A0[C65E0000]        <1>       mov    al, [CHAR_HEIGHT]
7948 00002782 66F7E6              <1>       mul    si
7949 00002785 6689C6              <1>       mov    si, ax
7950                              <1>       ;; esi = src = car * 8
7951                              <1>       ; esi = src = car * [CHAR_HEIGHT]
7952                              <1>       ; i = 0
7953                              <1>       ;add   esi, vgafont8 ; fdata [src+i]
7954                              <1>       ; 08/08/2016
7955 00002788 A1[CE5F0100]        <1>       mov    eax, [VGA_INT43H]
7956 0000278D 3D[A03C0100]        <1>       cmp    eax, vgafont16
7957 00002792 740F                <1>         je     short wgfxl_0
7958 00002794 3D[A02E0100]        <1>       cmp    eax, vgafont14
7959 00002799 7408                <1>       je     short wgfxl_0
7960 0000279B 81C6[A0260100]      <1>       add    esi, vgafont8
7961 000027A1 EB02                <1>       jmp    short wgfxl_1
7962                              <1> wgfxl_0:
7963 000027A3 01C6                <1>       add    esi, eax
7964                              <1> wgfxl_1:
7965 000027A5 28FF                <1>       sub    bh, bh ; i = 0
7966                              <1> wgfxl_2:
7967                              <1>       ; for(i=0;i<8;i++)
7968 000027A7 57                  <1>       push   edi ; addr
7969 000027A8 0FB605[C45E0000]    <1>       movzx  eax, byte [CRT_COLS] ; nbcols
7970 000027AF F6E7                <1>       mul    bh ; nbcols*i
7971 000027B1 66C1E003            <1>       shl    ax, 3 ; i*nbcols*8
7972                              <1>       ; dest=addr+i*nbcols*8;
7973 000027B5 01C7                <1>       add    edi, eax ; dest + j ; j = 0
7974 000027B7 B180                <1>       mov    cl, 80h ; mask = 0x80;
7975                              <1>       ; esi = fdata + src + i
7976                              <1>       ; for(j=0;j<8;j++)
7977 000027B9 29D2                <1>       sub    edx, edx ; j = 0
7978                              <1> wgfxl_3:
7979 000027BB 8A06                <1>       mov    al, [esi] ; al = fdata[src+i]
7980 000027BD 20C8                <1>       and    al, cl ; if (fdata[src+i] & mask)
7981 000027BF 7402                <1>       jz     short wgfxl_4  ; data = 0, zf = 1
7982 000027C1 88D8                <1>       mov    al, bl ; data = attr;
7983                              <1> wgfxl_4:
7984                              <1>       ; write_byte(0xa000,dest+j,data);
7985 000027C3 AA                  <1>       stosb  ; dest + j (+ 0A0000h)
7986                              <1>       ;inc   dl ; j++
7987                              <1>       ;cmp   dl, 8
7988 000027C4 80FA07              <1>       cmp    dl, 7
7989 000027C7 720E                <1>       jb     short wgfxl_5
7990 000027C9 5F                  <1>       pop    edi
7991                              <1>       ; 08/08/2016
7992                              <1>       ;cmp   bh, 7
7993                              <1>       ;jnb   short wgfxl_6
7994 000027CA FEC7                <1>       inc    bh ; i++
7995 000027CC 3A3D[C65E0000]      <1>       cmp    bh, [CHAR_HEIGHT]
7996 000027D2 7309                <1>       jnb    short wgfxl_6
7997 000027D4 46                  <1>       inc    esi
7998 000027D5 EBD0                <1>       jmp    short wgfxl_2
7999                              <1> wgfxl_5:
8000 000027D7 D0E9                <1>       shr    cl, 1 ; mask >>= 1;
8001 000027D9 FEC2                <1>       inc    dl ; j++
8002 000027DB EBDE                <1>         jmp    short wgfxl_3
8003                              <1> wgfxl_6:
8004 000027DD 58                  <1>       pop    eax
8005 000027DE 5A                  <1>       pop    edx
8006 000027DF 5B                  <1>       pop    ebx
8007 000027E0 59                  <1>       pop    ecx
8008 000027E1 C3                  <1>       retn
8009                              <1>
8010                              <1> write_gfx_char_pl4:
8011                              <1>       ; 08/08/2016
8012                              <1>       ; 08/07/2016 (TRDOS 386 = TRDOS v2.0)
8013                              <1>       ;
8014                              <1>       ; derived from 'Plex86/Bochs VGABios' source code
8015                              <1>       ; vgabios-0.7a (2011)
8016                              <1>       ; by the LGPL VGABios developers Team (2001-2008)
8017                              <1>       ; 'vgabios.c', 'write_gfx_char_pl4'
8018                              <1>
8019                              <1>       ; write_gfx_char_pl4(car,attr,xcurs,ycurs,nbcols,cheight)
8020                              <1>       ; INPUT ->
8021                              <1>       ; AL = car, BL = attr, DL = xcurs, DH = ycurs,
8022                              <1>       ; [CRT_COLS] = nbcols, [CHAR_HEIGHT] = cheight
8023                              <1>       ; OUTPUT ->
8024                              <1>       ; Regen buffer updated
8025                              <1>
8026 000027E2 51                  <1>       push   ecx
8027 000027E3 53                  <1>       push   ebx
8028 000027E4 52                  <1>       push   edx
8029 000027E5 50                  <1>       push   eax
8030                              <1> wgfxpl_f0:
8031                              <1>       ; switch(cheight)
8032 000027E6 8A25[C65E0000]      <1>       mov    ah, [CHAR_HEIGHT]
```

```
8033 000027EC 80FC10              <1>       cmp   ah, 16 ; case 16:
8034 000027EF 7507                <1>       jne   short wgfxpl_f1
8035                              <1>       ; fdata = &vgafont16;
8036 000027F1 BE[A03C0100]        <1>       mov   esi, vgafont16
8037 000027F6 EB13                <1>       jmp   short wgfxpl_f3
8038                              <1> wgfxpl_f1:
8039 000027F8 80FC0E              <1>       cmp   ah, 14 ; case 14:
8040 000027FB 7507                <1>       jne   short wgfxpl_f2
8041 000027FD BE[A02E0100]        <1>       mov   esi, vgafont14
8042 00002802 EB07                <1>       jmp   short wgfxpl_f3
8043                              <1> wgfxpl_f2:
8044                              <1>       ; default:
8045                              <1>       ;  fdata = &vgafont8;
8046 00002804 BE[A0260100]        <1>       mov   esi, vgafont8
8047 00002809 B408                <1>       mov   ah, 8
8048                              <1> wgfxpl_f3:
8049                              <1>       ; al = car
8050 0000280B F6E4                <1>       mul   ah ; ah = cheight
8051 0000280D 25FFFF0000          <1>       and   eax, 0FFFFh ; car * cheight
8052                              <1>       ; src = car * cheight;
8053 00002812 01C6                <1>       add   esi, eax ; esi = fdata[src+i]
8054                              <1>       ; addr=xcurs*8+ycurs*nbcols*64;
8055 00002814 88F0                <1>       mov   al, dh ; ycurs
8056 00002816 8A25[C45E0000]      <1>       mov   ah, [CRT_COLS] ; nbcols
8057 0000281C F6E4                <1>       mul   ah
8058                              <1>       ; 08/08/2016
8059                              <1>       ;shl   ax, 6 ; * 64
8060 0000281E 66C1E003            <1>       shl   ax, 3 ; * 8
8061                              <1>       ;sub   dh, dh ; 0
8062                              <1>       ;shl   dx, 3 ; xcurs * 8
8063                              <1>       ;movzx edi, dx
8064 00002822 0FB6FA              <1>       movzx  edi, dl
8065 00002825 66C1E703            <1>       shl   di, 3 ; xcurs * 8
8066 00002829 30F6                <1>       xor   dh, dh
8067 0000282B 8A15[C65E0000]      <1>       mov   dl, [CHAR_HEIGHT]
8068 00002831 66F7E2              <1>       mul   dx
8069                              <1>       ; eax = ycurs*nbcols*8*[CHAR_HEIGHT]
8070 00002834 01C7                <1>       add   edi, eax ; addr
8071 00002836 81C700000A00        <1>       add   edi, 0A0000h
8072                              <1>       ;
8073                              <1>       ; outw(VGAREG_SEQU_ADDRESS, 0x0f02);
8074                              <1>       ; outw(VGAREG_GRDC_ADDRESS, 0x0205);
8075 0000283C 66BAC403            <1>       mov   dx, 3C4h ; VGAREG_SEQU_ADDRESS
8076 00002840 66B8020F            <1>       mov   ax, 0F02h
8077 00002844 66EF                <1>       out   dx, ax
8078 00002846 66BACE03            <1>       mov   dx, 3CEh ; VGAREG_GRDC_ADDRESS
8079 0000284A 66B80502            <1>       mov   ax, 0205h
8080 0000284E 66EF                <1>       out   dx, ax
8081                              <1>       ;
8082 00002850 66BACE03            <1>       mov   dx, 3CEh ; VGAREG_GRDC_ADDRESS
8083 00002854 F6C380              <1>       test  bl, 80h ; if(attr&0x80)
8084 00002857 7406                <1>       jz    short wgfxpl_f4 ; else
8085                              <1>       ; outw(VGAREG_GRDC_ADDRESS, 0x1803);
8086 00002859 66B80318            <1>       mov   ax, 1803h
8087 0000285D EB04                <1>       jmp   short wgfxpl_f5
8088                              <1> wgfxpl_f4:
8089                              <1>       ; outw(VGAREG_GRDC_ADDRESS, 0x0003);
8090 0000285F 66B80300            <1>       mov   ax, 0003h
8091                              <1> wgfxpl_f5:
8092 00002863 66EF                <1>       out   dx, ax
8093                              <1>       ;
8094 00002865 28FF                <1>       sub   bh, bh ; i = 0
8095                              <1> wgfxpl_0:
8096                              <1>       ; for(i=0;i<cheight;i++)
8097 00002867 57                  <1>       push  edi ; addr
8098 00002868 0FB605[C45E0000]    <1>       movzx  eax, byte [CRT_COLS] ; nbcols
8099 0000286F F6E7                <1>       mul   bh ; nbcols*i
8100                              <1>       ; dest=addr+i*nbcols
8101 00002871 01C7                <1>       add   edi, eax ; dest
8102 00002873 B580                <1>       mov   ch, 80h ; mask = 0x80;
8103                              <1>       ; for(j=0;j<8;j++)
8104 00002875 28C9                <1>       sub   cl, cl ; j = 0
8105                              <1> wgfxpl_1:
8106 00002877 D2ED                <1>       shr   ch, cl ; mask=0x80>>j;
8107                              <1>       ;
8108                              <1>       ; outw(VGAREG_GRDC_ADDRESS, (mask << 8) | 0x08);
8109                              <1>       ; read_byte(0xa000,dest);
8110                              <1>       ;mov   dx, 3CEh ; VGAREG_GRDC_ADDRESS
8111 00002879 88EC                <1>       mov   ah, ch
8112 0000287B B008                <1>       mov   al, 8
8113 0000287D 66EF                <1>       out   dx, ax
8114 0000287F 8A07                <1>       mov   al, [edi] ; ? (io delay?)
8115                              <1>       ;
8116 00002881 28C0                <1>       sub   al, al ; attr = 0
8117                              <1>       ; if (fdata[src+i] & mask)
8118 00002883 842E                <1>       test  byte [esi], ch
8119 00002885 7404                <1>       jz    short wgfxpl_2  ; zf = 1
8120                              <1>       ; write_byte(0xa000,dest,attr&0x0f);
8121 00002887 88D8                <1>       mov   al, bl ; attr;
8122 00002889 240F                <1>       and   al, 0Fh      ; attr&0x0f
8123                              <1> wgfxpl_2:
8124                              <1>       ; write_byte(0xa000,dest,0x00);
8125 0000288B 8807                <1>       mov   [edi], al ; dest (+ 0A0000h)
8126 0000288D FEC1                <1>       inc   cl ; j++
8127 0000288F 80F908              <1>       cmp   cl, 8
8128 00002892 72E3                <1>       jb    short wgfxpl_1
8129 00002894 5F                  <1>       pop   edi
8130                              <1>       ; 08/08/2016
8131                              <1>       ;cmp   bh, 7
8132                              <1>       ;jnb   short wgfxpl_3
8133 00002895 FEC7                <1>       inc   bh ; i++
8134 00002897 3A3D[C65E0000]      <1>       cmp   bh, [CHAR_HEIGHT]
8135 0000289D 7303                <1>       jnb   short wgfxpl_3
```

```
8136 0000289F 46                <1>       inc    esi
8137 000028A0 EBC5              <1>       jmp    short wgfxpl_0
8138                            <1> wgfxpl_3:
8139                            <1>       ;mov   dx, 3CEh ; VGAREG_GRDC_ADDRESS
8140 000028A2 66B808FF          <1>       mov    ax, 0FF08h
8141 000028A6 66EF              <1>       out    dx, ax
8142 000028A8 66B80500          <1>       mov    ax, 0005h
8143 000028AC 66EF              <1>       out    dx, ax
8144 000028AE 66B80300          <1>       mov    ax, 0003h
8145 000028B2 66EF              <1>       out    dx, ax
8146                            <1>       ;
8147 000028B4 58                <1>       pop    eax
8148 000028B5 5A                <1>       pop    edx
8149 000028B6 5B                <1>       pop    ebx
8150 000028B7 59                <1>       pop    ecx
8151 000028B8 C3                <1>       retn
8152                            <1>
8153                            <1> vga_write_pixel:
8154                            <1>       ; 09/07/2016 (TRDOS 386 = TRDOS v2.0)
8155                            <1>       ;
8156                            <1>       ; derived from 'Plex86/Bochs VGABios' source code
8157                            <1>       ; vgabios-0.7a (2011)
8158                            <1>       ; by the LGPL VGABios developers Team (2001-2008)
8159                            <1>       ; 'vgabios.c', 'biosfn_write_pixel'
8160                            <1>
8161                            <1>       ; INPUT ->
8162                            <1>       ;     DX = row (0-239)
8163                            <1>       ;     CX = column (0-799)
8164                            <1>       ;     AL = pixel value
8165                            <1>       ;     (AH = [CRT_MODE])
8166                            <1>       ; OUTPUT ->
8167                            <1>       ;     none
8168                            <1>
8169 000028B9 88C3              <1>       mov    bl, al ; pixel value
8170                            <1>       ;mov   ah, [CRT_MODE]
8171 000028BB BE[DE5E0000]      <1>       mov    esi, vga_modes
8172 000028C0 89F7              <1>       mov    edi, esi
8173 000028C2 83C710            <1>       add    edi, vga_mode_count
8174                            <1> vga_wp_0:
8175 000028C5 AC                <1>       lodsb
8176 000028C6 38E0              <1>       cmp    al, ah ; [CRT_MODE]
8177 000028C8 7405              <1>       je     short vga_wp_1
8178 000028CA 39FE              <1>       cmp    esi, edi
8179 000028CC 72F7              <1>       jb     short vga_wp_0
8180 000028CE C3                <1>       retn   ; nothing to do
8181                            <1> vga_wp_1:
8182 000028CF 83C64F            <1>       add    esi, vga_memmodel - (vga_modes + 1)
8183                            <1>       ; [ESI] = VGA memory model number (LINEAR8, PLANAR4, PLANAR1)
8184 000028D2 BF00000A00        <1>       mov    edi, 0A0000h
8185                            <1>       ;
8186 000028D7 803E04            <1>       cmp    byte [esi], PLANAR4
8187 000028DA 741D              <1>       je     short vga_wp_planar
8188 000028DC 803E03            <1>       cmp    byte [esi], PLANAR1
8189 000028DF 7418              <1>       je     short vga_wp_planar
8190                            <1> vga_wp_linear8:
8191                            <1>       ; addr=CX+DX*(read_word(BIOSMEM_SEG,BIOSMEM_NB_COLS)*8);
8192 000028E1 0FB605[C45E0000]  <1>       movzx  eax, byte [CRT_COLS] ; = [VGA_COLS] ; nbcols
8193 000028E8 66C1E003          <1>       shl    ax, 3 ; * 8
8194 000028EC 66F7E2            <1>       mul    dx
8195 000028EF 50                <1>       push   eax
8196                            <1>       ;mov   edi, 0A0000h
8197 000028F0 6601CF            <1>       add    di, cx
8198 000028F3 58                <1>       pop    eax
8199 000028F4 01C7              <1>       add    edi, eax ; addr
8200                            <1>       ; write_byte(0xa000,addr,AL);
8201 000028F6 881F              <1>       mov    [edi], bl
8202 000028F8 C3                <1>       retn
8203                            <1> vga_wp_planar:
8204                            <1>       ; addr = CX/8+DX*read_word(BIOSMEM_SEG,BIOSMEM_NB_COLS);
8205 000028F9 0FB7C1            <1>       movzx  eax, cx
8206 000028FC 66C1E803          <1>       shr    ax, 3 ; CX/8
8207 00002900 50                <1>       push   eax
8208 00002901 28E4              <1>       sub    ah, ah ; 0
8209 00002903 A0[C45E0000]      <1>       mov    al, [CRT_COLS] ; = [VGA_COLS] ; nbcols
8210 00002908 66F7E2            <1>       mul    dx
8211                            <1>       ;mov   edi, 0A0000h
8212 0000290B 6601C7            <1>         add     di, ax
8213 0000290E 58                <1>       pop    eax
8214 0000290F 01C7              <1>       add    edi, eax ; addr
8215 00002911 80E107            <1>       and    cl, 7
8216 00002914 B580              <1>       mov    ch, 80h ; mask
8217 00002916 D2ED              <1>       shr    ch, cl       ; mask = 0x80 >> (CX & 0x07);
8218                            <1>
8219                            <1>       ; outw(VGAREG_GRDC_ADDRESS, (mask << 8) | 0x08);
8220 00002918 66BACE03          <1>       mov    dx, 3CEh ; VGAREG_GRDC_ADDRESS
8221 0000291C 88EC              <1>       mov    ah, ch
8222 0000291E B008              <1>       mov    al, 8
8223 00002920 66EF              <1>       out    dx, ax
8224                            <1>       ; outw(VGAREG_GRDC_ADDRESS, 0x0205);
8225 00002922 66B80502          <1>       mov    ax, 0205h
8226 00002926 66EF              <1>       out    dx, ax
8227                            <1>       ; data = read_byte(0xa000,addr);
8228 00002928 8A07              <1>       mov    al, [edi] ; (delay?)
8229                            <1>       ; if (AL & 0x80)
8230                            <1>       ; {
8231                            <1>       ;  outw(VGAREG_GRDC_ADDRESS, 0x1803);
8232                            <1>       ; }
8233 0000292A F6C380            <1>       test   bl, 80h
8234 0000292D 7406              <1>       jz     short vga_wp_2
8235 0000292F 66B80318          <1>       mov    ax, 1803h
8236 00002933 66EF              <1>       out    dx, ax
8237                            <1> vga_wp_2:
8238                            <1>       ; write_byte(0xa000,addr,AL);
```

```
8239 00002935 881F                <1>       mov   [edi], bl
8240                              <1>       ;
8241                              <1>       ;mov   dx, 3CEh ; VGAREG_GRDC_ADDRESS
8242 00002937 66B808FF            <1>       mov   ax, 0FF08h
8243 0000293B 66EF                <1>       out   dx, ax
8244 0000293D 66B80500            <1>       mov   ax, 0005h
8245 00002941 66EF                <1>       out   dx, ax
8246 00002943 66B80300            <1>       mov   ax, 0003h
8247 00002947 66EF                <1>       out   dx, ax
8248                              <1>       ;
8249 00002949 C3                  <1>       retn
8250                              <1>
8251                              <1> vga_read_pixel:
8252                              <1>       ; 09/07/2016 (TRDOS 386 = TRDOS v2.0)
8253                              <1>       ;
8254                              <1>       ; derived from 'Plex86/Bochs VGABios' source code
8255                              <1>       ; vgabios-0.7a (2011)
8256                              <1>       ; by the LGPL VGABios developers Team (2001-2008)
8257                              <1>       ; 'vgabios.c', 'biosfn_read_pixel'
8258                              <1>
8259                              <1>       ; INPUT ->
8260                              <1>       ;     DX = row (0-239)
8261                              <1>       ;     CX = column (0-799)
8262                              <1>       ;     (AH = [CRT_MODE])
8263                              <1>       ; OUTPUT ->
8264                              <1>       ;     AL = pixel value
8265                              <1>
8266                              <1>       ;mov   ah, [CRT_MODE]
8267 0000294A BE[DE5E0000]        <1>       mov   esi, vga_modes
8268 0000294F 89F7                <1>       mov   edi, esi
8269 00002951 83C710              <1>       add   edi, vga_mode_count
8270                              <1> vga_rp_0:
8271 00002954 AC                  <1>       lodsb
8272 00002955 38E0                <1>       cmp   al, ah ; [CRT_MODE]
8273 00002957 7405                <1>       je    short vga_rp_1
8274 00002959 39FE                <1>       cmp   esi, edi
8275 0000295B 72F7                <1>       jb    short vga_rp_0
8276 0000295D C3                  <1>       retn  ; nothing to do
8277                              <1> vga_rp_1:
8278 0000295E 83C64F              <1>       add   esi, vga_memmodel - (vga_modes + 1)
8279                              <1>       ; [ESI] = VGA memory model number (LINEAR8, PLANAR4, PLANAR1)
8280 00002961 BF00000A00          <1>       mov   edi, 0A0000h
8281                              <1>       ;
8282 00002966 803E04              <1>       cmp   byte [esi], PLANAR4
8283 00002969 741D                <1>       je    short vga_rp_planar
8284 0000296B 803E03              <1>       cmp   byte [esi], PLANAR1
8285 0000296E 7418                <1>       je    short vga_rp_planar
8286                              <1> vga_rp_linear8:
8287                              <1>       ; addr=CX+DX*(read_word(BIOSMEM_SEG,BIOSMEM_NB_COLS)*8);
8288 00002970 0FB605[C45E0000]    <1>       movzx  eax, byte [CRT_COLS] ; = [VGA_COLS] ; nbcols
8289 00002977 66C1E003            <1>       shl   ax, 3 ; * 8
8290 0000297B 66F7E2              <1>       mul   dx
8291 0000297E 50                  <1>       push  eax
8292                              <1>       ;mov   edi, 0A0000h
8293 0000297F 6601CF              <1>       add   di, cx
8294 00002982 58                  <1>       pop   eax
8295 00002983 01C7                <1>       add   edi, eax ; addr
8296                              <1>       ; attr=read_byte(0xa000,addr);
8297 00002985 8A07                <1>       mov   al, [edi] ; pixel value
8298 00002987 C3                  <1>       retn
8299                              <1> vga_rp_planar:
8300                              <1>       ; addr = CX/8+DX*read_word(BIOSMEM_SEG,BIOSMEM_NB_COLS);
8301 00002988 0FB7C1              <1>       movzx  eax, cx
8302 0000298B 66C1E803            <1>       shr   ax, 3 ; CX/8
8303 0000298F 50                  <1>       push  eax
8304 00002990 28E4                <1>       sub   ah, ah ; 0
8305 00002992 A0[C45E0000]        <1>       mov   al, [CRT_COLS] ; = [VGA_COLS] ; nbcols
8306 00002997 66F7E2              <1>       mul   dx
8307                              <1>       ;mov   edi, 0A0000h
8308 0000299A 6601C7              <1>         add     di, ax
8309 0000299D 58                  <1>       pop   eax
8310 0000299E 01C7                <1>       add   edi, eax ; addr
8311 000029A0 80E107              <1>       and   cl, 7
8312 000029A3 B580                <1>       mov   ch, 80h ; mask
8313 000029A5 D2ED                <1>       shr   ch, cl       ; mask = 0x80 >> (CX & 0x07);
8314                              <1>       ; attr = 0x00;
8315 000029A7 30DB                <1>       xor   bl, bl ; attr = bl = 0,
8316 000029A9 30C9                <1>       xor   cl, cl ; i = cl = 0
8317                              <1>       ; for(i=0;i<4;i++)
8318                              <1>             ; {
8319                              <1>             ;  outw(VGAREG_GRDC_ADDRESS, (i << 8) | 0x04);
8320                              <1>             ;  data = read_byte(0xa000,addr) & mask;
8321                              <1>             ;  if (data > 0) attr |= (0x01 << i);
8322                              <1>             ; }
8323                              <1> vga_rp_2:
8324 000029AB 88CC                <1>       mov   ah, cl ; i << 8
8325 000029AD B004                <1>       mov   al, 4  ; | 0x04
8326 000029AF 66BACE03            <1>       mov   dx, 3CEh ; VGAREG_GRDC_ADDRESS
8327 000029B3 66EF                <1>       out   dx, ax
8328                              <1>       ; data = read_byte(0xa000,addr) & mask;
8329 000029B5 8A07                <1>       mov   al, [edi]
8330 000029B7 20E8                <1>       and   al, ch ; & mask
8331                              <1>       ; if (data > 0) attr |= (0x01 << i);
8332 000029B9 08C0                <1>       or    al, al
8333 000029BB 7408                <1>       jz    short vga_rp_3 ; al = 0
8334 000029BD B701                <1>       mov   bh, 1
8335 000029BF D2E7                <1>       shl   bh, cl ; (0x01 << i)
8336 000029C1 08FB                <1>       or    bl, bh ; attr |= (0x01 << i)
8337 000029C3 88D8                <1>       mov   al, bl ; pixel value
8338                              <1> vga_rp_3:
8339 000029C5 C3                  <1>       retn
8340                              <1>
8341                              <1> vga_beeper:
```

```
                                <1>       ; 04/08/2016  (TRDOS 386 = TRDOS v2.0)
000029C6 FB                     <1>       sti
                                <1>       ;mov   bh, [ACTIVE_PAGE]
000029C7 E9CFF3FFFF             <1>         jmp     beeper_gfx
                                <1>
                                <1> vga_write_teletype:
                                <1>       ; 09/12/2017
                                <1>       ; 06/08/2016
                                <1>       ; 04/08/2016
                                <1>       ; 01/08/2016
                                <1>       ; 31/07/2016
                                <1>       ; 09/07/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>       ;
                                <1>       ; derived from 'Plex86/Bochs VGABios' source code
                                <1>       ; vgabios-0.7a (2011)
                                <1>       ; by the LGPL VGABios developers Team (2001-2008)
                                <1>       ; 'vgabios.c', 'biosfn_write_teletype'
                                <1>       ; 'biosfn_write_char_only'
                                <1>
                                <1>       ; INPUT ->
                                <1>       ; [CRT_MODE] = current video mode (>7)
                                <1>       ; AL = Character to write
                                <1>       ; BL = Color of character
                                <1>       ; OUTPUT ->
                                <1>       ; Regen buffer updated
                                <1>
                                <1>       ; biosfn_write_teletype (car, page, attr, flag)
                                <1>       ; car = character (AL)
                                <1>       ; page = 0
                                <1>       ; attr = color (BL)
                                <1>       ; 'flag' not used
                                <1>
000029CC 8A25[C25E0000]         <1>       mov   ah, [CRT_MODE]
000029D2 88C7                   <1>       mov   bh, al ; character
000029D4 668B15[3E520100]       <1>       mov   dx, [CURSOR_POSN] ; cursor pos for page 0
                                <1>
000029DB BE[E65E0000]           <1>       mov   esi, vga_g_modes
000029E0 89F7                   <1>       mov   edi, esi
000029E2 83C708                 <1>       add   edi, vga_g_mode_count
                                <1> vga_wtty_0:
000029E5 AC                     <1>       lodsb
000029E6 38E0                   <1>       cmp   al, ah ; [CRT_MODE]
000029E8 7405                   <1>       je    short vga_wtty_2
000029EA 39FE                   <1>       cmp   esi, edi
000029EC 72F7                   <1>       jb    short vga_wtty_0
                                <1> vga_wtty_1:
000029EE C3                     <1>       retn  ; nothing to do
                                <1> vga_wtty_2:
000029EF 80FF07                 <1>       cmp   bh, 07h ; bell (beep)
000029F2 74D2                   <1>       je    short vga_beeper  ; u11
000029F4 80FF08                 <1>       cmp   bh, 08h ; backspace
000029F7 7508                   <1>       jne   short vga_wtty_3
                                <1>       ; if(xcurs>0)xcurs--;
000029F9 08D2                   <1>       or    dl, dl ; xcurs (column)
000029FB 74F1                   <1>       jz    short vga_wtty_1
000029FD FECA                   <1>       dec   dl ; xcurs--;
000029FF EB59                   <1>         jmp     short vga_wtty_12
                                <1> vga_wtty_3:
00002A01 80FF0D                 <1>       cmp   bh, 0Dh ; carriage return (\r)
00002A04 7504                   <1>       jne   short vga_wtty_4
                                <1>       ; xcurs=0;
00002A06 28D2                   <1>       sub   dl, dl ; 0
00002A08 EB50                   <1>         jmp     short vga_wtty_12
                                <1> vga_wtty_4:
00002A0A 80FF0A                 <1>       cmp   bh, 0Ah ; new line (\n)
00002A0D 7504                   <1>       jne   short vga_wtty_5
                                <1>       ; ycurs++;
00002A0F FEC6                   <1>       inc   dh ; next row
00002A11 EB62                   <1>         jmp     short vga_wtty_11
                                <1> vga_wtty_5:
00002A13 80FF09                 <1>       cmp   bh, 09h ; tab stop
00002A16 7527                   <1>       jne   short vga_wtty_8
00002A18 88D0                   <1>       mov   al, dl
                                <1>       ;cbw
00002A1A 30E4                   <1>       xor   ah, ah ; 09/12/2017
00002A1C B108                   <1>       mov   cl, 8
00002A1E F6F1                   <1>       div   cl
00002A20 28E1                   <1>       sub   cl, ah
                                <1>       ;
00002A22 B720                   <1>       mov   bh, 20h ; space
                                <1> vga_wtty_6: ; tab stop loop
00002A24 6651                   <1>       push  cx
00002A26 6653                   <1>       push  bx
00002A28 E812000000             <1>       call  vga_wtty_8
00002A2D 665B                   <1>       pop   bx  ; bh = character, bl = color
00002A2F 6659                   <1>       pop   cx
00002A31 FEC9                   <1>       dec   cl
00002A33 7409                   <1>       jz    short vga_wtty_7
00002A35 668B15[3E520100]       <1>       mov   dx, [CURSOR_POSN] ; new cursor position (pg 0)
00002A3C EBE6                   <1>       jmp   short vga_wtty_6
                                <1> vga_wtty_7:
00002A3E C3                     <1>       retn
                                <1>       ;
                                <1> vga_wtty_8:
00002A3F 83C64F                 <1>       add   esi, vga_g_memmodel - (vga_g_modes + 1)
                                <1>       ; [ESI] = VGA memory model number (LINEAR8, PLANAR4, PLANAR1)
00002A42 BF00000A00             <1>       mov   edi, 0A0000h
                                <1>       ;
00002A47 88F8                   <1>       mov   al, bh ; character
                                <1>       ;
00002A49 803E04                 <1>       cmp   byte [esi], PLANAR4
00002A4C 7414                   <1>       je    short vga_wtty_planar
00002A4E 803E03                 <1>       cmp   byte [esi], PLANAR1
```

```
8445 00002A51 740F                <1>       je    short vga_wtty_planar
8446                              <1> vga_wtty_linear8:
8447                              <1>       ; write_gfx_char_lin(car,attr,xcurs,ycurs,nbcols);
8448                              <1>       ; AL = car, BL = attr (color), DL = xcurs, DH = ycurs,
8449                              <1>       ; [CRT_COLS] = nbcols
8450 00002A53 E8F3FCFFFF          <1>       call  write_gfx_char_lin
8451 00002A58 EB0D                <1>       jmp   short vga_wtty_9
8452                              <1>
8453                              <1> vga_wtty_12:
8454                              <1>       ; 09/07/2016
8455                              <1>       ; set cursor position
8456                              <1>       ; NOTE: Hardware cursor position will not be set
8457                              <1>       ;   in any VGA modes (>7)
8458                              <1>       ;   But, cursor position will be saved into
8459                              <1>       ;   [CURSOR_POSN].
8460                              <1>       ;   TRDOS 386 (TRDOS v2.0) uses only one page
8461                              <1>       ;   (page 0) for all graphics modes.
8462                              <1>
8463 00002A5A 668915[3E520100]    <1>       mov   [CURSOR_POSN], dx ; save cursor pos for pg 0
8464                              <1>       ; 04/08/2016
8465                              <1>       ;mov   bh, [ACTIVE_PAGE] ; = 0
8466                              <1>       ;call  _set_cpos
8467 00002A61 C3                  <1>       retn
8468                              <1>
8469                              <1> vga_wtty_planar:
8470                              <1>       ; write_gfx_char_pl4(car,attr,xcurs,ycurs,nbcols,cheight);
8471                              <1>       ; AL = car, BL = attr (color), DL = xcurs, DH = ycurs,
8472                              <1>       ; [CRT_COLS]= nbcols, [CHAR_HEIGHT] = cheight
8473 00002A62 E87BFDFFFF          <1>       call  write_gfx_char_pl4
8474                              <1> vga_wtty_9:
8475 00002A67 FEC2                <1>       inc   dl ; xcurs++;
8476                              <1> vga_wtty_10:
8477                              <1>       ; Do we need to wrap ?
8478                              <1>       ; if(xcurs==nbcols)
8479 00002A69 3A15[C45E0000]      <1>       cmp   dl, [CRT_COLS] ; [VGA_COLS]
8480 00002A6F 7204                <1>       jb    short vga_wtty_11 ; no
8481 00002A71 28D2                <1>       sub   dl, dl ; xcurs=0;
8482 00002A73 FEC6                <1>       inc   dh ;  ycurs++;
8483                              <1> vga_wtty_11:
8484                              <1>       ; Do we need to scroll ?
8485                              <1>       ; if(ycurs==nbrows)
8486 00002A75 3A35[CA5E0000]      <1>       cmp   dh, [VGA_ROWS]
8487 00002A7B 72DD                <1>       jb    short vga_wtty_12 ; no
8488                              <1>       ;
8489                              <1>       ; biosfn_scroll (nblines,attr,rul,cul,rlr,clr,page,dir)
8490                              <1>       ; al = nblines = 1, bl = attr (color) = 0
8491                              <1>       ; ch = rul, cl = cul, dh = rlr, dl = clr, page = 0
8492                              <1>       ; dir = SCROLL_UP
8493                              <1>
8494 00002A7D B001                <1>       mov   al, 1
8495 00002A7F 28DB                <1>       sub   bl, bl ; 0 ; blank/black line (attr=0) will be used
8496 00002A81 6629C9              <1>       sub   cx, cx ; 0,0
8497                              <1>
8498                              <1>       ; 06/08/2016
8499 00002A84 8A35[CA5E0000]      <1>       mov   dh, [VGA_ROWS]
8500 00002A8A FECE                <1>       dec   dh ; nbrows -1
8501                              <1>
8502 00002A8C 6652                <1>       push  dx    ; 04/08/2016
8503 00002A8E 8A15[C45E0000]      <1>       mov   dl, [CRT_COLS]
8504 00002A94 FECA                <1>       dec   dl ; nbcols -1
8505                              <1>
8506 00002A96 8A25[C25E0000]      <1>       mov   ah, [CRT_MODE]
8507                              <1>
8508                              <1>       ; biosfn_scroll(0x01,0x00,0,0,nbrows-1,nbcols-1,page,SCROLL_UP);
8509 00002A9C E808F5FFFF          <1>       call  vga_graphics_up
8510                              <1>       ; 04/08/2016
8511 00002AA1 665A                <1>       pop   dx
8512                              <1>       ;dec  dh ; ycurs-=1
8513 00002AA3 EBB5                <1>       jmp   short vga_wtty_12
8514                              <1>
8515                              <1> font_setup:
8516                              <1>       ; 09/07/2016
8517                              <1>       ; character generator (font loading) functions
8518                              <1>       ;
8519                              <1>       ; derived from 'Plex86/Bochs VGABios' source code
8520                              <1>       ; vgabios-0.7a (2011)
8521                              <1>       ; by the LGPL VGABios developers Team (2001-2008)
8522                              <1>       ; 'vgabios.c', 'int10_func'
8523                              <1>
8524                              <1>       ; AX = 1100H ; Load User-Defined Font (EGA/VGA)
8525                              <1>       ;
8526                              <1>         ; BH    height of each character (bytes per character definition)
8527                              <1>         ; (BL   font block to load (EGA: 0-3; VGA: 0-7))
8528                              <1>       ; CX    number of characters to redefine (<=256)
8529                              <1>         ; DX    ASCII code of the first character defined at ES:BP
8530                              <1>         ; EBP      address of font-definition information
8531                              <1>       ;     (in user's memory space)
8532                              <1>
8533                              <1>       ; case 0x11:
8534                              <1>       ; switch(GET_AL())
8535                              <1>             ; {
8536                              <1>       ; case 0x00:
8537                              <1>         ; case 0x10:
8538                              <1>         ; biosfn_load_text_user_pat(GET_AL(),ES,BP,CX,DX,GET_BL(),GET_BH());
8539                              <1>         ; break;
8540                              <1>
8541                              <1>       ; AX = 1110H ; Load and Activate User-Defined Font (EGA/VGA)
8542 00002AA5 08C0                <1>       or    al, al ; 0
8543 00002AA7 7404                <1>       jz    short font_setup_0
8544 00002AA9 3C10                <1>       cmp   al, 10h
8545 00002AAB 7511                <1>       jne   short font_setup_1
8546                              <1> font_setup_0:
8547 00002AAD E8B7000000          <1>       call  transfer_user_fonts
```

```
8548 00002AB2 721C              <1>       jc     short font_setup_error
8549 00002AB4 E8C2000000        <1>       call   load_text_user_pat
8550 00002AB9 E996EAFFFF        <1>         jmp     VIDEO_RETURN
8551                            <1> font_setup_1:
8552                            <1>       ; AX = 1101H ; Load ROM 8x14 Character Set (EGA/VGA)
8553                            <1>       ; case 0x01:
8554                            <1>         ; case 0x11:
8555                            <1>         ; biosfn_load_text_8_14_pat(GET_AL(),GET_BL());
8556                            <1>         ; break;
8557 00002ABE 3C01              <1>       cmp    al, 1
8558 00002AC0 7404              <1>       je     short font_setup_2
8559 00002AC2 3C11              <1>       cmp    al, 11h
8560 00002AC4 7511              <1>       jne    short font_setup_3
8561                            <1> font_setup_2:
8562                            <1>       ; AX = 1111H ; Load and Activate ROM 8x14 Character Set (EGA/VGA)
8563                            <1>       ; (BL = font block to load (EGA: 0-3; VGA: 0-7))
8564 00002AC6 E8EE010000        <1>       call   load_text_8_14_pat
8565 00002ACB E984EAFFFF        <1>         jmp     VIDEO_RETURN
8566                            <1> font_setup_error:
8567 00002AD0 29C0              <1>       sub    eax, eax ; 0 -> fonts could not be loaded
8568 00002AD2 E982EAFFFF        <1>       jmp    _video_return
8569                            <1> font_setup_3:
8570                            <1>       ; AX = 1102H ; Load ROM 8x8 Character Set (EGA/VGA)
8571                            <1>       ; case 0x02:
8572                            <1>         ; case 0x12:
8573                            <1>         ; biosfn_load_text_8_8_pat(GET_AL(),GET_BL());
8574                            <1>         ; break;
8575 00002AD7 3C02              <1>       cmp    al, 2
8576 00002AD9 7404              <1>       je     short font_setup_4
8577 00002ADB 3C12              <1>       cmp    al, 12h
8578 00002ADD 750A              <1>       jne    short font_setup_5
8579                            <1> font_setup_4:
8580                            <1>       ; AX = 1112H ; Load and Activate ROM 8x8 Character Set (EGA/VGA)
8581                            <1>       ; (BL = font block to load (EGA: 0-3; VGA: 0-7))
8582 00002ADF E805020000        <1>       call   load_text_8_8_pat
8583 00002AE4 E96BEAFFFF        <1>         jmp     VIDEO_RETURN
8584                            <1> font_setup_5:
8585                            <1>       ; AX = 1104H ; Load ROM 8x16 Character Set (EGA/VGA)
8586                            <1>       ; case 0x04:
8587                            <1>         ; case 0x14:
8588                            <1>         ; biosfn_load_text_8_16_pat(GET_AL(),GET_BL());
8589                            <1>         ; break;
8590 00002AE9 3C04              <1>       cmp    al, 4
8591 00002AEB 7404              <1>       je     short font_setup_6
8592 00002AED 3C14              <1>       cmp    al, 14h
8593 00002AEF 750A              <1>       jne    short font_setup_7
8594                            <1> font_setup_6:
8595                            <1>       ; AX = 1114H ; Load and Activate ROM 8x16 Character Set (EGA/VGA)
8596                            <1>       ; (BL = font block to load (EGA: 0-3; VGA: 0-7))
8597 00002AF1 E823020000        <1>       call   load_text_8_16_pat
8598 00002AF6 E959EAFFFF        <1>         jmp     VIDEO_RETURN
8599                            <1> font_setup_7:
8600                            <1>       ; Note: AX=1120h (Setup INT 1Fh, EXT_PTR) is not needed
8601                            <1>       ; for TRDOS 386 (TRDIOS v2.0) video functionality;
8602                            <1>       ; because, originally EXT_PTR (font address) was used for
8603                            <1>       ; chars 80h to 0FFh (after the first 128 ASCII char fonts), for
8604                            <1>       ; CGA graphics mode; currenty, 'vgafont8' address has 256 chars!
8605                            <1>       ;
8606                            <1>       ; case 0x20:
8607                            <1>         ; biosfn_load_gfx_8_8_chars(ES,BP);
8608                            <1>         ; break;
8609                            <1>       ; case 0x21:
8610                            <1>         ; biosfn_load_gfx_user_chars(ES,BP,CX,GET_BL(),GET_DL());
8611                            <1>         ; break;
8612                            <1>       ; AX = 1121H ; Setup User-Defined Font for Graphics Mode (VGA)
8613                            <1>       ; BL   screen rows code: 00H = user-specified (in DL)
8614                            <1>         ;                        01H = 14 rows
8615                            <1>         ;                        02H = 25 rows
8616                            <1>         ;                        03H = 43 rows
8617                            <1>         ; CX   bytes per character definition
8618                            <1>         ; DL   (when BL=0) custom number of character rows on screen
8619                            <1>         ; EBP  address of font-definition information (user's mem space)
8620                            <1>
8621 00002AFB 3C21              <1>       cmp    al, 21h
8622 00002AFD 751A              <1>       jne    short font_setup_9
8623                            <1>
8624                            <1>       ; TRDOS 386 modification !
8625                            <1>       ; dh = 0 -> 256 characters
8626                            <1>       ; dh = 80h -> 128 characters
8627                            <1>       ; (If DH <> 0 and DH <> 80h -> invalid)
8628 00002AFF 20F6              <1>       and    dh, dh
8629 00002B01 7405              <1>       jz     short font_setup_8 ; 256 characters
8630 00002B03 80FE80            <1>       cmp    dh, 80h ; 128 characters
8631 00002B06 75C8              <1>       jne    short font_setup_error ; invalid !
8632                            <1> font_setup_8:
8633 00002B08 E85C000000        <1>       call   transfer_user_fonts
8634 00002B0D 72C1              <1>       jc     short font_setup_error
8635                            <1>       ; ebp = user's font data address in system's memory space
8636 00002B0F E836020000        <1>       call   load_gfx_user_chars
8637 00002B14 E93BEAFFFF        <1>         jmp     VIDEO_RETURN
8638                            <1> font_setup_9:
8639                            <1>       ; case 0x22:
8640                            <1>         ; biosfn_load_gfx_8_14_chars(GET_BL());
8641                            <1>         ; break;
8642 00002B19 3C22              <1>       cmp    al, 22h
8643 00002B1B 750A              <1>       jne    short font_setup_10
8644 00002B1D E866020000        <1>       call   load_gfx_8_14_chars
8645 00002B22 E92DEAFFFF        <1>         jmp     VIDEO_RETURN
8646                            <1> font_setup_10:
8647                            <1>       ; case 0x23:
8648                            <1>         ; biosfn_load_gfx_8_8_dd_chars(GET_BL());
8649                            <1>         ; break;
8650 00002B27 3C23              <1>       cmp    al, 23h
```

```
8651 00002B29 750A              <1>       jne    short font_setup_11
8652 00002B2B E899020000        <1>       call   load_gfx_8_8_chars
8653 00002B30 E91FEAFFFF        <1>         jmp     VIDEO_RETURN
8654                            <1> font_setup_11:
8655                            <1>       ; case 0x24:
8656                            <1>         ; biosfn_load_gfx_8_16_chars(GET_BL());
8657                            <1>         ; break;
8658 00002B35 3C24              <1>       cmp    al, 24h
8659 00002B37 750A              <1>       jne    short font_setup_12
8660 00002B39 E8CC020000        <1>       call   load_gfx_8_16_chars
8661 00002B3E E911EAFFFF        <1>         jmp     VIDEO_RETURN
8662                            <1> font_setup_12:
8663                            <1>       ; case 0x30:
8664                            <1>         ; biosfn_get_font_info(GET_BH(),&ES,&BP,&CX,&DX);
8665                            <1>         ; break;
8666 00002B43 3C30              <1>       cmp    al, 30h
8667 00002B45 750A              <1>       jne    short font_setup_13
8668 00002B47 E8FF020000        <1>       call   get_font_info
8669                            <1>       ; eax = return value (info: 4 bytes for 4 parms)
8670                            <1>       ; eax = 0 -> invalid function (input)
8671 00002B4C E908EAFFFF        <1>         jmp     _video_return
8672                            <1> font_setup_13:
8673 00002B51 3C03              <1>       cmp    al, 03h ; AX = 1103h
8674 00002B53 750D              <1>       jne    short font_setup_14
8675                            <1>       ; biosfn_set_text_block_specifier:
8676                            <1>       ; BL = font block selector code
8677                            <1>       ; NOTE: TRDOS 386 only uses and sets font block 0
8678                            <1>       ; (It is as BL = 0 for TRDOS 386)
8679 00002B55 66BAC403          <1>       mov    dx, 3C4h ; VGAREG_SEQU_ADDRESS
8680                            <1>       ;mov   ah, bl
8681 00002B59 28E4              <1>       sub    ah, ah ; 0
8682                            <1>       ;mov   al, 03h
8683 00002B5B 66EF              <1>       out    dx, ax
8684 00002B5D E9F2E9FFFF        <1>       jmp     VIDEO_RETURN
8685                            <1>
8686                            <1> font_setup_14:
8687 00002B62 29C0              <1>       sub    eax, eax ; 0 = invalid function
8688 00002B64 E9F0E9FFFF        <1>         jmp     _video_return
8689                            <1>
8690                            <1> transfer_user_fonts:
8691                            <1>       ; 09/07/2016
8692                            <1>       ;and   ecx, 0FFFFh
8693                            <1>       ; ECX = byte count
8694                            <1>       ;push  ecx
8695 00002B69 89EE              <1>       mov    esi, ebp ; user buffer
8696 00002B6B BF00000700        <1>       mov    edi, Cluster_Buffer  ; system buffer
8697 00002B70 E882BD0000        <1>       call   transfer_from_user_buffer
8698                            <1>       ;pop   ecx
8699                            <1>       ; ecx = transfer (byte) count = character count
8700 00002B75 BD00000700        <1>       mov    ebp, Cluster_Buffer
8701                            <1>       ; jc   VIDEO_RETURN -> failed
8702 00002B7A C3                <1>       retn
8703                            <1>
8704                            <1> load_text_user_pat:
8705                            <1>       ; 26/07/2016
8706                            <1>       ; 09/07/2016
8707                            <1>       ; load user defined (EGA/VGA) text fonts
8708                            <1>       ;
8709                            <1>       ; derived from 'Plex86/Bochs VGABios' source code
8710                            <1>       ; vgabios-0.7a (2011)
8711                            <1>       ; by the LGPL VGABios developers Team (2001-2008)
8712                            <1>       ; 'vgabios.c', 'biosfn_load_text_user_pat'
8713                            <1>
8714                            <1>       ; biosfn_load_text_user_pat (AL,ES,BP,CX,DX,BL,BH)
8715                            <1>
8716                            <1>       ; get_font_access();
8717                            <1>       ; blockaddr = ((BL & 0x03) << 14) + ((BL & 0x04) << 11);
8718                            <1>       ; for(i=0;i<CX;i++)
8719                            <1>       ; {
8720                            <1>       ;  src = BP + i * BH;
8721                            <1>       ;  dest = blockaddr + (DX + i) * 32;
8722                            <1>       ;  memcpyb(0xA000, dest, ES, src, BH);
8723                            <1>       ; }
8724                            <1>       ; release_font_access();
8725                            <1>       ; if(AL>=0x10)
8726                            <1>       ; {
8727                            <1>       ; set_scan_lines(BH);
8728                            <1>       ; }
8729                            <1>
8730 00002B7B 50                <1>       push   eax
8731 00002B7C E83C000000        <1>       call   get_font_access
8732 00002B81 28DB              <1>       sub    bl, bl ; i = 0
8733                            <1> ltup_1:
8734 00002B83 88D8              <1>       mov    al, bl
8735 00002B85 F6E7              <1>       mul    bh
8736 00002B87 0FB7F0            <1>       movzx  esi, ax
8737 00002B8A 01EE              <1>       add    esi, ebp
8738 00002B8C 88D8              <1>       mov    al, bl
8739 00002B8E 28E4              <1>       sub    ah, ah
8740 00002B90 6601D0            <1>       add    ax, dx ; (DX + i)
8741 00002B93 66C1E005          <1>       shl    ax, 5  ; * 32
8742 00002B97 0FB7F8            <1>       movzx  edi, ax
8743 00002B9A 81C700000A00      <1>       add    edi, 0A0000h
8744 00002BA0 51                <1>       push   ecx
8745 00002BA1 0FB6CF            <1>       movzx  ecx, bh
8746 00002BA4 F3A4              <1>       rep    movsb
8747 00002BA6 59                <1>       pop    ecx
8748 00002BA7 FEC3              <1>       inc    bl
8749 00002BA9 38CB              <1>       cmp    bl, cl
8750 00002BAB 75D6              <1>       jne    short ltup_1
8751                            <1>       ;
8752 00002BAD E840000000        <1>       call   release_font_access
8753                            <1>       ;
```

```
8754 00002BB2 58                <1>       pop    eax
8755                            <1>       ; if(AL>=0x10)
8756 00002BB3 3C10              <1>       cmp    al, 10h
8757 00002BB5 7205              <1>       jb     short ltup_2
8758                            <1>       ; set_scan_lines(BH);
8759 00002BB7 E875000000        <1>       call   set_scan_lines
8760                            <1> ltup_2:
8761 00002BBC C3                <1>       retn
8762                            <1>
8763                            <1> get_font_access:
8764                            <1>       ; 09/07/2016
8765                            <1>       ;
8766                            <1>       ; derived from 'Plex86/Bochs VGABios' source code
8767                            <1>       ; vgabios-0.7a (2011)
8768                            <1>       ; by the LGPL VGABios developers Team (2001-2008)
8769                            <1>       ; 'vgabios.c', 'get_font_access'
8770                            <1>
8771                            <1>       ; get_font_access()
8772 00002BBD 52                <1>       push   edx
8773 00002BBE 66BAC403          <1>       mov    dx, 3C4h ; VGAREG_SEQU_ADDRESS
8774 00002BC2 66B80001          <1>       mov    ax, 0100h
8775 00002BC6 66EF              <1>       out    dx, ax
8776 00002BC8 66B80204          <1>       mov    ax, 0402h
8777 00002BCC 66EF              <1>       out    dx, ax
8778 00002BCE 66B80407          <1>       mov    ax, 0704h
8779 00002BD2 66EF              <1>       out    dx, ax
8780 00002BD4 66B80003          <1>       mov    ax, 0300h
8781 00002BD8 66EF              <1>       out    dx, ax
8782 00002BDA 66BACE03          <1>       mov    dx, 3CEh ; VGAREG_GRDC_ADDRESS
8783 00002BDE 66B80402          <1>       mov    ax, 0204h
8784 00002BE2 66EF              <1>       out    dx, ax
8785 00002BE4 66B80500          <1>       mov    ax, 0005h
8786 00002BE8 66EF              <1>       out    dx, ax
8787 00002BEA 66B80604          <1>       mov    ax, 0406h
8788 00002BEE 66EF              <1>       out    dx, ax
8789 00002BF0 5A                <1>       pop    edx
8790 00002BF1 C3                <1>       retn
8791                            <1>
8792                            <1> release_font_access:
8793                            <1>       ; 29/07/2016
8794                            <1>       ; 09/07/2016
8795                            <1>       ;
8796                            <1>       ; derived from 'Plex86/Bochs VGABios' source code
8797                            <1>       ; vgabios-0.7a (2011)
8798                            <1>       ; by the LGPL VGABios developers Team (2001-2008)
8799                            <1>       ; 'vgabios.c', 'release_font_access'
8800                            <1>
8801 00002BF2 66BAC403          <1>       mov    dx, 3C4h ; VGAREG_SEQU_ADDRESS
8802 00002BF6 66B80001          <1>       mov    ax, 0100h
8803 00002BFA 66EF              <1>       out    dx, ax
8804 00002BFC 66B80203          <1>       mov    ax, 0302h
8805 00002C00 66EF              <1>       out    dx, ax
8806 00002C02 66B80403          <1>       mov    ax, 0304h
8807 00002C06 66EF              <1>       out    dx, ax
8808 00002C08 66B80003          <1>       mov    ax, 0300h
8809 00002C0C 66EF              <1>       out    dx, ax
8810 00002C0E 66BACC03          <1>       mov    dx, 3CCh ; VGAREG_READ_MISC_OUTPUT
8811 00002C12 EC                <1>       in     al, dx
8812 00002C13 2401              <1>       and    al, 01h
8813 00002C15 C0E002            <1>       shl    al, 2
8814 00002C18 0C0A              <1>       or     al, 0Ah
8815 00002C1A 88C4              <1>       mov    ah, al
8816 00002C1C B006              <1>       mov    al, 06h
8817 00002C1E 66BACE03          <1>       mov    dx, 3CEh ; VGAREG_GRDC_ADDRESS
8818 00002C22 66EF              <1>       out    dx, ax
8819 00002C24 66B80400          <1>       mov    ax, 0004h
8820 00002C28 66EF              <1>       out    dx, ax
8821 00002C2A 66B80510          <1>       mov    ax, 1005h
8822 00002C2E 66EF              <1>       out    dx, ax
8823 00002C30 C3                <1>       retn
8824                            <1>
8825                            <1> set_scan_lines:
8826                            <1>       ; 09/07/2016
8827                            <1>       ;
8828                            <1>       ; derived from 'Plex86/Bochs VGABios' source code
8829                            <1>       ; vgabios-0.7a (2011)
8830                            <1>       ; by the LGPL VGABios developers Team (2001-2008)
8831                            <1>       ; 'vgabios.c', 'set_scan_lines'
8832                            <1>
8833                            <1>       ; set_scan_lines(lines)
8834                            <1>       ; BH = lines
8835                            <1>
8836                            <1>       ; outb(crtc_addr, 0x09);
8837 00002C31 66BAD403          <1>       mov    dx, 3D4h ; CRTC_ADDRESS = 3D4h (always)
8838 00002C35 B009              <1>       mov    al, 09h
8839 00002C37 EE                <1>       out    dx, al
8840                            <1>       ; crtc_r9 = inb(crtc_addr+1);
8841 00002C38 6642              <1>       inc    dx ; 3D5h
8842 00002C3A EC                <1>       in     al, dx
8843                            <1>       ; crtc_r9 = (crtc_r9 & 0xe0) | (lines - 1);
8844 00002C3B 24E0              <1>       and    al, 0E0h
8845 00002C3D FECF              <1>       dec    bh ; lines - 1
8846 00002C3F 08F8              <1>       or     al, bh
8847                            <1>       ; outb(crtc_addr+1, crtc_r9);
8848 00002C41 EE                <1>       out    dx, al
8849                            <1>       ;inc   bh
8850                            <1>       ; if(lines==8)
8851                            <1>       ;cmp   bh, 8
8852 00002C42 80FF07            <1>       cmp    bh, 7
8853 00002C45 7506              <1>       jne    short ssl_1
8854                            <1>       ; biosfn_set_cursor_shape(0x06,0x07);
8855 00002C47 66B90706          <1>       mov    cx, 0607h
8856 00002C4B EB06              <1>       jmp    short ssl_2
```

```
                            <1> ssl_1:
                            <1>       ; biosfn_set_cursor_shape(lines-4,lines-3);
00002C4D 88F9               <1>       mov   cl, bh ; lines - 1
00002C4F 88CD               <1>       mov   ch, cl ; lines - 1 (16 -> 15)
00002C51 FECD               <1>       dec   ch  ; lines - 2 (16 -> 14)
                            <1> ssl_2:
                            <1>       ; CH = start line, CL = stop line
00002C53 B40A               <1>       mov   ah, 10 ; 6845 register for cursor set
00002C55 66890D[DB5E0000]   <1>       mov   [CURSOR_MODE], cx ; save in data area
00002C5C E812F1FFFF         <1>       call  m16   ; output cx register
                            <1>       ; write_word(BIOSMEM_SEG,BIOSMEM_CHAR_HEIGHT, lines);
00002C61 FEC7               <1>       inc   bh ; lines
00002C63 883D[C65E0000]     <1>       mov   [CHAR_HEIGHT], bh
                            <1>       ;  outb(crtc_addr, 0x12);
00002C69 66BAD403           <1>       mov   dx, 3D4h ; CRTC_ADDRESS
00002C6D B012               <1>       mov   al, 12h
00002C6F EE                 <1>       out   dx, al
                            <1>       ; vde = inb(crtc_addr+1);
00002C70 6642               <1>       inc   dx
00002C72 EC                 <1>       in    al, dx
00002C73 88C4               <1>       mov   ah, al
                            <1>       ; outb(crtc_addr, 0x07);
00002C75 664A               <1>       dec   dx
00002C77 B007               <1>       mov   al, 07h
00002C79 EE                 <1>       out   dx, al
                            <1>       ; ovl = inb(crtc_addr+1);
00002C7A 6642               <1>       inc   dx
00002C7C EC                 <1>       in    al, dx
                            <1>       ; vde += (((ovl & 0x02) << 7) + ((ovl & 0x40) << 3) + 1);
00002C7D 88E2               <1>       mov   dl, ah ; vde
00002C7F 88C6               <1>       mov   dh, al ; ovl
00002C81 6683E002           <1>       and   ax, 02h
00002C85 66C1E007           <1>       shl   ax, 7
00002C89 6689C1             <1>       mov   cx, ax ; (ovl & 0x02) << 7)
00002C8C 88F0               <1>       mov   al, dh ; ovl
00002C8E 6683E040           <1>       and   ax, 40h
00002C92 66C1E003           <1>       shl   ax, 3  ; (ovl & 0x40) << 3)
00002C96 6640               <1>       inc   ax ; + 1
00002C98 6601C8             <1>       add   ax, cx
00002C9B 30F6               <1>       xor   dh, dh
00002C9D 6601D0             <1>       add   ax, dx ; + vde
                            <1>       ; rows = vde / lines;
00002CA0 F6F7               <1>       div   bh
                            <1>       ;dec  al ; rows -1
                            <1>       ; write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, rows-1);
00002CA2 A2[CA5E0000]       <1>       mov   [VGA_ROWS], al ; rows (not 'rows-1' !)
                            <1>       ; write_word(BIOSMEM_SEG,BIOSMEM_PAGE_SIZE, rows * cols * 2);
00002CA7 8A25[C45E0000]     <1>       mov   ah, [CRT_COLS]
00002CAD F6E4               <1>       mul   ah
00002CAF 66D1E0             <1>       shl   ax, 1
00002CB2 66A3[BC5F0100]     <1>       mov   [CRT_LEN], ax
00002CB8 C3                 <1>       retn
                            <1>
                            <1> load_text_8_14_pat:
                            <1>       ; 26/07/2016
                            <1>       ; 25/07/2016
                            <1>       ; 23/07/2016
                            <1>       ; 09/07/2016
                            <1>       ; load user defined (EGA/VGA) text fonts
                            <1>       ;
                            <1>       ; derived from 'Plex86/Bochs VGABios' source code
                            <1>       ; vgabios-0.7a (2011)
                            <1>       ; by the LGPL VGABios developers Team (2001-2008)
                            <1>       ; 'vgabios.c', 'biosfn_load_text_8_14_pat'
                            <1>
                            <1>       ; biosfn_load_text_8_14_pat (AL,BL)
                            <1>
                            <1>       ; get_font_access();
                            <1>       ; blockaddr = ((BL & 0x03) << 14) + ((BL & 0x04) << 11);
                            <1>       ; for(i=0;i<0x100;i++)
                            <1>       ; {
                            <1>       ;  src = i * 14;
                            <1>       ;  dest = blockaddr + i * 32;
                            <1>       ;  memcpyb(0xA000, dest, 0xC000, vgafont14+src, 14);
                            <1>       ; }
                            <1>       ; release_font_access();
                            <1>       ; if(AL>=0x10)
                            <1>       ; {
                            <1>       ; set_scan_lines(14);
                            <1>       ; }
                            <1>
00002CB9 50                 <1>       push  eax
00002CBA E8FEFEFFFF         <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>
00002CBF 28DB               <1>       sub   bl, bl ; i = 0
00002CC1 B70E               <1>       mov   bh, 14
00002CC3 BE[A02E0100]       <1>       mov   esi, vgafont14
00002CC8 BF00000A00         <1>       mov   edi, 0A0000h
                            <1> lt8_14_1:
                            <1>       ;mov  al, bl
                            <1>       ;mul  bh
```

```
                                <1>       ;movzx esi, ax
                                <1>       ;add   esi, vgafont14
                                <1>       ;mov   al, bl
                                <1>       ;sub   ah, ah
                                <1>       ;shl   ax, 5 ; * 32
                                <1>       ;;add  ax, dx ; blockaddr + i * 32;
                                <1>       ;movzx edi, ax ; dest
                                <1>       ;add   edi, 0A0000h
00002CCD 0FB6CF                 <1>       movzx  ecx, bh
00002CD0 F3A4                   <1>       rep    movsb
00002CD2 83C712                 <1>       add    edi, 18 ; 32 - 14
00002CD5 FEC3                   <1>       inc    bl
00002CD7 75F4                   <1>       jnz    short lt8_14_1
                                <1>       ;
00002CD9 E814FFFFFF             <1>       call   release_font_access
                                <1>       ;
00002CDE 58                     <1>       pop    eax
                                <1>       ; if(AL>=0x10)
00002CDF 3C10                   <1>       cmp    al, 10h
00002CE1 7205                   <1>       jb     short lt8_14_4
                                <1>       ; BH = 14
                                <1>       ; set_scan_lines(14);
00002CE3 E849FFFFFF             <1>       call   set_scan_lines
                                <1> lt8_14_4:
00002CE8 C3                     <1>       retn
                                <1>
                                <1> load_text_8_8_pat:
                                <1>       ; 26/07/2016
                                <1>       ; 25/07/2016
                                <1>       ; 23/07/2016
                                <1>       ; 09/07/2016
                                <1>       ; load user defined (EGA/VGA) text fonts
                                <1>       ;
                                <1>       ; derived from 'Plex86/Bochs VGABios' source code
                                <1>       ; vgabios-0.7a (2011)
                                <1>       ; by the LGPL VGABios developers Team (2001-2008)
                                <1>       ; 'vgabios.c', 'biosfn_load_text_8_8_pat'
                                <1>
                                <1>       ; biosfn_load_text_8_8_pat (AL,BL)
                                <1>
                                <1>       ; get_font_access();
                                <1>       ; blockaddr = ((BL & 0x03) << 14) + ((BL & 0x04) << 11);
                                <1>       ; for(i=0;i<0x100;i++)
                                <1>       ; {
                                <1>       ;  src = i * 8;
                                <1>       ;  dest = blockaddr + i * 32;
                                <1>       ;  memcpyb(0xA000, dest, 0xC000, vgafont8+src, 8);
                                <1>       ; }
                                <1>       ; release_font_access();
                                <1>       ; if(AL>=0x10)
                                <1>       ; {
                                <1>       ; set_scan_lines(8);
                                <1>       ; }
                                <1>
00002CE9 50                     <1>       push   eax
00002CEA E8CEFEFFFF             <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>
00002CEF 28DB                   <1>       sub    bl, bl ; i = 0
00002CF1 B708                   <1>       mov    bh, 8
00002CF3 BE[A0260100]           <1>       mov    esi, vgafont8
00002CF8 BF00000A00             <1>       mov    edi, 0A0000h
                                <1> lt8_8_1:
                                <1>       ;mov   al, bl
                                <1>       ;mul   bh
                                <1>       ;movzx esi, ax
                                <1>       ;add   esi, vgafont8
                                <1>       ;mov   al, bl
                                <1>       ;sub   ah, ah
                                <1>       ;shl   ax, 5 ; * 32
                                <1>       ;;add  ax, dx ; blockaddr + i * 32;
                                <1>       ;movzx edi, ax ; dest
                                <1>       ;add   edi, 0A0000h
00002CFD 0FB6CF                 <1>       movzx  ecx, bh
00002D00 F3A4                   <1>       rep    movsb
00002D02 83C718                 <1>       add    edi, 24 ; 32 - 8
00002D05 FEC3                   <1>       inc    bl
00002D07 75F4                   <1>       jnz    short lt8_8_1
                                <1>       ;
00002D09 E8E4FEFFFF             <1>       call   release_font_access
                                <1>       ;
00002D0E 58                     <1>       pop    eax
                                <1>       ; if(AL>=0x10)
00002D0F 3C10                   <1>       cmp    al, 10h
00002D11 7205                   <1>       jb     short lt8_8_2
                                <1>       ; BH = 8
                                <1>       ; set_scan_lines(8);
00002D13 E819FFFFFF             <1>       call   set_scan_lines
                                <1> lt8_8_2:
00002D18 C3                     <1>       retn
                                <1>
                                <1> load_text_8_16_pat:
```

```
                                <1>       ; 26/07/2016
                                <1>       ; 25/07/2016
                                <1>       ; 23/07/2016
                                <1>       ; 09/07/2016
                                <1>       ; load user defined (EGA/VGA) text fonts
                                <1>       ;
                                <1>       ; derived from 'Plex86/Bochs VGABios' source code
                                <1>       ; vgabios-0.7a (2011)
                                <1>       ; by the LGPL VGABios developers Team (2001-2008)
                                <1>       ; 'vgabios.c', 'biosfn_load_text_8_16_pat'
                                <1>
                                <1>       ; biosfn_load_text_8_16_pat (AL,BL)
                                <1>
                                <1>       ; get_font_access();
                                <1>       ; blockaddr = ((BL & 0x03) << 14) + ((BL & 0x04) << 11);
                                <1>       ; for(i=0;i<0x100;i++)
                                <1>       ; {
                                <1>       ;  src = i * 16;
                                <1>       ;  dest = blockaddr + i * 32;
                                <1>       ;  memcpyb(0xA000, dest, 0xC000, vgafont16+src, 16);
                                <1>       ; }
                                <1>       ; release_font_access();
                                <1>       ; if(AL>=0x10)
                                <1>       ; {
                                <1>       ; set_scan_lines(16);
                                <1>       ; }
                                <1>
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[A03C0100]           <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[CE5F0100]           <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
00002D6D 740B           <1>       jz     short l_gfx_uc_3 ; bl = 2
00002D6F FECB           <1>       dec    bl
00002D71 750E           <1>       jnz    short l_gfx_uc_4 ; bl > 3
                        <1>       ; bl = 3
                        <1>       ; case 3:
                        <1>       ;    write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, 42);
                        <1>       ;    break;
00002D73 C605[CA5E0000]2B <1>     mov    byte [VGA_ROWS], 43  ; not 42 !
                        <1> l_gfx_uc_3:
                        <1>       ; case 2:
                        <1>       ; default:
                        <1>       ;    write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, 24);
                        <1>       ;    break;
                        <1>       ; bl = 2 or bl > 3
00002D7A C605[CA5E0000]19 <1>     mov    byte [VGA_ROWS], 25  ; not 24 !
                        <1>       ; }
                        <1> l_gfx_uc_4:
                        <1>       ; write_byte(BIOSMEM_SEG, BIOSMEM_CHAR_HEIGHT, CX);
00002D81 880D[C65E0000] <1>       mov    [CHAR_HEIGHT], cl
                        <1>       ; }
00002D87 C3             <1>       retn
                        <1>
                        <1> load_gfx_8_14_chars:
                        <1>       ; 08/08/2016
                        <1>       ; 10/07/2016
                        <1>       ; Setup ROM 8x14 Font for Graphics Mode (VGA)
                        <1>       ;
                        <1>       ; derived from 'Plex86/Bochs VGABios' source code
                        <1>       ; vgabios-0.7a (2011)
                        <1>       ; by the LGPL VGABios developers Team (2001-2008)
                        <1>       ; 'vgabios.c', 'biosfn_load_gfx_8_14_chars'
                        <1>
                        <1>       ; biosfn_load_gfx_8_14_chars (BL)
                        <1>       ; /* set 0x43 INT pointer */
                        <1>       ; write_word(0x0, 0x43*4, &vgafont14);
                        <1>       ; write_word(0x0, 0x43*4+2, 0xC000);
00002D88 C705[CE5F0100]- <1>      mov    dword [VGA_INT43H], vgafont14
00002D8E [A02E0100]     <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;
00002D92 20DB           <1>       and    bl, bl
00002D94 7508           <1>       jnz    short l_gfx_8_14c_1
00002D96 8815[CA5E0000] <1>       mov    [VGA_ROWS], dl  ; not DL-1 !
00002D9C EB23           <1>       jmp    short l_gfx_8_14c_4
                        <1> l_gfx_8_14c_1:
                        <1>       ; case 1:
                        <1>       ;    write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, 13);
                        <1>       ;    break;
00002D9E FECB           <1>       dec    bl
00002DA0 7509           <1>       jnz    short l_gfx_8_14c_2
                        <1>       ; bl = 1
00002DA2 C605[CA5E0000]0E <1>     mov    byte [VGA_ROWS], 14  ; not 13 !
00002DA9 EB16           <1>       jmp    short l_gfx_8_14c_4
                        <1> l_gfx_8_14c_2:
00002DAB FECB           <1>       dec    bl
00002DAD 740B           <1>       jz     short l_gfx_8_14c_3 ; bl = 2
00002DAF FECB           <1>       dec    bl
00002DB1 750E           <1>       jnz    short l_gfx_8_14c_4 ; bl > 3
                        <1>       ; bl = 3
                        <1>       ; case 3:
                        <1>       ;    write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, 42);
                        <1>       ;    break;
00002DB3 C605[CA5E0000]2B <1>     mov    byte [VGA_ROWS], 43  ; not 42 !
                        <1> l_gfx_8_14c_3:
                        <1>       ; case 2:
                        <1>       ; default:
                        <1>       ;    write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, 24);
                        <1>       ;    break;
                        <1>       ; bl = 2 or bl > 3
00002DBA C605[CA5E0000]19 <1>     mov    byte [VGA_ROWS], 25  ; not 24 !
                        <1>       ; }
                        <1> l_gfx_8_14c_4:
```

```
                                <1>       ; write_byte(BIOSMEM_SEG, BIOSMEM_CHAR_HEIGHT, 14);
00002DC1 C605[C65E0000]0E       <1>         mov     byte [CHAR_HEIGHT], 14
                                <1>       ; }
00002DC8 C3                     <1>       retn
                                <1>
                                <1> load_gfx_8_8_chars:
                                <1>       ; 08/08/2016
                                <1>       ; 10/07/2016
                                <1>       ; Setup ROM 8x14 Font for Graphics Mode (VGA)
                                <1>       ;
                                <1>       ; derived from 'Plex86/Bochs VGABios' source code
                                <1>       ; vgabios-0.7a (2011)
                                <1>       ; by the LGPL VGABios developers Team (2001-2008)
                                <1>       ; 'vgabios.c', 'biosfn_load_gfx_8_8_dd_chars'
                                <1>
                                <1>       ; biosfn_load_gfx_8_8_dd_chars (BL)
                                <1>       ; /* set 0x43 INT pointer */
                                <1>       ; write_word(0x0, 0x43*4, &vgafont8);
                                <1>       ; write_word(0x0, 0x43*4+2, 0xC000);
00002DC9 C705[CE5F0100]-        <1>       mov   dword [VGA_INT43H], vgafont8
00002DCF [A0260100]             <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);
00002E02 C605[C65E0000]08       <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[CE5F0100]-        <1>       mov   dword [VGA_INT43H], vgafont16
00002E10 [A03C0100]             <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);
                                <1>       ;    break;
                                <1>       ; bl = 2 or bl > 3
00002E3C C605[CA5E0000]19       <1>       mov    byte [VGA_ROWS], 25  ; not 24 !
                                <1>       ; }
                                <1> l_gfx_8_16c_4:
                                <1>       ; write_byte(BIOSMEM_SEG, BIOSMEM_CHAR_HEIGHT, 16);
00002E43 C605[C65E0000]10       <1>         mov     byte [CHAR_HEIGHT], 16
                                <1>       ; }
00002E4A C3                     <1>       retn
                                <1>
                                <1> get_font_info:
                                <1>       ; 19/09/2016
                                <1>       ; 08/08/2016
                                <1>       ; 10/07/2016
                                <1>       ; Get Current Character Generator Info (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_get_font_info'
                                <1>
                                <1>       ; Modified for TRDOS 386 !
                                <1>       ;
                                <1>       ; INPUT ->
                                <1>       ;    AX = 1130h
                                <1>       ;    BL = 0 -> Get info for current VGA font
                                <1>       ;             (BH = unused)
                                <1>       ;    19/09/2016
                                <1>       ;    BL > 0 -> Get requested character font data
                                <1>       ;       BL = 1 -> vgafont8
                                <1>       ;        BL = 2 -> vgafont14
                                <1>       ;       BL = 3 -> vgafont16
                                <1>       ;       BL > 3 -> Invalid function (for now!)
                                <1>       ;       BH = ASCII code of the first character
                                <1>       ;       ECX = Number of characters from the 1st char
                                <1>       ;       ECX >= 256 -> All (256-BH) characters
                                <1>       ;       ECX = 0 -> All characters (BH = unused)
                                <1>       ;        EDX = User's Buffer Address
                                <1>       ; OUTPUT ->
                                <1>       ;    AL = height (scanlines), bytes per character
                                <1>       ;    AH = screen rows
                                <1>       ;    Byte 16-23 of EAX = number of columns
                                <1>       ;    Byte 24-31 of EAX =
                                <1>       ;        0 -> default font (not configured yet)
                                <1>       ;        0FFh -> user defined font
                                <1>       ;        14 = vgafont14
                                <1>       ;         8 = vgafont8
                                <1>       ;        16 = vgafont16
                                <1>       ;   If BL input > 0 ->
                                <1>       ;       EAX = Actual transfer count
                                <1>       ;
00002E4B 20DB                   <1>       and    bl, bl
00002E4D 7408                   <1>       jz     short gfi_0
                                <1>       ; invalid function (input)
00002E4F 80FB03                 <1>       cmp    bl, 3
00002E52 7642                   <1>       jna    short gfi_4
00002E54 31C0                   <1>       xor    eax, eax ; 0
00002E56 C3                     <1>       retn
                                <1> gfi_0:
00002E57 A0[C65E0000]           <1>       mov    al, [CHAR_HEIGHT]
00002E5C 8A25[CA5E0000]         <1>       mov    ah, [VGA_ROWS]
00002E62 C1E010                 <1>       shl    eax, 16
00002E65 A0[C45E0000]           <1>       mov    al, [CRT_COLS]
00002E6A 8B0D[CE5F0100]         <1>       mov    ecx, [VGA_INT43H]
00002E70 21C9                   <1>       and    ecx, ecx
00002E72 741E                   <1>       jz     short gfi_2 ; 0 = default font
00002E74 41                     <1>       inc    ecx ; 0FFFFFFFFh -> 0 (user defined font)
00002E75 7504                   <1>       jnz    short gfi_1
00002E77 FECC                   <1>       dec    ah ; 0FFh
00002E79 EB17                   <1>       jmp    short gfi_2
                                <1> gfi_1:
00002E7B 49                     <1>       dec    ecx ; 08/08/2016
00002E7C B40E                   <1>       mov    ah, 14
00002E7E 81F9[A02E0100]         <1>       cmp    ecx, vgafont14
00002E84 740C                   <1>       je     short gfi_2
00002E86 B408                   <1>       mov    ah, 8
00002E88 81F9[A0260100]         <1>       cmp    ecx, vgafont8
00002E8E 7402                   <1>       je     short gfi_2
                                <1>       ; vgafont16
00002E90 D0E4                   <1>       shl    ah, 1 ; ah = 16
                                <1> gfi_2:
00002E92 C1C010                 <1>       rol    eax, 16
                                <1> gfi_3:
```

```
9475 00002E95 C3                <1>       retn
9476                            <1> gfi_4:
9477 00002E96 89D7              <1>       mov    edi, edx ; **
9478 00002E98 80FB02            <1>       cmp    bl, 2
9479 00002E9B 720B              <1>       jb     short gfi_5
9480 00002E9D 772F              <1>       ja     short gfi_7
9481                            <1>       ;BL = 2 -> vgafont14
9482 00002E9F BE[A02E0100]      <1>       mov    esi, vgafont14 ; *
9483 00002EA4 B30E              <1>       mov    bl, 14
9484 00002EA6 EB07              <1>       jmp    short gfi_6
9485                            <1> gfi_5:
9486                            <1>       ;BL = 1 -> vgafont8
9487 00002EA8 BE[A0260100]      <1>       mov    esi, vgafont8 ; *
9488 00002EAD B308              <1>       mov    bl, 8
9489                            <1> gfi_6:
9490 00002EAF 09C9              <1>       or     ecx, ecx
9491 00002EB1 7424              <1>       jz     short gfi_8 ; all chars from the 00h
9492 00002EB3 88F8              <1>       mov    al, bh ; character index
9493 00002EB5 F6E3              <1>       mul    bl ; char index * char height/size
9494 00002EB7 0FB7D0            <1>       movzx  edx, ax
9495 00002EBA 01D6              <1>       add    esi, edx ; *
9496 00002EBC 66BAFF00          <1>       mov    dx, 255
9497 00002EC0 28FA              <1>       sub    dl, bh
9498 00002EC2 6642              <1>       inc    dx
9499 00002EC4 39D1              <1>       cmp    ecx, edx
9500 00002EC6 770F              <1>       ja     short gfi_8
9501 00002EC8 7412              <1>       je     short gfi_9
9502 00002ECA 89D1              <1>       mov    ecx, edx
9503 00002ECC EB0E              <1>       jmp    short gfi_9
9504                            <1> gfi_7:
9505                            <1>       ;BL = 3 -> vgafont16
9506 00002ECE BE[A03C0100]      <1>       mov    esi, vgafont16 ; *
9507 00002ED3 B310              <1>       mov    bl, 16
9508 00002ED5 EBD8              <1>       jmp    short gfi_6
9509                            <1> gfi_8:
9510 00002ED7 B900010000        <1>       mov    ecx, 256
9511                            <1> gfi_9:
9512 00002EDC 6689C8            <1>       mov    ax, cx ; character count
9513 00002EDF 30FF              <1>       xor    bh, bh
9514 00002EE1 66F7E3            <1>       mul    bx ; char count * char height/size
9515 00002EE4 6689C1            <1>       mov    cx, ax
9516                            <1>
9517                            <1>       ; ESI = source address in system space
9518                            <1>       ; EDI = user's buffer address
9519                            <1>       ; ECX = transfer (byte) count
9520 00002EE7 E8C1B90000        <1>       call   transfer_to_user_buffer
9521 00002EEC 89C8              <1>       mov    eax, ecx ; actual transfer count
9522 00002EEE C3                <1>       retn
9523                            <1>
9524                            <1> vga_pal_funcs:
9525                            <1>       ; 10/08/2016
9526                            <1>       ; VGA Palette functions
9527                            <1>       ;
9528                            <1>       ; derived from 'Plex86/Bochs VGABios' source code
9529                            <1>       ; vgabios-0.7a (2011)
9530                            <1>       ; by the LGPL VGABios developers Team (2001-2008)
9531                            <1>       ; 'vgabios.c', 'vgarom.asm'
9532                            <1>
9533 00002EEF 3C00              <1>       cmp    al, 0
9534 00002EF1 0F848F000000      <1>         je      set_single_palette_reg
9535                            <1> vga_palf_1001:
9536 00002EF7 3C01              <1>       cmp    al, 1
9537 00002EF9 0F84B4000000      <1>         je      set_overscan_border_color
9538                            <1> vga_palf_1002:
9539 00002EFF 3C02              <1>       cmp    al, 2
9540 00002F01 0F84B0000000      <1>         je      set_all_palette_reg
9541                            <1> vga_palf_1003:
9542 00002F07 3C03              <1>       cmp    al, 3
9543 00002F09 0F84E8000000      <1>         je      toggle_intensity
9544                            <1> vga_palf_1007:
9545 00002F0F 3C07              <1>       cmp    al, 7
9546 00002F11 0F840D010000      <1>         je      get_single_palette_reg
9547 00002F17 7266              <1>       jb     short vga_palf_unknown
9548                            <1> vga_palf_1008:
9549 00002F19 3C08              <1>       cmp    al, 8
9550 00002F1B 0F8437010000      <1>         je      read_overscan_border_color
9551                            <1> vga_palf_1009:
9552 00002F21 3C09              <1>       cmp    al, 9
9553 00002F23 0F8433010000      <1>       je     get_all_palette_reg
9554                            <1> vga_palf_1010:
9555 00002F29 3C10              <1>       cmp    al, 10h
9556 00002F2B 0F8487010000      <1>       je     set_single_dac_reg
9557 00002F31 724C              <1>       jb     short vga_palf_unknown
9558                            <1> vga_palf_1012:
9559 00002F33 3C12              <1>       cmp    al, 12h
9560 00002F35 0F8498010000      <1>         je      set_all_dac_reg
9561 00002F3B 7242              <1>       jb     short vga_palf_unknown
9562                            <1> vga_palf_1013:
9563 00002F3D 3C13              <1>       cmp    al, 13h
9564 00002F3F 0F84CC010000      <1>         je      select_video_dac_color_page
9565                            <1> vga_palf_1015:
9566 00002F45 3C15              <1>       cmp    al, 15h
9567 00002F47 0F8412020000      <1>       je     read_single_dac_reg
9568 00002F4D 7230              <1>       jb     short vga_palf_unknown
9569                            <1> vga_palf_1017:
9570 00002F4F 3C17              <1>       cmp    al, 17h
9571 00002F51 0F8428020000      <1>         je      read_all_dac_reg
9572 00002F57 7226              <1>       jb     short vga_palf_unknown
9573                            <1> vga_palf_1018:
9574 00002F59 3C18              <1>       cmp    al, 18h
9575 00002F5B 0F845E020000      <1>         je      set_pel_mask
9576                            <1> vga_palf_1019:
9577 00002F61 3C19              <1>       cmp    al, 19h
```

```
9578 00002F63 0F8462020000        <1>         je      read_pel_mask
9579                              <1> vga_palf_101A:
9580 00002F69 3C1A                <1>         cmp     al, 1Ah
9581 00002F6B 0F8468020000        <1>         je      read_video_dac_state
9582                              <1> vga_palf_101B:
9583 00002F71 3C1B                <1>         cmp     al, 1Bh
9584                              <1>         ;jne    short vga_palf_unknown
9585 00002F73 770A                <1>         ja      short vga_palf_unknown
9586                              <1>
9587 00002F75 E80CF7FFFF          <1>         call    gray_scale_summing
9588 00002F7A E9D5E5FFFF          <1>         jmp     VIDEO_RETURN
9589                              <1>
9590                              <1> vga_palf_unknown:
9591 00002F7F 29C0                <1>         sub     eax, eax ; 0 = invalid function
9592 00002F81 E9D3E5FFFF          <1>         jmp     _video_return
9593                              <1>
9594                              <1> set_single_palette_reg:
9595                              <1>         ; 10/08/2016
9596                              <1>         ; Set One Palette Register
9597                              <1>         ; BL = register number to set
9598                              <1>         ;     (a 4-bit attribute nibble: 00h-0Fh)
9599                              <1>         ; BH = 6-bit RGB color to display
9600                              <1>         ;      for that attribute
9601                              <1>
9602 00002F86 80FB14              <1>         cmp     bl, 14h
9603                              <1>         ;ja     short no_actl_reg1
9604 00002F89 0F87C5E5FFFF        <1>         ja      VIDEO_RETURN
9605 00002F8F 6650                <1>         push    ax
9606 00002F91 6652                <1>         push    dx
9607 00002F93 66BADA03            <1>         mov     dx, 3DAh ; VGAREG_ACTL_RESET
9608 00002F97 EC                  <1>         in      al, dx
9609 00002F98 66BAC003            <1>         mov     dx, 3C0h ; VGAREG_ACTL_ADDRESS
9610 00002F9C 88D8                <1>         mov     al, bl
9611 00002F9E EE                  <1>         out     dx, al
9612 00002F9F 88F8                <1>         mov     al, bh
9613 00002FA1 EE                  <1>         out     dx, al
9614 00002FA2 B020                <1>         mov     al, 20h
9615 00002FA4 EE                  <1>         out     dx, al
9616                              <1>         ; ifdef VBOX
9617 00002FA5 66BADA03            <1>         mov     dx, 3DAh ; VGAREG_ACTL_RESET
9618 00002FA9 EC                  <1>         in      al, dx
9619                              <1>         ; endif ; VBOX
9620 00002FAA 665A                <1>         pop     dx
9621 00002FAC 6658                <1>         pop     ax
9622                              <1> ;no_actl_reg1:
9623 00002FAE E9A1E5FFFF          <1>         jmp     VIDEO_RETURN
9624                              <1>
9625                              <1> set_overscan_border_color:
9626                              <1>         ; 10/08/2016
9627                              <1>         ; Set Overscan/Border Color Register
9628                              <1>         ; BH = 6-bit RGB color to display
9629                              <1>         ;      for that attribute
9630                              <1>
9631 00002FB3 B311                <1>         mov     bl, 11h
9632 00002FB5 EBCF                <1>         jmp     short set_single_palette_reg
9633                              <1>
9634                              <1> set_all_palette_reg:
9635                              <1>         ; 10/08/2016
9636                              <1>         ; Set All Palette Registers and Overscan
9637                              <1>         ; EDX = Address of 17 bytes;
9638                              <1>         ; an rgbRGB value for each of 16 palette
9639                              <1>         ; registers plus one for the border.
9640                              <1>
9641 00002FB7 89D6                <1>         mov     esi, edx ; user buffer
9642 00002FB9 B911000000          <1>         mov     ecx, 17
9643 00002FBE 89E7                <1>         mov     edi, esp
9644 00002FC0 83EC14              <1>         sub     esp, 20
9645 00002FC3 E82FB90000          <1>         call    transfer_from_user_buffer
9646                              <1>         ;jc     VIDEO_RETURN
9647                              <1>
9648 00002FC8 66BADA03            <1>         mov     dx, 3DAh ; VGAREG_ACTL_RESET
9649 00002FCC EC                  <1>         in      al, dx
9650 00002FCD B100                <1>         mov     cl, 0
9651 00002FCF 66BAC003            <1>         mov     dx, 3C0h ; VGAREG_ACTL_ADDRESS
9652                              <1> set_palette_loop:
9653 00002FD3 88C8                <1>         mov     al, cl
9654 00002FD5 EE                  <1>         out     dx, al
9655 00002FD6 8A07                <1>         mov     al, [edi]
9656 00002FD8 EE                  <1>         out     dx, al
9657 00002FD9 47                  <1>         inc     edi
9658 00002FDA FEC1                <1>         inc     cl
9659 00002FDC 80F910              <1>         cmp     cl, 10h
9660 00002FDF 75F2                <1>         jne     short set_palette_loop
9661 00002FE1 B011                <1>         mov     al, 11h
9662 00002FE3 EE                  <1>         out     dx, al
9663 00002FE4 8A07                <1>         mov     al, [edi]
9664 00002FE6 EE                  <1>         out     dx, al
9665 00002FE7 B020                <1>         mov     al, 20h
9666 00002FE9 EE                  <1>         out     dx, al
9667                              <1>         ; ifdef VBOX
9668 00002FEA 66BADA03            <1>         mov     dx, 3DAh ; VGAREG_ACTL_RESET
9669 00002FEE EC                  <1>         in      al, dx
9670                              <1>         ; endif ; VBOX
9671 00002FEF 83C414              <1>         add     esp, 20
9672 00002FF2 E95DE5FFFF          <1>         jmp     VIDEO_RETURN
9673                              <1>
9674                              <1> toggle_intensity:
9675                              <1>         ; 10/08/2016
9676                              <1>         ; Select Foreground Blink or Bold Background
9677                              <1>         ; BL = 00h = enable bold backgrounds
9678                              <1>         ;          (16 background colors)
9679                              <1>         ;      01h = enable blinking foreground
9680                              <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
0000301A 66BADA03               <1>         mov    dx, 3DAh ; VGAREG_ACTL_RESET
0000301E EC                     <1>         in     al, dx
                                <1>         ; endif ; VBOX
0000301F E930E5FFFF             <1>         jmp    VIDEO_RETURN
                                <1>
                                <1> get_single_palette_reg:
                                <1>         ; 10/08/2016
                                <1>         ; Read One Palette Register
                                <1>           ; INPUT:
                                <1>         ; BL = Palette register to read (00h-0Fh)
                                <1>         ; OUTPUT:
                                <1>         ; BH = Current rgbRGB value of specified register
                                <1>         ;      for that attribute
                                <1>
00003024 80FB14                 <1>         cmp    bl, 14h
                                <1>         ;ja    short no_actl_reg2
00003027 0F8727E5FFFF           <1>         ja     VIDEO_RETURN
                                <1>
0000302D 66BADA03               <1>         mov    dx, 3DAh ; VGAREG_ACTL_RESET
00003031 EC                     <1>         in     al, dx
00003032 66BAC003               <1>         mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
00003036 88D8                   <1>         mov    al, bl
00003038 EE                     <1>         out    dx, al
00003039 66BAC103               <1>         mov    dx, 3C1h ; VGAREG_ACTL_READ_DATA
0000303D EC                     <1>         in     al, dx
0000303E 8844240D               <1>         mov    [esp+13], al ; bh
00003042 66BADA03               <1>         mov    dx, 3DAh ; VGAREG_ACTL_RESET
00003046 EC                     <1>         in     al, dx
00003047 66BAC003               <1>         mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
0000304B B020                   <1>         mov    al, 20h
0000304D EE                     <1>         out    dx, al
                                <1>         ; ifdef VBOX
0000304E 66BADA03               <1>         mov    dx, 3DAh ; VGAREG_ACTL_RESET
00003052 EC                     <1>         in     al, dx
                                <1>         ; endif ; VBOX
00003053 E9FCE4FFFF             <1>         jmp    VIDEO_RETURN
                                <1>
                                <1> read_overscan_border_color:
                                <1>         ; 10/08/2016
                                <1>         ; Read Overscan Register
                                <1>         ; OUTPUT:
                                <1>         ; BH = current rgbRGB value
                                <1>         ;      of the overscan/border register
                                <1>
00003058 B311                   <1>         mov    bl, 11h
0000305A EBC8                   <1>         jmp    short get_single_palette_reg
                                <1>
                                <1> get_all_palette_reg:
                                <1>         ; 10/08/2016
                                <1>         ; Read All Palette Registers
                                <1>         ; EDX = Address of 17-byte buffer
                                <1>         ;      to receive data
                                <1>
0000305C 89D7                   <1>         mov    edi, edx
0000305E 89E3                   <1>         mov    ebx, esp
00003060 89DE                   <1>         mov    esi, ebx
00003062 83EC14                 <1>         sub    esp, 20
                                <1>
00003065 B100                   <1>         mov    cl, 0
                                <1> get_palette_loop:
00003067 66BADA03               <1>         mov    dx, 3DAh ; VGAREG_ACTL_RESET
0000306B EC                     <1>         in     al, dx
0000306C 66BAC003               <1>         mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
00003070 88C8                   <1>         mov    al, cl
00003072 EE                     <1>         out    dx, al
00003073 66BAC103               <1>         mov    dx, 3C1h ; VGAREG_ACTL_READ_DATA
00003077 EC                     <1>         in     al, dx
00003078 8803                   <1>         mov    [ebx], al
0000307A 43                     <1>         inc    ebx
0000307B FEC1                   <1>         inc    cl
0000307D 80F910                 <1>         cmp    cl, 10h
00003080 75E5                   <1>         jne    short get_palette_loop
00003082 66BADA03               <1>         mov    dx, 3DAh ; VGAREG_ACTL_RESET
00003086 EC                     <1>         in     al, dx
00003087 66BAC003               <1>         mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
0000308B B011                   <1>         mov    al, 11h
0000308D EE                     <1>         out    dx, al
0000308E 66BAC103               <1>         mov    dx, 3C1h ; VGAREG_ACTL_READ_DATA
00003092 EC                     <1>         in     al, dx
00003093 8803                   <1>         mov    [ebx], al
00003095 66BADA03               <1>         mov    dx, 3DAh ; VGAREG_ACTL_RESET
00003099 EC                     <1>         in     al, dx
0000309A 66BAC003               <1>         mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
0000309E B020                   <1>         mov    al, 20h
000030A0 EE                     <1>         out    dx, al
                                <1>         ; ifdef VBOX
```

```
9784 000030A1 66BADA03            <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
9785 000030A5 EC                  <1>       in     al, dx
9786                              <1>       ; endif ; VBOX
9787                              <1>
9788 000030A6 B911000000          <1>       mov    ecx, 17 ; transfer (byte) count
9789                              <1>       ; ESI = source address in system space
9790                              <1>       ; EDI = user's buffer address
9791 000030AB E8FDB70000          <1>       call   transfer_to_user_buffer
9792                              <1>
9793 000030B0 83C414              <1>       add    esp, 20
9794 000030B3 E99CE4FFFF          <1>       jmp    VIDEO_RETURN
9795                              <1>
9796                              <1> set_single_dac_reg:
9797                              <1>       ; 10/08/2016
9798                              <1>       ; Set One DAC Color Register
9799                              <1>       ; BX = color register to set (0-255)
9800                              <1>         ; CH = green value (00h-3Fh)
9801                              <1>         ; CL = blue value  (00h-3Fh)
9802                              <1>         ; DH = red value   (00h-3Fh)
9803                              <1>
9804 000030B8 6652                <1>       push   dx
9805 000030BA 66BAC803            <1>       mov    dx, 3C8h ; VGAREG_DAC_WRITE_ADDRESS
9806 000030BE 88D8                <1>       mov    al, bl
9807 000030C0 EE                  <1>       out    dx, al
9808                              <1>       ;mov   dx, 3C9h ; VGAREG_DAC_DATA
9809 000030C1 6642                <1>       inc    dx
9810 000030C3 6658                <1>       pop    ax
9811 000030C5 88E0                <1>       mov    al, ah
9812 000030C7 EE                  <1>       out    dx, al
9813 000030C8 88E8                <1>       mov    al, ch
9814 000030CA EE                  <1>       out    dx, al
9815 000030CB 88C8                <1>       mov    al, cl
9816 000030CD EE                  <1>       out    dx, al
9817 000030CE E981E4FFFF          <1>       jmp    VIDEO_RETURN
9818                              <1>
9819                              <1> set_all_dac_reg:
9820                              <1>       ; 12/08/2016
9821                              <1>       ; 11/08/2016
9822                              <1>       ; 10/08/2016
9823                              <1>       ; Set a Block of DAC Color Register
9824                              <1>       ; BX = first DAC register to set (0-00FFh)
9825                              <1>       ; ECX = number of registers to set (0-00FFh)
9826                              <1>       ; EDX = addr of a table of R,G,B values
9827                              <1>       ;     (it will be CX*3 bytes long)
9828                              <1>
9829 000030D3 89D6                <1>       mov    esi, edx ; user buffer
9830 000030D5 89CA                <1>       mov    edx, ecx
9831 000030D7 66D1E1              <1>       shl    cx, 1 ; *2
9832 000030DA 01D1                <1>       add    ecx, edx ; ecx = 3*ecx
9833 000030DC 89E5                <1>       mov    ebp, esp
9834 000030DE 89EF                <1>       mov    edi, ebp
9835 000030E0 29CF                <1>       sub    edi, ecx
9836 000030E2 6683E7FC            <1>       and    di, 0FFFCh ; (dword alignment)
9837 000030E6 89FC                <1>       mov    esp, edi
9838 000030E8 E80AB80000          <1>       call   transfer_from_user_buffer
9839                              <1>       ;jc    VIDEO_RETURN
9840                              <1>
9841 000030ED 89D1                <1>       mov    ecx, edx
9842 000030EF 66BAC803            <1>       mov    dx, 3C8h ; VGAREG_DAC_WRITE_ADDRESS
9843 000030F3 88D8                <1>       mov    al, bl
9844 000030F5 EE                  <1>       out    dx, al
9845 000030F6 66BAC903            <1>       mov    dx, 3C9h ; VGAREG_DAC_DATA
9846                              <1> set_dac_loop:
9847 000030FA 8A07                <1>       mov    al, [edi]
9848 000030FC EE                  <1>       out    dx, al
9849 000030FD 47                  <1>       inc    edi
9850 000030FE 8A07                <1>       mov    al, [edi]
9851 00003100 EE                  <1>       out    dx, al
9852 00003101 47                  <1>       inc    edi
9853 00003102 8A07                <1>       mov    al, [edi]
9854 00003104 EE                  <1>       out    dx, al
9855 00003105 47                  <1>       inc    edi
9856 00003106 6649                <1>       dec    cx
9857 00003108 75F0                <1>       jnz    short set_dac_loop
9858 0000310A 89EC                <1>       mov    esp, ebp
9859 0000310C E943E4FFFF          <1>       jmp    VIDEO_RETURN
9860                              <1>
9861                              <1> select_video_dac_color_page:
9862                              <1>       ; 10/08/2016
9863                              <1>       ; DAC Color Paging Functions
9864                              <1>       ; BL = 00H = select color paging mode
9865                              <1>         ;       BH = paging mode
9866                              <1>         ;            00h = 4 blocks of 64 registers
9867                              <1>         ;            01h = 16 blocks of 16 registers
9868                              <1>       ; BL = 01H = activate color page
9869                              <1>         ;       BH = DAC color page number
9870                              <1>         ;            00h-03h (4-page/64-reg mode)
9871                              <1>         ;            00h-0Fh (16-page/16-reg mode)
9872                              <1>
9873 00003111 66BADA03            <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
9874 00003115 EC                  <1>       in     al, dx
9875 00003116 66BAC003            <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
9876 0000311A B010                <1>       mov    al, 10h
9877 0000311C EE                  <1>       out    dx, al
9878 0000311D 66BAC103            <1>       mov    dx, 3C1h ; VGAREG_ACTL_READ_DATA
9879 00003121 EC                  <1>       in     al, dx
9880 00003122 80E301              <1>       and    bl, 01h
9881 00003125 750E                <1>       jnz    short set_dac_page
9882 00003127 247F                <1>       and    al, 07Fh
9883 00003129 C0E707              <1>       shl    bh, 7
9884 0000312C 08F8                <1>       or     al, bh
9885 0000312E 66BAC003            <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
9886 00003132 EE                  <1>       out    dx, al
```

```
9887 00003133 EB1D              <1>       jmp    short set_actl_normal
9888                            <1> set_dac_page:
9889 00003135 6650              <1>       push   ax
9890 00003137 66BADA03          <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
9891 0000313B EC                <1>       in     al, dx
9892 0000313C 66BAC003          <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
9893 00003140 B014              <1>       mov    al, 14h
9894 00003142 EE                <1>       out    dx, al
9895 00003143 6658              <1>       pop    ax
9896 00003145 2480              <1>       and    al, 80h
9897 00003147 7503              <1>       jnz    short set_dac_16_page
9898 00003149 C0E702            <1>       shl    bh, 2
9899                            <1> set_dac_16_page:
9900 0000314C 80E70F            <1>       and    bh, 0Fh
9901 0000314F 88F8              <1>       mov    al, bh
9902 00003151 EE                <1>       out    dx, al
9903                            <1> set_actl_normal:
9904 00003152 B020              <1>       mov    al, 20h
9905 00003154 EE                <1>       out    dx, al
9906                            <1>       ; ifdef VBOX
9907 00003155 66BADA03          <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
9908 00003159 EC                <1>       in     al, dx
9909                            <1>       ; endif ; VBOX
9910 0000315A E9F5E3FFFF        <1>       jmp    VIDEO_RETURN
9911                            <1>
9912                            <1> read_single_dac_reg:
9913                            <1>       ; 10/08/2016
9914                            <1>       ; Read One DAC Color Register
9915                            <1>       ; INPUT:
9916                            <1>       ; BX = color register to read (0-255)
9917                            <1>       ; OUTPUT:
9918                            <1>       ; CH = green value (00h-3Fh)
9919                            <1>         ; CL = blue value  (00h-3Fh)
9920                            <1>         ; DH = red value   (00h-3Fh)
9921                            <1>
9922 0000315F 66BAC703          <1>       mov    dx, 3C7h ; VGAREG_DAC_READ_ADDRESS
9923 00003163 88D8              <1>       mov    al, bl
9924 00003165 EE                <1>       out    dx, al
9925 00003166 66BAC903          <1>       mov    dx, 3C9h ; VGAREG_DAC_DATA
9926 0000316A EC                <1>       in     al, dx
9927 0000316B 88442415          <1>       mov    [esp+21], al ; dh
9928 0000316F EC                <1>       in     al, dx
9929 00003170 88C5              <1>       mov    ch, al
9930 00003172 EC                <1>       in     al, dx
9931 00003173 88C1              <1>       mov    cl, al
9932 00003175 66894C2410        <1>       mov    [esp+16], cx ; cx
9933 0000317A E9D5E3FFFF        <1>       jmp    VIDEO_RETURN
9934                            <1>
9935                            <1> read_all_dac_reg:
9936                            <1>       ; 12/08/2016
9937                            <1>       ; 11/08/2016
9938                            <1>       ; 10/08/2016
9939                            <1>       ; Read a Block of DAC Color Registers
9940                            <1>         ; BX = first DAC register to read (0-00FFh)
9941                            <1>         ; ECX = number of registers to read (0-00FFh)
9942                            <1>         ; EDX = addr of a buffer to hold R,G,B values
9943                            <1>       ;     (CX*3 bytes long)
9944                            <1>
9945 0000317F 89D7              <1>       mov    edi, edx ; user buffer
9946 00003181 89CA              <1>       mov    edx, ecx
9947 00003183 66D1E2            <1>       shl    dx, 1 ; *2
9948 00003186 01CA              <1>       add    edx, ecx ; edx = 3*ecx
9949 00003188 89E5              <1>       mov    ebp, esp
9950 0000318A 89EE              <1>       mov    esi, ebp
9951 0000318C 29D6              <1>       sub    esi, edx
9952 0000318E 6683E6FC          <1>       and    si, 0FFFCh ; (dword alignment)
9953 00003192 89F4              <1>       mov    esp, esi
9954 00003194 52                <1>       push   edx ; 3*ecx
9955 00003195 66BAC703          <1>       mov    dx, 3C7h ; VGAREG_DAC_READ_ADDRESS
9956 00003199 88D8              <1>       mov    al, bl
9957 0000319B EE                <1>       out    dx, al
9958 0000319C 66BAC903          <1>       mov    dx, 3C9h ; VGAREG_DAC_DATA
9959 000031A0 89F3              <1>       mov    ebx, esi
9960                            <1> read_dac_loop:
9961 000031A2 EC                <1>       in     al, dx
9962 000031A3 8803              <1>       mov    [ebx], al
9963 000031A5 43                <1>       inc    ebx
9964 000031A6 EC                <1>       in     al, dx
9965 000031A7 8803              <1>       mov    [ebx], al
9966 000031A9 43                <1>       inc    ebx
9967 000031AA EC                <1>       in     al, dx
9968 000031AB 8803              <1>       mov    [ebx], al
9969 000031AD 43                <1>       inc    ebx
9970 000031AE 6649              <1>       dec    cx
9971 000031B0 75F0              <1>       jnz    short read_dac_loop
9972 000031B2 59                <1>       pop    ecx ; 3*ecx
9973                            <1>       ; ECX = transfer (byte) count
9974                            <1>       ; ESI = source address in system space
9975                            <1>       ; EDI = user's buffer address
9976 000031B3 E8F5B60000        <1>       call   transfer_to_user_buffer
9977 000031B8 89EC              <1>       mov    esp, ebp
9978 000031BA E995E3FFFF        <1>       jmp    VIDEO_RETURN
9979                            <1>
9980                            <1> set_pel_mask:
9981                            <1>       ; 10/08/2016
9982                            <1>       ; BL = mask value
9983 000031BF 66BAC603          <1>       mov    dx, 3C6h ; VGAREG_PEL_MASK
9984 000031C3 88D8              <1>       mov    al, bl
9985 000031C5 EE                <1>       out    dx, al
9986 000031C6 E989E3FFFF        <1>       jmp    VIDEO_RETURN
9987                            <1>
9988                            <1> read_pel_mask:
9989                            <1>       ; 10/08/2016
```

```
 9990                                   <1>       ; Output: BL = mask value
 9991 000031CB 66BAC603                 <1>       mov    dx, 3C6h ; VGAREG_PEL_MASK
 9992 000031CF EC                       <1>       in     al, dx
 9993 000031D0 8844240C                 <1>       mov    [esp+12], al ; bl
 9994 000031D4 E97BE3FFFF               <1>       jmp    VIDEO_RETURN
 9995                                   <1>
 9996                                   <1> read_video_dac_state:
 9997                                   <1>       ; 10/08/2016
 9998                                   <1>       ; Query DAC Color Paging State
 9999                                   <1>       ; Output:
10000                                   <1>       ; BH = current active DAC color page
10001                                   <1>         ; BL = current active DAC paging mode
10002                                   <1>
10003 000031D9 66BADA03                <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
10004 000031DD EC                      <1>       in     al, dx
10005 000031DE 66BAC003                <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
10006 000031E2 B010                    <1>       mov    al, 10h
10007 000031E4 EE                      <1>       out    dx, al
10008 000031E5 66BAC103                <1>       mov    dx, 3C1h ; VGAREG_ACTL_READ_DATA
10009 000031E9 EC                      <1>       in     al, dx
10010 000031EA 88C3                    <1>       mov    bl, al
10011 000031EC C0EB07                  <1>       shr    bl, 7
10012 000031EF 66BADA03                <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
10013 000031F3 EC                      <1>       in     al, dx
10014 000031F4 66BAC003                <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
10015 000031F8 B014                    <1>       mov    al, 14h
10016 000031FA EE                      <1>       out    dx, al
10017 000031FB 66BAC103                <1>       mov    dx, 3C1h ; VGAREG_ACTL_READ_DATA
10018 000031FF EC                      <1>       in     al, dx
10019 00003200 88C7                    <1>       mov    bh, al
10020 00003202 80E70F                  <1>       and    bh, 0Fh
10021 00003205 F6C301                  <1>       test   bl, 01
10022 00003208 7503                    <1>       jnz    short get_dac_16_page
10023 0000320A C0EF02                  <1>       shr    bh, 2
10024                                   <1> get_dac_16_page:
10025 0000320D 66BADA03                <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
10026 00003211 EC                      <1>       in     al, dx
10027 00003212 66BAC003                <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
10028 00003216 B020                    <1>       mov    al, 20h
10029 00003218 EE                      <1>       out    dx, al
10030                                   <1>       ; ifdef VBOX
10031 00003219 66BADA03                <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
10032 0000321D EC                      <1>       in     al, dx
10033                                   <1>       ; endif ; VBOX
10034 0000321E 66895C240C              <1>       mov    [esp+12], bx ; bx
10035 00003223 E92CE3FFFF              <1>       jmp    VIDEO_RETURN
10036                                   <1>
10037                                   <1> ; % include 'vidata.s' ; VIDEO DATA
10038                                   <1>
10039                                   <1> ; /// End Of VIDEO FUNCTIONS ///
10040
10041                                       setup_rtc_int:
10042                                       ; source: http://wiki.osdev.org/RTC
10043 00003228 FA                              cli          ; disable interrupts
10044                                         ; default int frequency is 1024 Hz (Lower 4 bits of register A is 0110b or 6)
10045                                         ; in order to change this ...
10046                                         ; frequency  = 32768 >> (rate-1) --> 32768 >> 5 = 1024
10047                                         ; (rate must be above 2 and not over 15)
10048                                         ; new rate = 15 --> 32768 >> (15-1) = 2 Hz
10049 00003229 B08A                          mov    al, 8Ah
10050 0000322B E670                          out    70h, al ; set index to register A, disable NMI
10051 0000322D 90                            nop
10052 0000322E E471                          in     al, 71h ; get initial value of register A
10053 00003230 88C4                          mov    ah, al
10054 00003232 80E4F0                        and    ah, 0F0h
10055 00003235 B08A                          mov    al, 8Ah
10056 00003237 E670                          out    70h, al ; reset index to register A
10057 00003239 88E0                          mov    al, ah
10058 0000323B 0C0F                          or     al, 0Fh      ; new rate (0Fh -> 15)
10059 0000323D E671                          out    71h, al ; write only our rate to A. Note, rate is the bottom 4 bits.
10060                                         ; enable RTC interrupt
10061 0000323F B08B                          mov    al, 8Bh ;
10062 00003241 E670                          out    70h, al ; select register B and disable NMI
10063 00003243 90                            nop
10064 00003244 E471                          in     al, 71h ; read the current value of register B
10065 00003246 88C4                          mov    ah, al  ;
10066 00003248 B08B                          mov    al, 8Bh ;
10067 0000324A E670                          out    70h, al ; set the index again (a read will reset the index to register B)
10068 0000324C 88E0                          mov    al, ah  ;
10069 0000324E 0C40                          or     al, 40h ;
10070 00003250 E671                          out    71h, al ; write the previous value ORed with 0x40. This turns on bit 6 of register B
10071 00003252 FB                            sti
10072 00003253 C3                            retn
10073
10074                                   ; Write memory information
10075                                   ; 29/01/2016
10076                                   ; 06/11/2014
10077                                   ; 14/08/2015
10078                                   memory_info:
10079 00003254 A1[24520100]                  mov    eax, [memory_size] ; in pages
10080 00003259 50                            push   eax
10081 0000325A C1E00C                        shl    eax, 12               ; in bytes
10082 0000325D BB0A000000                    mov    ebx, 10
10083 00003262 89D9                          mov    ecx, ebx        ; 10
10084 00003264 BE[C9120100]                  mov    esi, mem_total_b_str
10085 00003269 E8BD000000                    call   bintdstr
10086 0000326E 58                            pop    eax
10087 0000326F B107                          mov    cl, 7
10088 00003271 BE[ED120100]                  mov    esi, mem_total_p_str
10089 00003276 E8B0000000                    call   bintdstr
10090                                        ; 14/08/2015
10091 0000327B E8C8000000                    call   calc_free_mem
10092                                        ; edx = calculated free pages
```

```
                                        ; ecx = 0
00003280 A1[28520100]                   mov   eax, [free_pages]
00003285 39D0                           cmp   eax, edx ; calculated free mem value
                                        ; and initial free mem value are same or not?
00003287 751D                           jne   short pmim ; print mem info with '?' if not
00003289 52                             push  edx ; free memory in pages
                                        ;mov  eax, edx
0000328A C1E00C                         shl   eax, 12 ; convert page count
                                              ; to byte count
0000328D B10A                           mov   cl, 10
0000328F BE[0D130100]                   mov   esi, free_mem_b_str
00003294 E892000000                     call  bintdstr
00003299 58                             pop   eax
0000329A B107                           mov   cl, 7
0000329C BE[31130100]                   mov   esi, free_mem_p_str
000032A1 E885000000                     call  bintdstr
                                pmim:
000032A6 BE[B7120100]                   mov   esi, msg_memory_info
                                        ;
000032AB B407                           mov   ah, 07h ; Black background,
                                              ; light gray forecolor
                                print_kmsg: ; 29/01/2016
000032AD 8825[4F520100]                 mov   [ccolor], ah
                                pkmsg_loop:
000032B3 AC                             lodsb
000032B4 08C0                           or    al, al
000032B6 7410                           jz    short pkmsg_ok
000032B8 56                             push  esi
                                        ; 13/05/2016
000032B9 0FB61D[4F520100]               movzx ebx, byte [ccolor]
                                              ; Video page 0 (bh=0)
000032C0 E8EDE9FFFF                     call  _write_tty
000032C5 5E                             pop   esi
000032C6 EBEB                           jmp   short pkmsg_loop
                                pkmsg_ok:
000032C8 C3                             retn

                                ; Convert binary number to hexadecimal string
                                ; 10/05/2015
                                ; dsectpm.s (28/02/2015)
                                ; Retro UNIX 386 v1 - Kernel v0.2.0.6
                                ; 01/12/2014
                                ; 25/11/2014
                                ;
                                bytetohex:
                                        ; INPUT ->
                                        ;     AL = byte (binary number)
                                        ; OUTPUT ->
                                        ;     AX = hexadecimal string
                                        ;
000032C9 53                             push  ebx
000032CA 31DB                           xor   ebx, ebx
000032CC 88C3                           mov   bl, al
000032CE C0EB04                         shr   bl, 4
000032D1 8A9B[1B330000]                 mov   bl, [ebx+hexchrs]
000032D7 86D8                           xchg  bl, al
000032D9 80E30F                         and   bl, 0Fh
000032DC 8AA3[1B330000]                 mov   ah, [ebx+hexchrs]
000032E2 5B                             pop   ebx
000032E3 C3                             retn

                                wordtohex:
                                        ; INPUT ->
                                        ;     AX = word (binary number)
                                        ; OUTPUT ->
                                        ;     EAX = hexadecimal string
                                        ;
000032E4 53                             push  ebx
000032E5 31DB                           xor   ebx, ebx
000032E7 86E0                           xchg  ah, al
000032E9 6650                           push  ax
000032EB 88E3                           mov   bl, ah
000032ED C0EB04                         shr   bl, 4
000032F0 8A83[1B330000]                 mov   al, [ebx+hexchrs]
000032F6 88E3                           mov   bl, ah
000032F8 80E30F                         and   bl, 0Fh
000032FB 8AA3[1B330000]                 mov   ah, [ebx+hexchrs]
00003301 C1E010                         shl   eax, 16
00003304 6658                           pop   ax
00003306 5B                             pop   ebx
00003307 EBC0                           jmp   short bytetohex
                                        ;mov  bl, al
                                        ;shr  bl, 4
                                        ;mov  bl, [ebx+hexchrs]
                                        ;xchg bl, al
                                        ;and  bl, 0Fh
                                        ;mov  ah, [ebx+hexchrs]
                                        ;pop  ebx
                                        ;retn

                                dwordtohex:
                                        ; INPUT ->
                                        ;     EAX = dword (binary number)
                                        ; OUTPUT ->
                                        ;     EDX:EAX = hexadecimal string
                                        ;
00003309 50                             push  eax
0000330A C1E810                         shr   eax, 16
0000330D E8D2FFFFFF                     call  wordtohex
00003312 89C2                           mov   edx, eax
00003314 58                             pop   eax
00003315 E8CAFFFFFF                     call  wordtohex
0000331A C3                             retn
```

```
10196
10197                                   ; 10/05/2015
10198                                   hex_digits:
10199                                   hexchrs:
10200 0000331B 303132333435363738-              db '0123456789ABCDEF'
10201 00003324 39414243444546
10202
10203                                   ; Convert binary number to decimal/numeric string
10204                                   ; 06/11/2014
10205                                   ; Temporary Code
10206                                   ;
10207
10208                                   bintdstr:
10209                                           ; EAX = binary number
10210                                           ; ESI = decimal/numeric string address
10211                                           ; EBX = divisor (10)
10212                                           ; ECX = string length (<=10)
10213 0000332B 01CE                            add     esi, ecx
10214                                   btdstr0:
10215 0000332D 4E                              dec     esi
10216 0000332E 31D2                            xor     edx, edx
10217 00003330 F7F3                            div     ebx
10218 00003332 80C230                          add     dl, 30h
10219 00003335 8816                            mov     [esi], dl
10220 00003337 FEC9                            dec     cl
10221 00003339 740C                            jz      short btdstr2 ; 08/09/2016
10222 0000333B 09C0                            or      eax, eax
10223 0000333D 75EE                            jnz     short btdstr0
10224                                   btdstr1:
10225 0000333F 4E                              dec     esi
10226 00003340 C60620                            mov     byte [esi], 20h ; blank space
10227 00003343 FEC9                            dec     cl
10228 00003345 75F8                            jnz     short btdstr1
10229                                   btdstr2:
10230 00003347 C3                              retn
10231
10232                                   ; Calculate free memory pages on M.A.T.
10233                                   ; 06/11/2014
10234                                   ; Temporary Code
10235                                   ;
10236
10237                                   calc_free_mem:
10238 00003348 31D2                            xor     edx, edx
10239                                           ;xor    ecx, ecx
10240 0000334A 668B0D[38520100]                mov     cx, [mat_size] ; in pages
10241 00003351 C1E10A                          shl     ecx, 10       ; 1024 dwords per page
10242 00003354 BE00001000                      mov     esi, MEM_ALLOC_TBL
10243                                   cfm0:
10244 00003359 AD                              lodsd
10245 0000335A 51                              push    ecx
10246 0000335B B920000000                      mov     ecx, 32
10247                                   cfm1:
10248 00003360 D1E8                            shr     eax, 1
10249 00003362 7301                            jnc     short cfm2
10250 00003364 42                              inc     edx
10251                                   cfm2:
10252 00003365 E2F9                            loop    cfm1
10253 00003367 59                              pop     ecx
10254 00003368 E2EF                            loop    cfm0
10255 0000336A C3                              retn
10256
10257                                       %include 'diskio.s'  ; 07/03/2015
10258                                  <1> ; ****************************************************************************
10259                                  <1> ; TRDOS386.ASM (TRDOS 386 Kernel) - v2.0.0 - diskio.s
10260                                  <1> ; ----------------------------------------------------------------------------
10261                                  <1> ; Last Update: 09/12/2017
10262                                  <1> ; ----------------------------------------------------------------------------
10263                                  <1> ; Beginning: 24/01/2016
10264                                  <1> ; ----------------------------------------------------------------------------
10265                                  <1> ; Assembler: NASM version 2.11 (trdos386.s)
10266                                  <1> ; ----------------------------------------------------------------------------
10267                                  <1> ; Turkish Rational DOS
10268                                  <1> ; Operating System Project v2.0 by ERDOGAN TAN (Beginning: 04/01/2016)
10269                                  <1> ;
10270                                  <1> ; Derived from 'Retro UNIX 386 Kernel - v0.2.1.0' source code by Erdogan Tan
10271                                  <1> ; diskio.inc (22/08/2015)
10272                                  <1> ;
10273                                  <1> ; Derived from 'IBM PC-XT-286' BIOS source code (1986)
10274                                  <1> ; ****************************************************************************
10275                                  <1>
10276                                  <1> ; Retro UNIX 386 v1 Kernel - DISKIO.INC
10277                                  <1> ; Last Modification: 22/08/2015
10278                                  <1> ;     (Initialized Disk Parameters Data is in 'DISKDATA.INC')
10279                                  <1> ;     (Uninitialized Disk Parameters Data is in 'DISKBSS.INC')
10280                                  <1>
10281                                  <1> ; DISK I/O SYSTEM - Erdogan Tan (Retro UNIX 386 v1 project)
10282                                  <1>
10283                                  <1> ; //////// DISK I/O SYSTEM ///////////////
10284                                  <1>
10285                                  <1> ; 06/02/2015
10286                                  <1> diskette_io:
10287 0000336B 9C                      <1>       pushfd
10288 0000336C 0E                      <1>       push   cs
10289 0000336D E809000000              <1>       call   DISKETTE_IO_1
10290 00003372 C3                      <1>       retn
10291                                  <1>
10292                                  <1> ;;;;;; DISKETTE I/O ;;;;;;;;;;;;;;;;;;; 06/02/2015 ;;;
10293                                  <1> ;////////////////////////////////////////////////////
10294                                  <1>
10295                                  <1> ; DISKETTE I/O - Erdogan Tan (Retro UNIX 386 v1 project)
10296                                  <1> ; 20/02/2015
10297                                  <1> ; 06/02/2015 (unix386.s)
10298                                  <1> ; 16/12/2014 - 02/01/2015 (dsectrm2.s)
```

```
10299                                  <1> ;
10300                                  <1> ; Code (DELAY) modifications - AWARD BIOS 1999 (ADISK.EQU, COMMON.MAC)
10301                                  <1> ;
10302                                  <1> ; ADISK.EQU
10303                                  <1>
10304                                  <1> ;----- Wait control constants
10305                                  <1>
10306                                  <1> ;amount of time to wait while RESET is active.
10307                                  <1>
10308                                  <1> WAITCPU_RESET_ON   EQU    21           ;Reset on must last at least 14us
10309                                  <1>                                       ;at 250 KBS xfer rate.
10310                                  <1>                                       ;see INTEL MCS, 1985, pg. 5-456
10311                                  <1>
10312                                  <1> WAITCPU_FOR_STATUS EQU    100          ;allow 30 microseconds for
10313                                  <1>                                       ;status register to become valid
10314                                  <1>                                       ;before re-reading.
10315                                  <1>
10316                                  <1> ;After sending a byte to NEC, status register may remain
10317                                  <1> ;incorrectly set for 24 us.
10318                                  <1>
10319                                  <1> WAITCPU_RQM_LOW           EQU    24           ;number of loops to check for
10320                                  <1>                                       ;RQM low.
10321                                  <1>
10322                                  <1> ; COMMON.MAC
10323                                  <1> ;
10324                                  <1> ;     Timing macros
10325                                  <1> ;
10326                                  <1>
10327                                  <1> %macro             SIODELAY 0                ; SHORT IODELAY
10328                                  <1>             jmp short $+2
10329                                  <1> %endmacro
10330                                  <1>
10331                                  <1> %macro             IODELAY  0                ; NORMAL IODELAY
10332                                  <1>             jmp short $+2
10333                                  <1>             jmp short $+2
10334                                  <1> %endmacro
10335                                  <1>
10336                                  <1> %macro             NEWIODELAY 0
10337                                  <1>             out    0ebh,al
10338                                  <1> %endmacro
10339                                  <1>
10340                                  <1> ; (According to) AWARD BIOS 1999 - ATORGS.ASM (dw -> equ, db -> equ)
10341                                  <1> ;;; WAIT_FOR_MEM
10342                                  <1> ;WAIT_FDU_INT_LO   equ    017798       ; 2.5 secs in 30 micro units.
10343                                  <1> ;WAIT_FDU_INT_HI   equ    1
10344                                  <1> WAIT_FDU_INT_LH           equ    83334        ; 27/02/2015 (2.5 seconds waiting)
10345                                  <1> ;;; WAIT_FOR_PORT
10346                                  <1> ;WAIT_FDU_SEND_LO  equ    16667        ; .5 secons in 30 us units.
10347                                  <1> ;WAIT_FDU_SEND_HI  equ    0
10348                                  <1> WAIT_FDU_SEND_LH   equ    16667        ; 27/02/2015
10349                                  <1> ;Time to wait while waiting for each byte of NEC results = .5
10350                                  <1> ;seconds.  .5 seconds = 500,000 micros.  500,000/30 = 16,667.
10351                                  <1> ;WAIT_FDU_RESULTS_LO      equ    16667        ; .5 seconds in 30 micro units.
10352                                  <1> ;WAIT_FDU_RESULTS_HI      equ    0
10353                                  <1> WAIT_FDU_RESULTS_LH       equ    16667  ; 27/02/2015
10354                                  <1> ;;; WAIT_REFRESH
10355                                  <1> ;amount of time to wait for head settle, per unit in parameter
10356                                  <1> ;table = 1 ms.
10357                                  <1> WAIT_FDU_HEAD_SETTLE      equ    33           ; 1 ms in 30 micro units.
10358                                  <1>
10359                                  <1>
10360                                  <1> ; ///////////////// DISKETTE I/O ///////////////
10361                                  <1>
10362                                  <1> ; 11/12/2014 (copy from IBM PC-XT Model 286 BIOS - POSTEQU.INC)
10363                                  <1>
10364                                  <1> ;----------------------------------------
10365                                  <1> ;     EQUATES USED BY POST AND BIOS    :
10366                                  <1> ;----------------------------------------
10367                                  <1>
10368                                  <1> ;--------- 8042 KEYBOARD INTERFACE AND DIAGNOSTIC CONTROL REGISTERS ------------
10369                                  <1> ;PORT_A            EQU    060H         ; 8042 KEYBOARD SCAN CODE/CONTROL PORT
10370                                  <1> ;PORT_B            EQU    061H         ; PORT B READ/WRITE DIAGNOSTIC REGISTER
10371                                  <1> ;REFRESH_BIT EQU    00010000B    ; REFRESH TEST BIT
10372                                  <1>
10373                                  <1> ;----------------------------------------
10374                                  <1> ;     CMOS EQUATES FOR THIS SYSTEM     :
10375                                  <1> ;-------------------------------------------------------------------------------
10376                                  <1> ;CMOS_PORT   EQU    070H         ; I/O ADDRESS OF CMOS ADDRESS PORT
10377                                  <1> ;CMOS_DATA   EQU    071H         ; I/O ADDRESS OF CMOS DATA PORT
10378                                  <1> ;NMI         EQU    10000000B    ; DISABLE NMI INTERRUPTS MASK -
10379                                  <1>                                 ;  HIGH BIT OF CMOS LOCATION ADDRESS
10380                                  <1>
10381                                  <1> ;---------- CMOS TABLE LOCATION ADDRESS'S ## ---------------------------------
10382                                  <1> CMOS_DISKETTE      EQU    010H         ; DISKETTE DRIVE TYPE BYTE      ;
10383                                  <1> ;            EQU    011H         ; - RESERVED                    ;C
10384                                  <1> CMOS_DISK    EQU    012H         ; FIXED DISK TYPE BYTE                ;H
10385                                  <1> ;            EQU    013H         ; - RESERVED                    ;E
10386                                  <1> CMOS_EQUIP   EQU    014H         ; EQUIPMENT WORD LOW BYTE       ;C
10387                                  <1>
10388                                  <1> ;---------- DISKETTE EQUATES ------------------------------------------------
10389                                  <1> INT_FLAG     EQU    10000000B    ; INTERRUPT OCCURRENCE FLAG
10390                                  <1> DSK_CHG      EQU    10000000B    ; DISKETTE CHANGE FLAG MASK BIT
10391                                  <1> DETERMINED   EQU    00010000B    ; SET STATE DETERMINED IN STATE BITS
10392                                  <1> HOME         EQU    00010000B    ; TRACK 0 MASK
10393                                  <1> SENSE_DRV_ST EQU    00000100B    ; SENSE DRIVE STATUS COMMAND
10394                                  <1> TRK_SLAP     EQU    030H         ; CRASH STOP (48 TPI DRIVES)
10395                                  <1> QUIET_SEEK   EQU    00AH         ; SEEK TO TRACK 10
10396                                  <1> ;MAX_DRV     EQU    2            ; MAX NUMBER OF DRIVES
10397                                  <1> HD12_SETTLE  EQU    15           ; 1.2 M HEAD SETTLE TIME
10398                                  <1> HD320_SETTLE EQU    20           ; 320 K HEAD SETTLE TIME
10399                                  <1> MOTOR_WAIT   EQU    37           ; 2 SECONDS OF COUNTS FOR MOTOR TURN OFF
10400                                  <1>
10401                                  <1> ;---------- DISKETTE ERRORS -------------------------------------------------
```

```
10402                                  <1> ;TIME_OUT    EQU    080H         ; ATTACHMENT FAILED TO RESPOND
10403                                  <1> ;BAD_SEEK    EQU    040H         ; SEEK OPERATION FAILED
10404                                  <1> BAD_NEC      EQU    020H         ; DISKETTE CONTROLLER HAS FAILED
10405                                  <1> BAD_CRC      EQU    010H         ; BAD CRC ON DISKETTE READ
10406                                  <1> MED_NOT_FND  EQU    00CH         ; MEDIA TYPE NOT FOUND
10407                                  <1> DMA_BOUNDARY EQU    009H         ; ATTEMPT TO DMA ACROSS 64K BOUNDARY
10408                                  <1> BAD_DMA      EQU    008H         ; DMA OVERRUN ON OPERATION
10409                                  <1> MEDIA_CHANGE EQU    006H         ; MEDIA REMOVED ON DUAL ATTACH CARD
10410                                  <1> RECORD_NOT_FND     EQU    004H         ; REQUESTED SECTOR NOT FOUND
10411                                  <1> WRITE_PROTECT      EQU    003H         ; WRITE ATTEMPTED ON WRITE PROTECT DISK
10412                                  <1> BAD_ADDR_MARK      EQU    002H         ; ADDRESS MARK NOT FOUND
10413                                  <1> BAD_CMD      EQU    001H         ; BAD COMMAND PASSED TO DISKETTE I/O
10414                                  <1>
10415                                  <1> ;---------- DISK CHANGE LINE EQUATES ----------------------------------------
10416                                  <1> NOCHGLN      EQU    001H         ; NO DISK CHANGE LINE AVAILABLE
10417                                  <1> CHGLN        EQU    002H         ; DISK CHANGE LINE AVAILABLE
10418                                  <1>
10419                                  <1> ;---------- MEDIA/DRIVE STATE INDICATORS ------------------------------------
10420                                  <1> TRK_CAPA     EQU    00000001B    ; 80 TRACK CAPABILITY
10421                                  <1> FMT_CAPA     EQU    00000010B    ; MULTIPLE FORMAT CAPABILITY (1.2M)
10422                                  <1> DRV_DET      EQU    00000100B    ; DRIVE DETERMINED
10423                                  <1> MED_DET      EQU    00010000B    ; MEDIA DETERMINED BIT
10424                                  <1> DBL_STEP     EQU    00100000B    ; DOUBLE STEP BIT
10425                                  <1> RATE_MSK     EQU    11000000B    ; MASK FOR CLEARING ALL BUT RATE
10426                                  <1> RATE_500     EQU    00000000B    ; 500 KBS DATA RATE
10427                                  <1> RATE_300     EQU    01000000B    ; 300 KBS DATA RATE
10428                                  <1> RATE_250     EQU    10000000B    ; 250 KBS DATA RATE
10429                                  <1> STRT_MSK     EQU    00001100B    ; OPERATION START RATE MASK
10430                                  <1> SEND_MSK     EQU    11000000B    ; MASK FOR SEND RATE BITS
10431                                  <1>
10432                                  <1> ;---------- MEDIA/DRIVE STATE INDICATORS COMPATIBILITY ------------------------
10433                                  <1> M3D3U        EQU    00000000B    ; 360 MEDIA/DRIVE NOT ESTABLISHED
10434                                  <1> M3D1U        EQU    00000001B    ; 360 MEDIA,1.2DRIVE NOT ESTABLISHED
10435                                  <1> M1D1U        EQU    00000010B    ; 1.2 MEDIA/DRIVE NOT ESTABLISHED
10436                                  <1> MED_UNK      EQU    00000111B    ; NONE OF THE ABOVE
10437                                  <1>
10438                                  <1> ;---------- INTERRUPT EQUATES ------------------------------------------------
10439                                  <1> ;EOI         EQU    020H         ; END OF INTERRUPT COMMAND TO 8259
10440                                  <1> ;INTA00            EQU    020H         ; 8259 PORT
10441                                  <1> INTA01             EQU    021H         ; 8259 PORT
10442                                  <1> INTB00             EQU    0A0H         ; 2ND 8259
10443                                  <1> INTB01             EQU    0A1H         ;
10444                                  <1>
10445                                  <1> ;--------------------------------------------------------------------------------
10446                                  <1> DMA08        EQU    008H         ; DMA STATUS REGISTER PORT ADDRESS
10447                                  <1> DMA          EQU    000H         ; DMA CH.0 ADDRESS REGISTER PORT ADDRESS
10448                                  <1> DMA18        EQU    0D0H         ; 2ND DMA STATUS PORT ADDRESS
10449                                  <1> DMA1         EQU    0C0H         ; 2ND DMA CH.0 ADDRESS REGISTER ADDRESS
10450                                  <1> ;--------------------------------------------------------------------------------
10451                                  <1> ;TIMER             EQU    040H         ; 8254 TIMER - BASE ADDRESS
10452                                  <1>
10453                                  <1> ;--------------------------------------------------------------------------------
10454                                  <1> DMA_PAGE     EQU    081H         ; START OF DMA PAGE REGISTERS
10455                                  <1>
10456                                  <1> ; 06/02/2015 (unix386.s, protected mode modifications)
10457                                  <1> ; (unix386.s <-- dsectrm2.s)
10458                                  <1> ; 11/12/2014 (copy from IBM PC-XT Model 286 BIOS - DSEG.INC)
10459                                  <1>
10460                                  <1> ; 27/05/2016 - TRDOS 386 (TRDOS v2.0)
10461                                  <1> ; 10/12/2014
10462                                  <1> ;
10463                                  <1> ;int40h:
10464                                  <1> ;     pushf
10465                                  <1> ;     push   cs
10466                                  <1> ;     ;cli
10467                                  <1> ;     call   DISKETTE_IO_1
10468                                  <1> ;     retn
10469                                  <1>
10470                                  <1> ; DSKETTE ----- 04/21/86 DISKETTE BIOS
10471                                  <1> ; (IBM PC XT Model 286 System BIOS Source Code, 04-21-86)
10472                                  <1> ;
10473                                  <1>
10474                                  <1> ;-- INT13H ---------------------------------------------------------------------
10475                                  <1> ; DISKETTE I/O
10476                                  <1> ;     THIS INTERFACE PROVIDES ACCESS TO THE 5 1/4 INCH 360 KB,
10477                                  <1> ;     1.2 MB, 720 KB AND 1.44 MB DISKETTE DRIVES.
10478                                  <1> ; INPUT
10479                                  <1> ;     (AH) =  00H RESET DISKETTE SYSTEM
10480                                  <1> ;             HARD RESET TO NEC, PREPARE COMMAND, RECALIBRATE REQUIRED
10481                                  <1> ;             ON ALL DRIVES
10482                                  <1> ;--------------------------------------------------------------------------------
10483                                  <1> ;     (AH)= 01H  READ THE STATUS OF THE SYSTEM INTO (AH)
10484                                  <1> ;             @DISKETTE_STATUS FROM LAST OPERATION IS USED
10485                                  <1> ;--------------------------------------------------------------------------------
10486                                  <1> ;     REGISTERS FOR READ/WRITE/VERIFY/FORMAT
10487                                  <1> ;     (DL) - DRIVE NUMBER (0-1 ALLOWED, VALUE CHECKED)
10488                                  <1> ;     (DH) - HEAD NUMBER (0-1 ALLOWED, NOT VALUE CHECKED)
10489                                  <1> ;     (CH) - TRACK NUMBER (NOT VALUE CHECKED)
10490                                  <1> ;             MEDIA  DRIVE  TRACK NUMBER
10491                                  <1> ;             320/360      320/360          0-39
10492                                  <1> ;             320/360      1.2M       0-39
10493                                  <1> ;             1.2M   1.2M       0-79
10494                                  <1> ;             720K   720K       0-79
10495                                  <1> ;             1.44M  1.44M      0-79
10496                                  <1> ;     (CL) -       SECTOR NUMBER (NOT VALUE CHECKED, NOT USED FOR FORMAT)
10497                                  <1> ;             MEDIA  DRIVE  SECTOR NUMBER
10498                                  <1> ;             320/360      320/360           1-8/9
10499                                  <1> ;             320/360      1.2M        1-8/9
10500                                  <1> ;             1.2M   1.2M       1-15
10501                                  <1> ;             720K   720K       1-9
10502                                  <1> ;             1.44M  1.44M      1-18
10503                                  <1> ;     (AL)   NUMBER OF SECTORS (NOT VALUE CHECKED)
10504                                  <1> ;             MEDIA  DRIVE  MAX NUMBER OF SECTORS
```

```
10505                           <1> ;           320/360      320/360              8/9
10506                           <1> ;           320/360      1.2M          8/9
10507                           <1> ;           1.2M   1.2M         15
10508                           <1> ;           720K   720K         9
10509                           <1> ;           1.44M  1.44M        18
10510                           <1> ;
10511                           <1> ;    (ES:BX) - ADDRESS OF BUFFER (NOT REQUIRED FOR VERIFY)
10512                           <1> ;
10513                           <1> ;--------------------------------------------------------------------------
10514                           <1> ;    (AH)= 02H  READ THE DESIRED SECTORS INTO MEMORY
10515                           <1> ;--------------------------------------------------------------------------
10516                           <1> ;    (AH)= 03H  WRITE THE DESIRED SECTORS FROM MEMORY
10517                           <1> ;--------------------------------------------------------------------------
10518                           <1> ;    (AH)= 04H  VERIFY THE DESIRED SECTORS
10519                           <1> ;--------------------------------------------------------------------------
10520                           <1> ;    (AH)= 05H  FORMAT THE DESIRED TRACK
10521                           <1> ;          (ES,BX) MUST POINT TO THE COLLECTION OF DESIRED ADDRESS FIELDS
10522                           <1> ;          FOR THE      TRACK. EACH FIELD IS COMPOSED OF 4 BYTES, (C,H,R,N),
10523                           <1> ;          WHERE C = TRACK NUMBER, H=HEAD NUMBER, R = SECTOR NUMBER,
10524                           <1> ;          N= NUMBER OF BYTES PER SECTOR (00=128,01=256,02=512,03=1024),
10525                           <1> ;          THERE MUST BE ONE ENTRY FOR EVERY SECTOR ON THE TRACK.
10526                           <1> ;          THIS INFORMATION IS USED TO FIND THE REQUESTED SECTOR DURING
10527                           <1> ;          READ/WRITE ACCESS.
10528                           <1> ;          PRIOR TO FORMATTING A DISKETTE, IF THERE EXISTS MORE THAN
10529                           <1> ;          ONE SUPPORTED MEDIA FORMAT TYPE WITHIN THE DRIVE IN QUESTION,
10530                           <1> ;          THEN "SET DASD TYPE" (INT 13H, AH = 17H) OR 'SET MEDIA TYPE'
10531                           <1> ;          (INT 13H, AH =  18H) MUST BE CALLED TO SET THE DISKETTE TYPE
10532                           <1> ;          THAT IS TO BE FORMATTED. IF "SET DASD TYPE" OR "SET MEDIA TYPE"
10533                           <1> ;          IS NOT CALLED, THE FORMAT ROUTINE WILL ASSUME THE
10534                           <1> ;          MEDIA FORMAT TO BE THE MAXIMUM CAPACITY OF THE DRIVE.
10535                           <1> ;
10536                           <1> ;          THESE PARAMETERS OF DISK BASE MUST BE CHANGED IN ORDER TO
10537                           <1> ;          FORMAT THE FOLLOWING MEDIAS:
10538                           <1> ;          ----------------------------------------
10539                           <1> ;          : MEDIA  :     DRIVE      : PARM 1 : PARM 2 :
10540                           <1> ;          ----------------------------------------
10541                           <1> ;          : 320K  : 320K/360K/1.2M :  50H   :   8    :
10542                           <1> ;          : 360K  : 320K/360K/1.2M :  50H   :   9    :
10543                           <1> ;          : 1.2M  : 1.2M           :  54H   :  15    :
10544                           <1> ;          : 720K  : 720K/1.44M     :  50H   :   9    :
10545                           <1> ;          : 1.44M       : 1.44M          :  6CH   :  18    :
10546                           <1> ;          ----------------------------------------
10547                           <1> ;          NOTES: - PARM 1 = GAP LENGTH FOR FORMAT
10548                           <1> ;                 - PARM 2 = EOT (LAST SECTOR ON TRACK)
10549                           <1> ;                 - DISK BASE IS POINTED BY DISK POINTER LOCATED
10550                           <1> ;                  AT ABSOLUTE ADDRESS 0:78.
10551                           <1> ;                 - WHEN FORMAT OPERATIONS ARE COMPLETE, THE PARAMETERS
10552                           <1> ;                  SHOULD BE RESTORED TO THEIR RESPECTIVE INITIAL VALUES.
10553                           <1> ;--------------------------------------------------------------------------
10554                           <1> ;    (AH) = 08H READ DRIVE PARAMETERS
10555                           <1> ;    REGISTERS
10556                           <1> ;      INPUT
10557                           <1> ;        (DL) - DRIVE NUMBER (0-1 ALLOWED, VALUE CHECKED)
10558                           <1> ;         ** 27/05/2016 - TRDOS 386 (TRDOS v2.0) **
10559                           <1> ;           ** EBX = Buffer address for floppy disk parameters table **
10560                           <1> ;      OUTPUT
10561                           <1> ;        (ES:DI) POINTS TO DRIVE PARAMETER TABLE
10562                           <1> ;        *** TRDOS 386 note: floppy disk parameter table (16 bytes)
10563                           <1> ;        will be returned to user in EBX, buffer address *** 27/05/2016 ***
10564                           <1> ;
10565                           <1> ;        (CH) - LOW ORDER 8 OF 10 BITS MAXIMUM NUMBER OF TRACKS
10566                           <1> ;        (CL) - BITS 7 & 6 - HIGH ORDER TWO BITS OF MAXIMUM TRACKS
10567                           <1> ;               BITS 5 THRU 0 - MAXIMUM SECTORS PER TRACK
10568                           <1> ;        (DH) - MAXIMUM HEAD NUMBER
10569                           <1> ;        (DL) - NUMBER OF DISKETTE DRIVES INSTALLED
10570                           <1> ;        (BH) - 0
10571                           <1> ;        (BL) - BITS 7 THRU 4 - 0
10572                           <1> ;               BITS 3 THRU 0 - VALID DRIVE TYPE VALUE IN CMOS
10573                           <1> ;        (AX) - 0
10574                           <1> ;      UNDER THE FOLLOWING CIRCUMSTANCES:
10575                           <1> ;        (1) THE DRIVE NUMBER IS INVALID,
10576                           <1> ;        (2) THE DRIVE TYPE IS UNKNOWN AND CMOS IS NOT PRESENT,
10577                           <1> ;        (3) THE DRIVE TYPE IS UNKNOWN AND CMOS IS BAD,
10578                           <1> ;        (4) OR THE DRIVE TYPE IS UNKNOWN AND THE CMOS DRIVE TYPE IS INVALID
10579                           <1> ;        THEN ES,AX,BX,CX,DH,DI=0 ; DL=NUMBER OF DRIVES.
10580                           <1> ;        IF NO DRIVES ARE PRESENT THEN: ES,AX,BX,CX,DX,DI=0.
10581                           <1> ;        @DISKETTE_STATUS = 0 AND CY IS RESET.
10582                           <1> ;--------------------------------------------------------------------------
10583                           <1> ;    (AH)= 15H  READ DASD TYPE
10584                           <1> ;    OUTPUT REGISTERS
10585                           <1> ;    (AH) - ON RETURN IF CARRY FLAG NOT SET, OTHERWISE ERROR
10586                           <1> ;           00 - DRIVE NOT PRESENT
10587                           <1> ;           01 - DISKETTE, NO CHANGE LINE AVAILABLE
10588                           <1> ;           02 - DISKETTE, CHANGE LINE AVAILABLE
10589                           <1> ;           03 - RESERVED (FIXED DISK)
10590                           <1> ;    (DL) - DRIVE NUMBER (0-1 ALLOWED, VALUE CHECKED)
10591                           <1> ;--------------------------------------------------------------------------
10592                           <1> ;    (AH)= 16H  DISK CHANGE LINE STATUS
10593                           <1> ;    OUTPUT REGISTERS
10594                           <1> ;    (AH) - 00 - DISK CHANGE LINE NOT ACTIVE
10595                           <1> ;           06 - DISK CHANGE LINE ACTIVE & CARRY BIT ON
10596                           <1> ;    (DL) - DRIVE NUMBER (0-1 ALLOWED, VALUE CHECKED)
10597                           <1> ;--------------------------------------------------------------------------
10598                           <1> ;    (AH)= 17H  SET DASD TYPE FOR FORMAT
10599                           <1> ;    INPUT REGISTERS
10600                           <1> ;    (AL) - 00 - NOT USED
10601                           <1> ;           01 - DISKETTE 320/360K IN 360K DRIVE
10602                           <1> ;           02 - DISKETTE 360K IN 1.2M DRIVE
10603                           <1> ;           03 - DISKETTE 1.2M IN 1.2M DRIVE
10604                           <1> ;           04 - DISKETTE 720K IN 720K DRIVE
10605                           <1> ;    (DL) - DRIVE NUMBER (0-1 ALLOWED, VALUE CHECKED:
10606                           <1> ;           (DO NOT USE WHEN DISKETTE ATTACH CARD USED)
10607                           <1> ;--------------------------------------------------------------------------
```

```
10608                                  <1> ;     (AH)= 18H  SET MEDIA TYPE FOR FORMAT
10609                                  <1> ;     INPUT REGISTERS
10610                                  <1> ;     (CH) - LOW ORDER 8 OF 10 BITS MAXIMUM TRACKS
10611                                  <1> ;     (CL) - BITS 7 & 6 - HIGH ORDER TWO BITS OF MAXIMUM TRACKS
10612                                  <1> ;            BITS 5 THRU 0 - MAXIMUM SECTORS PER TRACK
10613                                  <1> ;     (DL) - DRIVE NUMBER (0-1 ALLOWED, VALUE CHACKED)
10614                                  <1> ;     OUTPUT REGISTERS:
10615                                  <1> ;     (ES:DI) - POINTER TO DRIVE PARAMETERS TABLE FOR THIS MEDIA TYPE,
10616                                  <1> ;              UNCHANGED IF (AH) IS NON-ZERO
10617                                  <1> ;     (AH) - 00H, CY = 0, TRACK AND SECTORS/TRACK COMBINATION IS SUPPORTED
10618                                  <1> ;          - 01H, CY = 1, FUNCTION IS NOT AVAILABLE
10619                                  <1> ;          - 0CH, CY = 1, TRACK AND SECTORS/TRACK COMBINATION IS NOT SUPPORTED
10620                                  <1> ;          - 80H, CY = 1, TIME OUT (DISKETTE NOT PRESENT)
10621                                  <1> ;--------------------------------------------------------------------------
10622                                  <1> ;     DISK CHANGE STATUS IS ONLY CHECKED WHEN A MEDIA SPECIFIED IS OTHER
10623                                  <1> ;     THAN 360 KB DRIVE. IF THE DISK CHANGE LINE IS FOUND TO BE
10624                                  <1> ;     ACTIVE THE FOLLOWING ACTIONS TAKE PLACE:
10625                                  <1> ;           ATTEMPT TO RESET DISK CHANGE LINE TO INACTIVE STATE.
10626                                  <1> ;           IF ATTEMPT SUCCEEDS SET DASD TYPE FOR FORMAT AND RETURN DISK
10627                                  <1> ;           CHANGE ERROR CODE
10628                                  <1> ;           IF ATTEMPT FAILS RETURN TIMEOUT ERROR CODE AND SET DASD TYPE
10629                                  <1> ;           TO A PREDETERMINED STATE INDICATING MEDIA TYPE UNKNOWN.
10630                                  <1> ;     IF THE DISK CHANGE LINE IN INACTIVE PERFORM SET DASD TYPE FOR FORMAT.
10631                                  <1> ;
10632                                  <1> ; DATA VARIABLE -- @DISK_POINTER
10633                                  <1> ;     DOUBLE WORD POINTER TO THE CURRENT SET OF DISKETTE PARAMETERS
10634                                  <1> ;--------------------------------------------------------------------------
10635                                  <1> ; OUTPUT FOR ALL FUNCTIONS
10636                                  <1> ;     AH = STATUS OF OPERATION
10637                                  <1> ;           STATUS BITS ARE DEFINED IN THE EQUATES FOR @DISKETTE_STATUS
10638                                  <1> ;           VARIABLE IN THE DATA SEGMENT OF THIS MODULE
10639                                  <1> ;     CY = 0 SUCCESSFUL OPERATION (AH=0 ON RETURN, EXCEPT FOR READ DASD
10640                                  <1> ;            TYPE AH=(15)).
10641                                  <1> ;     CY = 1 FAILED OPERATION (AH HAS ERROR REASON)
10642                                  <1> ;     FOR READ/WRITE/VERIFY
10643                                  <1> ;           DS,BX,DX,CX PRESERVED
10644                                  <1> ;     NOTE: IF AN ERROR IS REPORTED BY THE DISKETTE CODE, THE APPROPRIATE
10645                                  <1> ;           ACTION IS TO RESET THE DISKETTE, THEN RETRY THE OPERATION.
10646                                  <1> ;           ON READ ACCESSES, NO MOTOR START DELAY IS TAKEN, SO THAT
10647                                  <1> ;           THREE RETRIES ARE REQUIRED ON READS TO ENSURE THAT THE
10648                                  <1> ;           PROBLEM IS NOT DUE TO MOTOR START-UP.
10649                                  <1> ;--------------------------------------------------------------------------
10650                                  <1> ;
10651                                  <1> ; DISKETTE STATE MACHINE - ABSOLUTE ADDRESS 40:90 (DRIVE A) & 91 (DRIVE B)
10652                                  <1> ;
10653                                  <1> ;   ---------------------------------------------------------------
10654                                  <1> ;   |       |       |       |       |       |       |       |       |
10655                                  <1> ;   |   7   |   6   |   5   |   4   |   3   |   2   |   1   |   0   |
10656                                  <1> ;   |       |       |       |       |       |       |       |       |
10657                                  <1> ;   ---------------------------------------------------------------
10658                                  <1> ;     |       |       |       |       |       |       |       |
10659                                  <1> ;     |       |       |       |       |       -----------------
10660                                  <1> ;     |       |       |       |       |               |
10661                                  <1> ;     |       |       |       |    RESERVED           |
10662                                  <1> ;     |       |       |       |                 PRESENT STATE
10663                                  <1> ;     |       |       |       |       000: 360K IN 360K DRIVE UNESTABLISHED
10664                                  <1> ;     |       |       |       |       001: 360K IN 1.2M DRIVE UNESTABLISHED
10665                                  <1> ;     |       |       |       |       010: 1.2M IN 1.2M DRIVE UNESTABLISHED
10666                                  <1> ;     |       |       |       |       011: 360K IN 360K DRIVE ESTABLISHED
10667                                  <1> ;     |       |       |       |       100: 360K IN 1.2M DRIVE ESTABLISHED
10668                                  <1> ;     |       |       |       |       101: 1.2M IN 1.2M DRIVE ESTABLISHED
10669                                  <1> ;     |       |       |       |       110: RESERVED
10670                                  <1> ;     |       |       |       |       111: NONE OF THE ABOVE
10671                                  <1> ;     |       |       |       |
10672                                  <1> ;     |       |       |       ------>       MEDIA/DRIVE ESTABLISHED
10673                                  <1> ;     |       |       |
10674                                  <1> ;     |       |       --------------->     DOUBLE STEPPING REQUIRED (360K IN 1.2M
10675                                  <1> ;     |       |                       DRIVE)
10676                                  <1> ;     |       |
10677                                  <1> ;     ------------------------------>  DATA TRANSFER RATE FOR THIS DRIVE:
10678                                  <1> ;
10679                                  <1> ;                                       00: 500 KBS
10680                                  <1> ;                                       01: 300 KBS
10681                                  <1> ;                                       10: 250 KBS
10682                                  <1> ;                                       11: RESERVED
10683                                  <1> ;
10684                                  <1> ;
10685                                  <1> ;--------------------------------------------------------------------------
10686                                  <1> ; STATE OPERATION STARTED - ABSOLUTE ADDRESS 40:92 (DRIVE A) & 93 (DRIVE B)
10687                                  <1> ;--------------------------------------------------------------------------
10688                                  <1> ; PRESENT CYLINDER NUMBER - ABSOLUTE ADDRESS 40:94 (DRIVE A) & 95 (DRIVE B)
10689                                  <1> ;--------------------------------------------------------------------------
10690                                  <1>
10691                                  <1> struc MD
10692 00000000 <res 00000001>         <1>       .SPEC1       resb  1     ; SRT=D, HD UNLOAD=0F - 1ST SPECIFY BYTE
10693 00000001 <res 00000001>         <1>       .SPEC2       resb  1     ; HD LOAD=1, MODE=DMA - 2ND SPECIFY BYTE
10694 00000002 <res 00000001>         <1>       .OFF_TIM     resb  1     ; WAIT TIME AFTER OPERATION TILL MOTOR OFF
10695 00000003 <res 00000001>         <1>       .BYT_SEC     resb  1     ; 512 BYTES/SECTOR
10696 00000004 <res 00000001>         <1>       .SEC_TRK     resb  1     ; EOT (LAST SECTOR ON TRACK)
10697 00000005 <res 00000001>         <1>       .GAP         resb  1     ; GAP LENGTH
10698 00000006 <res 00000001>         <1>       .DTL         resb  1     ; DTL
10699 00000007 <res 00000001>         <1>       .GAP3        resb  1     ; GAP LENGTH FOR FORMAT
10700 00000008 <res 00000001>         <1>       .FIL_BYT     resb  1     ; FILL BYTE FOR FORMAT
10701 00000009 <res 00000001>         <1>       .HD_TIM            resb  1     ; HEAD SETTLE TIME (MILLISECONDS)
10702 0000000A <res 00000001>         <1>       .STR_TIM     resb  1     ; MOTOR START TIME (1/8 SECONDS)
10703 0000000B <res 00000001>         <1>       .MAX_TRK     resb  1     ; MAX. TRACK NUMBER
10704 0000000C <res 00000001>         <1>       .RATE        resb  1     ; DATA TRANSFER RATE
10705                                  <1> endstruc
10706                                  <1>
10707                                  <1> BIT7OFF     EQU   7FH
10708                                  <1> BIT7ON      EQU   80H
10709                                  <1>
10710                                  <1> ;;int13h: ; 16/02/2015
```

```
10711                                  <1> ;; 16/02/2015 - 21/02/2015
10712                                  <1> int40h:
10713 00003373 9C                      <1>         pushfd
10714 00003374 0E                      <1>         push   cs
10715 00003375 E801000000              <1>         call   DISKETTE_IO_1
10716 0000337A C3                      <1>         retn
10717                                  <1>
10718                                  <1> DISKETTE_IO_1:
10719                                  <1>
10720 0000337B FB                      <1>         STI                         ; INTERRUPTS BACK ON
10721 0000337C 55                      <1>         PUSH   eBP                  ; USER REGISTER
10722 0000337D 57                      <1>         PUSH   eDI                  ; USER REGISTER
10723 0000337E 52                      <1>         PUSH   eDX                  ; HEAD #, DRIVE # OR USER REGISTER
10724 0000337F 53                      <1>         PUSH   eBX                  ; BUFFER OFFSET PARAMETER OR REGISTER
10725 00003380 51                      <1>         PUSH   eCX                  ; TRACK #-SECTOR # OR USER REGISTER
10726 00003381 89E5                    <1>         MOV    eBP,eSP                   ; BP     => PARAMETER LIST DEP. ON AH
10727                                  <1>                                    ; [BP]   = SECTOR #
10728                                  <1>                                    ; [BP+1] = TRACK #
10729                                  <1>                                    ; [BP+2] = BUFFER OFFSET
10730                                  <1>                                    ; FOR RETURN OF DRIVE PARAMETERS:
10731                                  <1>                                    ; CL/[BP] = BITS 7&6 HI BITS OF MAX CYL
10732                                  <1>                                    ;         BITS 0-5 MAX SECTORS/TRACK
10733                                  <1>                                    ; CH/[BP+1] = LOW 8 BITS OF MAX CYL.
10734                                  <1>                                    ; BL/[BP+2] = BITS 7-4 = 0
10735                                  <1>                                    ;           BITS 3-0 = VALID CMOS TYPE
10736                                  <1>                                    ; BH/[BP+3] = 0
10737                                  <1>                                    ; DL/[BP+4] = # DRIVES INSTALLED
10738                                  <1>                                    ; DH/[BP+5] = MAX HEAD #
10739                                  <1>                                    ; DI/[BP+6] = OFFSET TO DISK BASE
10740 00003383 06                      <1>         push   es ; 06/02/2015
10741 00003384 1E                      <1>         PUSH   DS                   ; BUFFER SEGMENT PARM OR USER REGISTER
10742 00003385 56                      <1>         PUSH   eSI                  ; USER REGISTERS
10743                                  <1>         ;CALL  DDS                  ; SEGMENT OF BIOS DATA AREA TO DS
10744                                  <1>         ;mov   cx, cs
10745                                  <1>         ;mov   ds, cx
10746 00003386 66B91000                <1>         mov    cx, KDATA
10747 0000338A 8ED9                    <1>           mov     ds, cx
10748 0000338C 8EC1                    <1>           mov     es, cx
10749                                  <1>
10750                                  <1>         ;CMP   AH,(FNC_TAE-FNC_TAB)/2    ; CHECK FOR > LARGEST FUNCTION
10751 0000338E 80FC19                  <1>         cmp    ah,(FNC_TAE-FNC_TAB)/4  ; 18/02/2015
10752 00003391 7202                    <1>         JB     short OK_FUNC        ; FUNCTION OK
10753 00003393 B414                    <1>         MOV    AH,14H               ; REPLACE WITH KNOWN INVALID FUNCTION
10754                                  <1> OK_FUNC:
10755 00003395 80FC01                  <1>         CMP    AH,1                 ; RESET OR STATUS ?
10756 00003398 760C                    <1>         JBE    short OK_DRV         ; IF RESET OR STATUS DRIVE ALWAYS OK
10757 0000339A 80FC08                  <1>         CMP    AH,8                 ; READ DRIVE PARMS ?
10758 0000339D 7407                    <1>         JZ     short OK_DRV         ; IF SO DRIVE CHECKED LATER
10759 0000339F 80FA01                  <1>         CMP    DL,1                 ; DRIVES 0 AND 1 OK
10760 000033A2 7602                    <1>         JBE    short OK_DRV         ; IF 0 OR 1 THEN JUMP
10761 000033A4 B414                    <1>         MOV    AH,14H               ; REPLACE WITH KNOWN INVALID FUNCTION
10762                                  <1> OK_DRV:
10763 000033A6 31C9                    <1>         xor    ecx, ecx
10764                                  <1>         ;mov   esi, ecx ; 08/02/2015
10765 000033A8 89CF                    <1>         mov    edi, ecx ; 08/02/2015
10766 000033AA 88E1                    <1>         MOV    CL,AH                ; CL = FUNCTION
10767                                  <1>         ;XOR   CH,CH                ; CX = FUNCTION
10768                                  <1>         ;SHL   CL, 1                ; FUNCTION TIMES 2
10769 000033AC C0E102                  <1>         SHL    CL, 2 ; 20/02/2015   ; FUNCTION TIMES 4 (for 32 bit offset)
10770 000033AF BB[E7330000]            <1>         MOV    eBX,FNC_TAB          ; LOAD START OF FUNCTION TABLE
10771 000033B4 01CB                    <1>         ADD    eBX,eCX                   ; ADD OFFSET INTO TABLE => ROUTINE
10772 000033B6 88F4                    <1>         MOV    AH,DH                ; AX = HEAD #,# OF SECTORS OR DASD TYPE
10773 000033B8 30F6                    <1>         XOR    DH,DH                ; DX = DRIVE #
10774 000033BA 6689C6                  <1>         MOV    SI,AX                ; SI = HEAD #,# OF SECTORS OR DASD TYPE
10775 000033BD 6689D7                  <1>         MOV     DI,DX                    ; DI = DRIVE #
10776                                  <1>         ;
10777                                  <1>         ; 11/12/2014
10778 000033C0 8815[E55C0000]          <1>           mov     [cfd], dl              ; current floppy drive (for 'GET_PARM')
10779                                  <1>         ;
10780 000033C6 8A25[A8520100]          <1>         MOV    AH, [DSKETTE_STATUS]       ; LOAD STATUS TO AH FOR STATUS FUNCTION
10781 000033CC C605[A8520100]00        <1>         MOV    byte [DSKETTE_STATUS],0    ; INITIALIZE FOR ALL OTHERS
10782                                  <1>
10783                                  <1> ;     THROUGHOUT THE DISKETTE BIOS, THE FOLLOWING INFORMATION IS CONTAINED IN
10784                                  <1> ;     THE FOLLOWING MEMORY LOCATIONS AND REGISTERS. NOT ALL DISKETTE BIOS
10785                                  <1> ;     FUNCTIONS REQUIRE ALL OF THESE PARAMETERS.
10786                                  <1> ;
10787                                  <1> ;           DI     : DRIVE #
10788                                  <1> ;           SI-HI  : HEAD #
10789                                  <1> ;           SI-LOW : # OF SECTORS OR DASD TYPE FOR FORMAT
10790                                  <1> ;           ES     : BUFFER SEGMENT
10791                                  <1> ;           [BP]   : SECTOR #
10792                                  <1> ;           [BP+1] : TRACK #
10793                                  <1> ;           [BP+2] : BUFFER OFFSET
10794                                  <1> ;
10795                                  <1> ;     ACROSS CALLS TO SUBROUTINES THE CARRY FLAG (CY=1), WHERE INDICATED IN
10796                                  <1> ;     SUBROUTINE PROLOGUES, REPRESENTS AN EXCEPTION RETURN (NORMALLY AN ERROR
10797                                  <1> ;     CONDITION). IN MOST CASES, WHEN CY = 1, @DSKETTE_STATUS CONTAINS THE
10798                                  <1> ;     SPECIFIC ERROR CODE.
10799                                  <1> ;
10800                                  <1>                                    ; (AH) = @DSKETTE_STATUS
10801 000033D3 FF13                    <1>         CALL   dWORD [eBX]         ; CALL THE REQUESTED FUNCTION
10802 000033D5 5E                      <1>         POP    eSI                 ; RESTORE ALL REGISTERS
10803 000033D6 1F                      <1>         POP    DS
10804 000033D7 07                      <1>         pop    es     ; 06/02/2015
10805 000033D8 59                      <1>         POP    eCX
10806 000033D9 5B                      <1>         POP    eBX
10807 000033DA 5A                      <1>         POP    eDX
10808 000033DB 5F                      <1>         POP    eDI
10809 000033DC 89E5                    <1>         MOV    eBP, eSP
10810 000033DE 50                      <1>         PUSH   eAX
10811 000033DF 9C                      <1>         PUSHFd
10812 000033E0 58                      <1>         POP    eAX
10813                                  <1>         ;MOV   [BP+6], AX
```

```
10814 000033E1 89450C              <1>       mov    [ebp+12], eax  ; 18/02/2015, flags
10815 000033E4 58                  <1>       POP    eAX
10816 000033E5 5D                  <1>       POP    eBP
10817 000033E6 CF                  <1>       IRETd
10818                              <1>
10819                              <1> ;--------------------------------------------------------------------------------
10820                              <1> ; DW --> dd (06/02/2015)
10821 000033E7 [4B340000]          <1> FNC_TAB      dd     DSK_RESET           ; AH = 00H; RESET
10822 000033EB [C4340000]          <1>       dd     DSK_STATUS          ; AH = 01H; STATUS
10823 000033EF [D5340000]          <1>       dd     DSK_READ            ; AH = 02H; READ
10824 000033F3 [E6340000]          <1>       dd     DSK_WRITE           ; AH = 03H; WRITE
10825 000033F7 [F7340000]          <1>       dd     DSK_VERF            ; AH = 04H; VERIFY
10826 000033FB [08350000]          <1>       dd     DSK_FORMAT          ; AH = 05H; FORMAT
10827 000033FF [8D350000]          <1>       dd     FNC_ERR                   ; AH = 06H; INVALID
10828 00003403 [8D350000]          <1>       dd     FNC_ERR                   ; AH = 07H; INVALID
10829 00003407 [9A350000]          <1>       dd     DSK_PARMS           ; AH = 08H; READ DRIVE PARAMETERS
10830 0000340B [8D350000]          <1>       dd     FNC_ERR                   ; AH = 09H; INVALID
10831 0000340F [8D350000]          <1>       dd     FNC_ERR                   ; AH = 0AH; INVALID
10832 00003413 [8D350000]          <1>       dd     FNC_ERR                   ; AH = 0BH; INVALID
10833 00003417 [8D350000]          <1>       dd     FNC_ERR                   ; AH = 0CH; INVALID
10834 0000341B [8D350000]          <1>       dd     FNC_ERR                   ; AH = 0DH; INVALID
10835 0000341F [8D350000]          <1>       dd     FNC_ERR                   ; AH = 0EH; INVALID
10836 00003423 [8D350000]          <1>       dd     FNC_ERR                   ; AH = 0FH; INVALID
10837 00003427 [8D350000]          <1>       dd     FNC_ERR                   ; AH = 10H; INVALID
10838 0000342B [8D350000]          <1>       dd     FNC_ERR                   ; AH = 11H; INVALID
10839 0000342F [8D350000]          <1>       dd     FNC_ERR                   ; AH = 12H; INVALID
10840 00003433 [8D350000]          <1>       dd     FNC_ERR                   ; AH = 13H; INVALID
10841 00003437 [8D350000]          <1>       dd     FNC_ERR                   ; AH = 14H; INVALID
10842 0000343B [72360000]          <1>       dd     DSK_TYPE            ; AH = 15H; READ DASD TYPE
10843 0000343F [9D360000]          <1>       dd     DSK_CHANGE          ; AH = 16H; CHANGE STATUS
10844 00003443 [D7360000]          <1>       dd     FORMAT_SET          ; AH = 17H; SET DASD TYPE
10845 00003447 [5A370000]          <1>       dd     SET_MEDIA           ; AH = 18H; SET MEDIA TYPE
10846                              <1> FNC_TAE EQU    $                   ; END
10847                              <1>
10848                              <1> ;--------------------------------------------------------------------------------
10849                              <1> ; DISK_RESET(AH = 00H)
10850                              <1> ;           RESET THE DISKETTE SYSTEM.
10851                              <1> ;
10852                              <1> ; ON EXIT:  @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION
10853                              <1> ;--------------------------------------------------------------------------------
10854                              <1> DSK_RESET:
10855 0000344B 66BAF203            <1>       MOV    DX,03F2H            ; ADAPTER CONTROL PORT
10856 0000344F FA                  <1>       CLI                        ; NO INTERRUPTS
10857 00003450 A0[A6520100]        <1>       MOV    AL,[MOTOR_STATUS]   ; GET DIGITAL OUTPUT REGISTER REFLECTION
10858 00003455 243F                <1>       AND    AL,00111111B        ; KEEP SELECTED AND MOTOR ON BITS
10859 00003457 C0C004              <1>       ROL    AL,4                ; MOTOR VALUE TO HIGH NIBBLE
10860                              <1>                                  ; DRIVE SELECT TO LOW NIBBLE
10861 0000345A 0C08                <1>       OR     AL,00001000B        ; TURN ON INTERRUPT ENABLE
10862 0000345C EE                  <1>       OUT    DX,AL               ; RESET THE ADAPTER
10863 0000345D C605[A5520100]00    <1>       MOV    byte [SEEK_STATUS],0      ; SET RECALIBRATE REQUIRED ON ALL DRIVES
10864                              <1>       ;JMP   $+2                 ; WAIT FOR I/O
10865                              <1>       ;JMP   $+2                 ; WAIT FOR I/O (TO INSURE MINIMUM
10866                              <1>                                  ;      PULSE WIDTH)
10867                              <1>       ; 19/12/2014
10868                              <1>       NEWIODELAY
10869 00003464 E6EB                <2>  out 0ebh,al
10870                              <1>
10871                              <1>       ; 17/12/2014
10872                              <1>       ; AWARD BIOS 1999 - RESETDRIVES (ADISK.ASM)
10873 00003466 B915000000          <1>       mov    ecx, WAITCPU_RESET_ON     ; cx = 21 -- Min. 14 micro seconds !?
10874                              <1> wdw1:
10875                              <1>       NEWIODELAY   ; 27/02/2015
10876 0000346B E6EB                <2>  out 0ebh,al
10877 0000346D E2FC                <1>       loop   wdw1
10878                              <1>       ;
10879 0000346F 0C04                <1>       OR     AL,00000100B        ; TURN OFF RESET BIT
10880 00003471 EE                  <1>       OUT    DX,AL               ; RESET THE ADAPTER
10881                              <1>       ; 16/12/2014
10882                              <1>       IODELAY
10883 00003472 EB00                <2>  jmp short $+2
10884 00003474 EB00                <2>  jmp short $+2
10885                              <1>       ;
10886                              <1>       ;STI                        ; ENABLE THE INTERRUPTS
10887 00003476 E83C0C0000          <1>       CALL   WAIT_INT            ; WAIT FOR THE INTERRUPT
10888 0000347B 723E                <1>       JC     short DR_ERR        ; IF ERROR, RETURN IT
10889 0000347D 66B9C000            <1>       MOV    CX,11000000B        ; CL = EXPECTED @NEC_STATUS
10890                              <1> NXT_DRV:
10891 00003481 6651                <1>       PUSH   CX                  ; SAVE FOR CALL
10892 00003483 B8[B9340000]        <1>       MOV    eAX, DR_POP_ERR     ; LOAD NEC_OUTPUT ERROR ADDRESS
10893 00003488 50                  <1>       PUSH   eAX                 ; "
10894 00003489 B408                <1>       MOV    AH,08H              ; SENSE INTERRUPT STATUS COMMAND
10895 0000348B E81A0B0000          <1>       CALL   NEC_OUTPUT
10896 00003490 58                  <1>       POP    eAX                 ; THROW AWAY ERROR RETURN
10897 00003491 E8510C0000          <1>       CALL   RESULTS                   ; READ IN THE RESULTS
10898 00003496 6659                <1>       POP    CX                  ; RESTORE AFTER CALL
10899 00003498 7221                <1>       JC     short DR_ERR        ; ERROR RETURN
10900 0000349A 3A0D[A9520100]      <1>       CMP    CL, [NEC_STATUS]    ; TEST FOR DRIVE READY TRANSITION
10901 000034A0 7519                <1>       JNZ    short DR_ERR        ; EVERYTHING OK
10902 000034A2 FEC1                <1>       INC    CL                  ; NEXT EXPECTED @NEC_STATUS
10903 000034A4 80F9C3              <1>       CMP    CL,11000011B        ; ALL POSSIBLE DRIVES CLEARED
10904 000034A7 76D8                <1>       JBE    short NXT_DRV       ; FALL THRU IF 11000100B OR >
10905                              <1>       ;
10906 000034A9 E869030000          <1>       CALL   SEND_SPEC           ; SEND SPECIFY COMMAND TO NEC
10907                              <1> RESBAC:
10908 000034AE E81D090000          <1>       CALL   SETUP_END           ; VARIOUS CLEANUPS
10909 000034B3 6689F3              <1>       MOV    BX,SI               ; GET SAVED AL TO BL
10910 000034B6 88D8                <1>       MOV    AL,BL               ; PUT BACK FOR RETURN
10911 000034B8 C3                  <1>       RETn
10912                              <1> DR_POP_ERR:
10913 000034B9 6659                <1>       POP    CX                  ; CLEAR STACK
10914                              <1> DR_ERR:
10915 000034BB 800D[A8520100]20    <1>       OR     byte [DSKETTE_STATUS],BAD_NEC ; SET ERROR CODE
10916 000034C2 EBEA                <1>       JMP    SHORT RESBAC        ; RETURN FROM RESET
```

```
10917                                  <1>
10918                                  <1> ;--------------------------------------------------------------------------
10919                                  <1> ; DISK_STATUS      (AH = 01H)
10920                                  <1> ;     DISKETTE STATUS.
10921                                  <1> ;
10922                                  <1> ; ON ENTRY:  AH : STATUS OF PREVIOUS OPERATION
10923                                  <1> ;
10924                                  <1> ; ON EXIT:   AH, @DSKETTE_STATUS, CY REFLECT STATUS OF PREVIOUS OPERATION.
10925                                  <1> ;--------------------------------------------------------------------------
10926                                  <1> DSK_STATUS:
10927 000034C4 8825[A8520100]          <1>       MOV    [DSKETTE_STATUS],AH ; PUT BACK FOR SETUP END
10928 000034CA E801090000              <1>       CALL   SETUP_END           ; VARIOUS CLEANUPS
10929 000034CF 6689F3                  <1>       MOV    BX,SI               ; GET SAVED AL TO BL
10930 000034D2 88D8                    <1>       MOV    AL,BL               ; PUT BACK FOR RETURN
10931 000034D4 C3                      <1>       RETn
10932                                  <1>
10933                                  <1> ;--------------------------------------------------------------------------
10934                                  <1> ; DISK_READ  (AH = 02H)
10935                                  <1> ;     DISKETTE READ.
10936                                  <1> ;
10937                                  <1> ; ON ENTRY:  DI     : DRIVE #
10938                                  <1> ;            SI-HI  : HEAD #
10939                                  <1> ;            SI-LOW : # OF SECTORS
10940                                  <1> ;            ES     : BUFFER SEGMENT
10941                                  <1> ;            [BP]   : SECTOR #
10942                                  <1> ;            [BP+1] : TRACK #
10943                                  <1> ;            [BP+2] : BUFFER OFFSET
10944                                  <1> ;
10945                                  <1> ; ON EXIT:   @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION
10946                                  <1> ;--------------------------------------------------------------------------
10947                                  <1>
10948                                  <1> ; 06/02/2015, ES:BX -> EBX (unix386.s)
10949                                  <1>
10950                                  <1> DSK_READ:
10951 000034D5 8025[A6520100]7F        <1>       AND    byte [MOTOR_STATUS],01111111B ; INDICATE A READ OPERATION
10952 000034DC 66B846E6                <1>       MOV    AX,0E646H           ; AX = NEC COMMAND, DMA COMMAND
10953 000034E0 E83C040000              <1>       CALL   RD_WR_VF            ; COMMON READ/WRITE/VERIFY
10954 000034E5 C3                      <1>       RETn
10955                                  <1>
10956                                  <1> ;--------------------------------------------------------------------------
10957                                  <1> ; DISK_WRITE (AH = 03H)
10958                                  <1> ;     DISKETTE WRITE.
10959                                  <1> ;
10960                                  <1> ; ON ENTRY:  DI     : DRIVE #
10961                                  <1> ;            SI-HI  : HEAD #
10962                                  <1> ;            SI-LOW : # OF SECTORS
10963                                  <1> ;            ES     : BUFFER SEGMENT
10964                                  <1> ;            [BP]   : SECTOR #
10965                                  <1> ;            [BP+1] : TRACK #
10966                                  <1> ;            [BP+2] : BUFFER OFFSET
10967                                  <1> ;
10968                                  <1> ; ON EXIT:   @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION
10969                                  <1> ;--------------------------------------------------------------------------
10970                                  <1>
10971                                  <1> ; 06/02/2015, ES:BX -> EBX (unix386.s)
10972                                  <1>
10973                                  <1> DSK_WRITE:
10974 000034E6 66B84AC5                <1>       MOV    AX,0C54AH           ; AX = NEC COMMAND, DMA COMMAND
10975 000034EA 800D[A6520100]80        <1>         OR     byte [MOTOR_STATUS],10000000B ; INDICATE WRITE OPERATION
10976 000034F1 E82B040000              <1>       CALL   RD_WR_VF            ; COMMON READ/WRITE/VERIFY
10977 000034F6 C3                      <1>       RETn
10978                                  <1>
10979                                  <1> ;--------------------------------------------------------------------------
10980                                  <1> ; DISK_VERF  (AH = 04H)
10981                                  <1> ;     DISKETTE VERIFY.
10982                                  <1> ;
10983                                  <1> ; ON ENTRY:  DI     : DRIVE #
10984                                  <1> ;            SI-HI  : HEAD #
10985                                  <1> ;            SI-LOW : # OF SECTORS
10986                                  <1> ;            ES     : BUFFER SEGMENT
10987                                  <1> ;            [BP]   : SECTOR #
10988                                  <1> ;            [BP+1] : TRACK #
10989                                  <1> ;            [BP+2] : BUFFER OFFSET
10990                                  <1> ;
10991                                  <1> ; ON EXIT:   @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION
10992                                  <1> ;--------------------------------------------------------------------------
10993                                  <1> DSK_VERF:
10994 000034F7 8025[A6520100]7F        <1>       AND    byte [MOTOR_STATUS],01111111B ; INDICATE A READ OPERATION
10995 000034FE 66B842E6                <1>       MOV    AX,0E642H           ; AX = NEC COMMAND, DMA COMMAND
10996 00003502 E81A040000              <1>       CALL   RD_WR_VF            ; COMMON READ/WRITE/VERIFY
10997 00003507 C3                      <1>       RETn
10998                                  <1>
10999                                  <1> ;--------------------------------------------------------------------------
11000                                  <1> ; DISK_FORMAT      (AH = 05H)
11001                                  <1> ;     DISKETTE FORMAT.
11002                                  <1> ;
11003                                  <1> ; ON ENTRY:  DI     : DRIVE #
11004                                  <1> ;            SI-HI  : HEAD #
11005                                  <1> ;            SI-LOW : # OF SECTORS
11006                                  <1> ;            ES     : BUFFER SEGMENT
11007                                  <1> ;            [BP]   : SECTOR #
11008                                  <1> ;            [BP+1] : TRACK #
11009                                  <1> ;            [BP+2] : BUFFER OFFSET
11010                                  <1> ;            @DISK_POINTER POINTS TO THE PARAMETER TABLE OF THIS DRIVE
11011                                  <1> ;
11012                                  <1> ; ON EXIT:   @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION
11013                                  <1> ;--------------------------------------------------------------------------
11014                                  <1> DSK_FORMAT:
11015 00003508 E853030000              <1>       CALL   XLAT_NEW            ; TRANSLATE STATE TO PRESENT ARCH.
11016 0000350D E84F050000              <1>       CALL   FMT_INIT            ; ESTABLISH STATE IF UNESTABLISHED
11017 00003512 800D[A6520100]80        <1>         OR     byte [MOTOR_STATUS], 10000000B ; INDICATE WRITE OPERATION
11018 00003519 E897050000              <1>       CALL   MED_CHANGE          ; CHECK MEDIA CHANGE AND RESET IF SO
11019 0000351E 725D                    <1>         JC     short FM_DON            ; MEDIA CHANGED, SKIP
```

```
11020 00003520 E8F2020000          <1>       CALL   SEND_SPEC           ; SEND SPECIFY COMMAND TO NEC
11021 00003525 E8FD050000          <1>       CALL   CHK_LASTRATE        ; ZF=1 ATTEMPT RATE IS SAME AS LAST RATE
11022 0000352A 7405                <1>         JZ     short FM_WR             ; YES, SKIP SPECIFY COMMAND
11023 0000352C E8D4050000          <1>       CALL   SEND_RATE           ; SEND DATA RATE TO CONTROLLER
11024                              <1> FM_WR:
11025 00003531 E88A060000          <1>       CALL   FMTDMA_SET          ; SET UP THE DMA FOR FORMAT
11026 00003536 7245                <1>         JC     short FM_DON            ; RETURN WITH ERROR
11027 00003538 B44D                <1>       MOV    AH,04DH                ; ESTABLISH THE FORMAT COMMAND
11028 0000353A E8E7060000          <1>       CALL   NEC_INIT            ; INITIALIZE THE NEC
11029 0000353F 723C                <1>         JC     short FM_DON            ; ERROR - EXIT
11030 00003541 B8[7D350000]        <1>         MOV    eAX, FM_DON             ; LOAD ERROR ADDRESS
11031 00003546 50                  <1>       PUSH   eAX                 ; PUSH NEC_OUT ERROR RETURN
11032 00003547 B203                <1>       MOV    DL,3                ; BYTES/SECTOR VALUE TO NEC
11033 00003549 E856090000          <1>       CALL   GET_PARM
11034 0000354E E8570A0000          <1>       CALL   NEC_OUTPUT
11035 00003553 B204                <1>       MOV    DL,4                ; SECTORS/TRACK VALUE TO NEC
11036 00003555 E84A090000          <1>       CALL   GET_PARM
11037 0000355A E84B0A0000          <1>       CALL   NEC_OUTPUT
11038 0000355F B207                <1>       MOV    DL,7                ; GAP LENGTH VALUE TO NEC
11039 00003561 E83E090000          <1>       CALL   GET_PARM
11040 00003566 E83F0A0000          <1>       CALL   NEC_OUTPUT
11041 0000356B B208                <1>       MOV    DL,8                ; FILLER BYTE TO NEC
11042 0000356D E832090000          <1>       CALL   GET_PARM
11043 00003572 E8330A0000          <1>       CALL   NEC_OUTPUT
11044 00003577 58                  <1>       POP    eAX                 ; THROW AWAY ERROR
11045 00003578 E827070000          <1>       CALL   NEC_TERM            ; TERMINATE, RECEIVE STATUS, ETC,
11046                              <1> FM_DON:
11047 0000357D E80F030000          <1>       CALL   XLAT_OLD            ; TRANSLATE STATE TO COMPATIBLE MODE
11048 00003582 E849080000          <1>       CALL   SETUP_END           ; VARIOUS CLEANUPS
11049 00003587 6689F3              <1>       MOV    BX,SI               ; GET SAVED AL TO BL
11050 0000358A 88D8                <1>       MOV    AL,BL               ; PUT BACK FOR RETURN
11051 0000358C C3                  <1>       RETn
11052                              <1>
11053                              <1> ;--------------------------------------------------------------------------------
11054                              <1> ; FNC_ERR
11055                              <1> ;     INVALID FUNCTION REQUESTED OR INVALID DRIVE:
11056                              <1> ;     SET BAD COMMAND IN STATUS.
11057                              <1> ;
11058                              <1> ; ON EXIT:  @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION
11059                              <1> ;--------------------------------------------------------------------------------
11060                              <1> FNC_ERR:                           ; INVALID FUNCTION REQUEST
11061 0000358D 6689F0              <1>       MOV    AX,SI               ; RESTORE AL
11062 00003590 B401                <1>       MOV    AH,BAD_CMD          ; SET BAD COMMAND ERROR
11063 00003592 8825[A8520100]      <1>       MOV    [DSKETTE_STATUS],AH ; STORE IN DATA AREA
11064 00003598 F9                  <1>       STC                        ; SET CARRY INDICATING ERROR
11065 00003599 C3                  <1>       RETn
11066                              <1>
11067                              <1> ; 01/06/2016
11068                              <1> ; 28/05/2016
11069                              <1> ; 27/05/2016 - TRDOS 386 (TRDOS v.2.0)
11070                              <1> ;--------------------------------------------------------------------------------
11071                              <1> ; DISK_PARMS(AH = 08H)
11072                              <1> ;     READ DRIVE PARAMETERS.
11073                              <1> ;
11074                              <1> ; ON ENTRY:  DI : DRIVE #
11075                              <1> ;            ; 27/05/2016
11076                              <1> ;            EBX = Buffer Address for floppy disk parameters table (16 bytes)
11077                              <1> ;
11078                              <1> ; ON EXIT:   CL/[BP]   = BITS 7 & 6 HI 2 BITS OF MAX CYLINDER
11079                              <1> ;                        BITS 0-5 MAX SECTORS/TRACK
11080                              <1> ;            CH/[BP+1] = LOW 8 BITS OF MAX CYLINDER
11081                              <1> ;            BL/[BP+2] = BITS 7-4 = 0
11082                              <1> ;                        BITS 3-0 = VALID CMOS DRIVE TYPE
11083                              <1> ;            BH/[BP+3] = 0
11084                              <1> ;            DL/[BP+4] = # DRIVES INSTALLED (VALUE CHECKED)
11085                              <1> ;            DH/[BP+5] = MAX HEAD #
11086                              <1> ;          ** 27/05/2016 - TRDOS 386 (TRDOS v2.0) **
11087                              <1> ;            ** EBX = Buffer address for floppy disk parameters table **
11088                              <1> ;            ;DI/[BP+6] = OFFSET TO DISK_BASE
11089                              <1> ;            ;ES        = SEGMENT OF DISK_BASE
11090                              <1> ;
11091                              <1> ;            AX        = 0
11092                              <1> ;
11093                              <1> ;            NOTE : THE ABOVE INFORMATION IS STORED IN THE USERS STACK AT
11094                              <1> ;                   THE LOCATIONS WHERE THE MAIN ROUTINE WILL POP THEM
11095                              <1> ;                   INTO THE APPROPRIATE REGISTERS BEFORE RETURNING TO THE
11096                              <1> ;                   CALLER.
11097                              <1> ;--------------------------------------------------------------------------------
11098                              <1> DSK_PARMS:
11099 0000359A E8C1020000          <1>        CALL   XLAT_NEW            ; TRANSLATE STATE TO PRESENT ARCH,
11100                              <1>      ;        MOV    WORD [BP+2],0       ; DRIVE TYPE = 0
11101                              <1>      ;  MOV     AX, [EQUIP_FLAG]         ; LOAD EQUIPMENT FLAG FOR # DISKETTES
11102                              <1>      ;  AND     AL,11000001B             ; KEEP DISKETTE DRIVE BITS
11103                              <1>      ;  MOV     DL,2                     ; DISKETTE DRIVES = 2
11104                              <1>      ;  CMP     AL,01000001B             ; 2 DRIVES INSTALLED ?
11105                              <1>      ;  JZ      short STO_DL             ; IF YES JUMP
11106                              <1>      ;  DEC     DL                       ; DISKETTE DRIVES = 1
11107                              <1>      ;  CMP     AL,00000001B             ; 1 DRIVE INSTALLED ?
11108                              <1>      ;  JNZ     short NON_DRV            ; IF NO JUMP
11109 0000359F 29D2                <1>        sub    edx, edx
11110 000035A1 66A1[F65C0000]      <1>        mov     ax, [fd0_type]
11111 000035A7 6621C0              <1>        and     ax, ax
11112 000035AA 0F848A000000        <1>          jz      NON_DRV
11113 000035B0 FEC2                <1>        inc     dl
11114 000035B2 20E4                <1>        and     ah, ah
11115 000035B4 7402                <1>        jz      short STO_DL
11116 000035B6 FEC2                <1>        inc     dl
11117                              <1> STO_DL:
11118                              <1>        ;MOV   [BP+4],DL          ; STORE NUMBER OF DRIVES
11119 000035B8 895508              <1>        mov    [ebp+8], edx ; 20/02/2015
11120 000035BB 6683FF01            <1>        CMP    DI,1                ; CHECK FOR VALID DRIVE
11121 000035BF 777C                <1>        JA     short NON_DRV1            ; DRIVE INVALID
11122                              <1>        ;MOV   BYTE [BP+5],1       ; MAXIMUM HEAD NUMBER =   1
```

```
11123 000035C1 C6450901          <1>       mov    byte [ebp+9], 1  ; 20/02/2015
11124 000035C5 E8D1080000        <1>       CALL   CMOS_TYPE          ; RETURN DRIVE TYPE IN AL
11125                            <1>       ;;20/02/2015
11126                            <1>       ;;JC   short CHK_EST      ; IF CMOS BAD CHECKSUM ESTABLISHED
11127                            <1>       ;;OR   AL,AL              ; TEST FOR NO DRIVE TYPE
11128 000035CA 740F              <1>       JZ     short CHK_EST      ; JUMP IF SO
11129 000035CC E81B020000        <1>       CALL   DR_TYPE_CHECK      ; RTN CS:BX = MEDIA/DRIVE PARAM TBL
11130 000035D1 7208              <1>       JC     short CHK_EST      ; TYPE NOT IN TABLE (POSSIBLE BAD CMOS)
11131                            <1>       ;MOV   [BP+2],AL          ; STORE VALID CMOS DRIVE TYPE
11132                            <1>         ;mov [ebp+4], al ; 06/02/2015
11133 000035D3 8A4B04            <1>       MOV     CL, [eBX+MD.SEC_TRK]     ; GET SECTOR/TRACK
11134 000035D6 8A6B0B            <1>         MOV     CH, [eBX+MD.MAX_TRK]     ; GET MAX. TRACK NUMBER
11135 000035D9 EB36              <1>       JMP    SHORT STO_CX       ; CMOS GOOD, USE CMOS
11136                            <1> CHK_EST:
11137 000035DB 8AA7[B5520100]    <1>       MOV    AH, [DSK_STATE+eDI] ; LOAD STATE FOR THIS DRIVE
11138 000035E1 F6C410            <1>       TEST   AH,MED_DET         ; CHECK FOR ESTABLISHED STATE
11139 000035E4 7457              <1>       JZ     short NON_DRV1           ; CMOS BAD/INVALID OR UNESTABLISHED
11140                            <1> USE_EST:
11141 000035E6 80E4C0            <1>       AND    AH,RATE_MSK        ; ISOLATE STATE
11142 000035E9 80FC80            <1>       CMP    AH,RATE_250        ; RATE 250 ?
11143 000035EC 7570              <1>       JNE    short USE_EST2           ; NO, GO CHECK OTHER RATE
11144                            <1>
11145                            <1> ;-----       DATA RATE IS 250 KBS, TRY 360 KB TABLE FIRST
11146                            <1>
11147 000035EE B001              <1>       MOV    AL,01              ; DRIVE TYPE 1 (360KB)
11148 000035F0 E8F7010000        <1>       CALL   DR_TYPE_CHECK      ; RTN CS:BX = MEDIA/DRIVE PARAM TBL
11149 000035F5 8A4B04            <1>         MOV     CL, [eBX+MD.SEC_TRK]    ; GET SECTOR/TRACK
11150 000035F8 8A6B0B            <1>         MOV     CH, [eBX+MD.MAX_TRK]    ; GET MAX. TRACK NUMBER
11151 000035FB F687[B5520100]01  <1>       TEST   byte [DSK_STATE+eDI],TRK_CAPA ; 80 TRACK ?
11152 00003602 740D              <1>       JZ     short STO_CX       ; MUST BE 360KB DRIVE
11153                            <1>
11154                            <1> ;-----       IT IS 1.44 MB DRIVE
11155                            <1>
11156                            <1> PARM144:
11157 00003604 B004              <1>       MOV    AL,04              ; DRIVE TYPE 4 (1.44MB)
11158 00003606 E8E1010000        <1>       CALL   DR_TYPE_CHECK      ; RTN CS:BX = MEDIA/DRIVE PARAM TBL
11159 0000360B 8A4B04            <1>         MOV     CL, [eBX+MD.SEC_TRK]    ; GET SECTOR/TRACK
11160 0000360E 8A6B0B            <1>         MOV     CH, [eBX+MD.MAX_TRK]    ; GET MAX. TRACK NUMBER
11161                            <1> STO_CX:
11162 00003611 894D00            <1>       MOV    [eBP],eCX          ; SAVE POINTER IN STACK FOR RETURN
11163                            <1> ES_DI:
11164                            <1>       ;MOV   [BP+6],BX          ; ADDRESS OF MEDIA/DRIVE PARM TABLE
11165                            <1>       ;mov   [ebp+12], ebx ; 06/02/2015
11166                            <1>       ;MOV   AX,CS              ; SEGMENT MEDIA/DRIVE PARAMETER TABLE
11167                            <1>       ;MOV   ES,AX              ; ES IS SEGMENT OF TABLE
11168                            <1>       ;
11169                            <1>       ; 28/05/2016
11170                            <1>       ; 27/05/2016
11171                            <1>       ; return floppy disk parameters table to user
11172                            <1>       ; in user's buffer, which is pointed by EBX
11173                            <1>       ;
11174 00003614 57                <1>       push   edi
11175 00003615 8B7D04            <1>       mov    edi, [ebp+4]              ; ebx (input), user's buffer address
11176 00003618 0FB6C0            <1>       movzx  eax, al
11177 0000361B 894504            <1>         mov [ebp+4], eax   ; ebx      ; drive type (for floppy drives)
11178                            <1>       ; 01/06/2016 (INT 33h, disk type return for floppy disks, in BL)
11179 0000361E A3[B05F0100]      <1>       mov    [user_buffer], eax  ; 01/06/2016 (overwrite ebx return value)
11180                            <1>       ;(INT 33h, Function 08h will replace user's buffer addr with disk type!)
11181                            <1>       ;
11182 00003623 89DE              <1>       mov    esi, ebx           ; floppy disk parameter table (16 bytes)
11183 00003625 B910000000        <1>       mov    ecx, 16 ; 16 bytes
11184 0000362A E87EB20000        <1>         call    transfer_to_user_buffer ; trdosk6.s (16/05/2016)
11185 0000362F 5F                <1>       pop    edi
11186                            <1> DP_OUT:
11187 00003630 E85C020000        <1>       CALL   XLAT_OLD           ; TRANSLATE STATE TO COMPATIBLE MODE
11188 00003635 6631C0            <1>       XOR    AX,AX              ; CLEAR
11189 00003638 F8                <1>       CLC
11190 00003639 C3                <1>       RETn
11191                            <1>
11192                            <1> ;-----       NO DRIYE PRESENT HANDLER
11193                            <1>
11194                            <1> NON_DRV:
11195                            <1>       ;MOV   BYTE [BP+4],0       ; CLEAR NUMBER OF DRIVES
11196 0000363A 895508            <1>       mov    [ebp+8], edx ; 0 ; 20/02/2015
11197                            <1> NON_DRV1:
11198 0000363D 6681FF8000        <1>       CMP    DI,80H             ; CHECK FOR FIXED MEDIA TYPE REQUEST
11199 00003642 720C              <1>       JB     short NON_DRV2           ; CONTINUE IF NOT REQUEST FALL THROUGH
11200                            <1>
11201                            <1> ;-----       FIXED DISK REQUEST FALL THROUGH ERROR
11202                            <1>
11203 00003644 E848020000        <1>       CALL   XLAT_OLD           ; ELSE TRANSLATE TO COMPATIBLE MODE
11204 00003649 6689F0            <1>       MOV    AX,SI              ; RESTORE AL
11205 0000364C B401              <1>       MOV    AH,BAD_CMD         ; SET BAD COMMAND ERROR
11206 0000364E F9                <1>       STC
11207 0000364F C3                <1>       RETn
11208                            <1>
11209                            <1> NON_DRV2:
11210                            <1>       ;XOR   AX,AX              ; CLEAR PARMS IF NO DRIVES OR CMOS BAD
11211 00003650 31C0              <1>       xor    eax, eax
11212 00003652 66894500          <1>       MOV    [eBP],AX           ; TRACKS, SECTORS/TRACK = 0
11213                            <1>       ;MOV   [BP+5],AH          ; HEAD = 0
11214 00003656 886509            <1>       mov    [ebp+9], ah ; 06/02/2015
11215                            <1>       ;MOV   [BP+6],AX          ; OFFSET TO DISK_BASE = 0
11216 00003659 89450C            <1>       mov    [ebp+12], eax
11217                            <1>       ;MOV   ES,AX              ; ES IS SEGMENT OF TABLE
11218 0000365C EBD2              <1>       JMP    SHORT DP_OUT
11219                            <1>
11220                            <1> ;-----       DATA RATE IS EITHER 300 KBS OR 500 KBS, TRY 1.2 MB TABLE FIRST
11221                            <1>
11222                            <1> USE_EST2:
11223 0000365E B002              <1>       MOV    AL,02              ; DRIVE TYPE 2 (1.2MB)
11224 00003660 E887010000        <1>       CALL   DR_TYPE_CHECK      ; RTN CS:BX = MEDIA/DRIVE PARAM TBL
11225 00003665 8A4B04            <1>         MOV     CL, [eBX+MD.SEC_TRK]    ; GET SECTOR/TRACK
```

```
11226 00003668 8A6B0B              <1>         MOV     CH, [eBX+MD.MAX_TRK]    ; GET MAX. TRACK NUMBER
11227 0000366B 80FC40              <1>       CMP    AH,RATE_300         ; RATE 300 ?
11228 0000366E 74A1                <1>       JZ     short STO_CX        ; MUST BE 1.2MB DRIVE
11229 00003670 EB92                <1>       JMP    SHORT PARM144       ; ELSE, IT IS 1.44MB DRIVE
11230                              <1>
11231                              <1> ;--------------------------------------------------------------------------
11232                              <1> ; DISK_TYPE (AH = 15H)
11233                              <1> ;     THIS ROUTINE RETURNS THE TYPE OF MEDIA INSTALLED.
11234                              <1> ;
11235                              <1> ;  ON ENTRY: DI = DRIVE #
11236                              <1> ;
11237                              <1> ;  ON EXIT:  AH = DRIVE TYPE, CY=0
11238                              <1> ;--------------------------------------------------------------------------
11239                              <1> DSK_TYPE:
11240 00003672 E8E9010000          <1>       CALL   XLAT_NEW            ; TRANSLATE STATE TO PRESENT ARCH.
11241 00003677 8A87[B5520100]      <1>       MOV    AL, [DSK_STATE+eDI] ; GET PRESENT STATE INFORMATION
11242 0000367D 08C0                <1>       OR     AL,AL               ; CHECK FOR NO DRIVE
11243 0000367F 7418                <1>       JZ     short NO_DRV
11244 00003681 B401                <1>       MOV    AH,NOCHGLN          ; NO CHANGE LINE FOR 40 TRACK DRIVE
11245 00003683 A801                <1>       TEST   AL,TRK_CAPA         ; IS THIS DRIVE AN 80 TRACK DRIVE?
11246 00003685 7402                <1>       JZ     short DT_BACK             ; IF NO JUMP
11247 00003687 B402                <1>       MOV    AH,CHGLN            ; CHANGE LINE FOR 80 TRACK DRIVE
11248                              <1> DT_BACK:
11249 00003689 6650                <1>       PUSH   AX                  ; SAVE RETURN VALUE
11250 0000368B E801020000          <1>       CALL   XLAT_OLD            ; TRANSLATE STATE TO COMPATIBLE MODE
11251 00003690 6658                <1>       POP    AX                  ; RESTORE RETURN VALUE
11252 00003692 F8                  <1>       CLC                        ; NO ERROR
11253 00003693 6689F3              <1>       MOV    BX,SI               ; GET SAVED AL TO BL
11254 00003696 88D8                <1>       MOV    AL,BL               ; PUT BACK FOR RETURN
11255 00003698 C3                  <1>       RETn
11256                              <1> NO_DRV:
11257 00003699 30E4                <1>       XOR    AH,AH               ; NO DRIVE PRESENT OR UNKNOWN
11258 0000369B EBEC                <1>       JMP    SHORT DT_BACK
11259                              <1>
11260                              <1> ;--------------------------------------------------------------------------
11261                              <1> ; DISK_CHANGE      (AH = 16H)
11262                              <1> ;     THIS ROUTINE RETURNS THE STATE OF THE DISK CHANGE LINE.
11263                              <1> ;
11264                              <1> ; ON ENTRY:  DI = DRIVE #
11265                              <1> ;
11266                              <1> ; ON EXIT:   AH = @DSKETTE_STATUS
11267                              <1> ;                 00 - DISK CHANGE LINE INACTIVE, CY = 0
11268                              <1> ;                 06 - DISK CHANGE LINE ACTIVE, CY = 1
11269                              <1> ;--------------------------------------------------------------------------
11270                              <1> DSK_CHANGE:
11271 0000369D E8BE010000          <1>       CALL   XLAT_NEW            ; TRANSLATE STATE TO PRESENT ARCH.
11272 000036A2 8A87[B5520100]      <1>       MOV    AL, [DSK_STATE+eDI] ; GET MEDIA STATE INFORMATION
11273 000036A8 08C0                <1>       OR     AL,AL               ; DRIVE PRESENT ?
11274 000036AA 7422                <1>       JZ     short DC_NON        ; JUMP IF NO DRIVE
11275 000036AC A801                <1>       TEST   AL,TRK_CAPA         ; 80 TRACK DRIVE ?
11276 000036AE 7407                <1>       JZ     short SETIT         ; IF SO , CHECK CHANGE LINE
11277                              <1> DC0:
11278 000036B0 E88D0A0000          <1>         CALL    READ_DSKCHNG            ; GO CHECK STATE OF DISK CHANGE LINE
11279 000036B5 7407                <1>       JZ     short FINIS         ; CHANGE LINE NOT ACTIVE
11280                              <1>
11281 000036B7 C605[A8520100]06    <1> SETIT:      MOV    byte [DSKETTE_STATUS], MEDIA_CHANGE ; INDICATE MEDIA REMOVED
11282                              <1>
11283 000036BE E8CE010000          <1> FINIS:      CALL   XLAT_OLD           ; TRANSLATE STATE TO COMPATIBLE MODE
11284 000036C3 E808070000          <1>       CALL   SETUP_END           ; VARIOUS CLEANUPS
11285 000036C8 6689F3              <1>       MOV    BX,SI               ; GET SAVED AL TO BL
11286 000036CB 88D8                <1>       MOV    AL,BL               ; PUT BACK FOR RETURN
11287 000036CD C3                  <1>       RETn
11288                              <1> DC_NON:
11289 000036CE 800D[A8520100]80    <1>       OR     byte [DSKETTE_STATUS], TIME_OUT ; SET TIMEOUT, NO DRIVE
11290 000036D5 EBE7                <1>       JMP    SHORT FINIS
11291                              <1>
11292                              <1> ;--------------------------------------------------------------------------
11293                              <1> ; FORMAT_SET(AH = 17H)
11294                              <1> ;     THIS ROUTINE IS USED TO ESTABLISH THE TYPE OF MEDIA TO BE USED
11295                              <1> ;     FOR THE FOLLOWING FORMAT OPERATION.
11296                              <1> ;
11297                              <1> ; ON ENTRY:  SI LOW = DASD TYPE FOR FORMAT
11298                              <1> ;            DI     = DRIVE #
11299                              <1> ;
11300                              <1> ; ON EXIT:   @DSKETTE_STATUS REFLECTS STATUS
11301                              <1> ;            AH = @DSKETTE_STATUS
11302                              <1> ;            CY = 1 IF ERROR
11303                              <1> ;--------------------------------------------------------------------------
11304                              <1> FORMAT_SET:
11305 000036D7 E884010000          <1>       CALL   XLAT_NEW            ; TRANSLATE STATE TO PRESENT ARCH.
11306 000036DC 6656                <1>       PUSH   SI                  ; SAVE DASD TYPE
11307 000036DE 6689F0              <1>       MOV    AX,SI               ; AH = ? , AL , DASD TYPE
11308 000036E1 30E4                <1>       XOR    AH,AH               ; AH , 0 , AL , DASD TYPE
11309 000036E3 6689C6              <1>       MOV    SI,AX               ; SI = DASD TYPE
11310 000036E6 80A7[B5520100]0F    <1>       AND    byte [DSK_STATE+eDI], ~(MED_DET+DBL_STEP+RATE_MSK) ; CLEAR STATE
11311 000036ED 664E                <1>       DEC    SI                  ; CHECK FOR 320/360K MEDIA & DRIVE
11312 000036EF 7509                <1>       JNZ    short NOT_320       ; BYPASS IF NOT
11313 000036F1 808F[B5520100]90    <1>       OR     byte [DSK_STATE+eDI], MED_DET+RATE_250 ; SET TO 320/360
11314 000036F8 EB48                <1>       JMP    SHORT S0
11315                              <1>
11316                              <1> NOT_320:
11317 000036FA E8B6030000          <1>       CALL   MED_CHANGE          ; CHECK FOR TIME_OUT
11318 000036FF 803D[A8520100]80    <1>       CMP    byte [DSKETTE_STATUS], TIME_OUT
11319 00003706 743A                <1>       JZ     short S0            ; IF TIME OUT TELL CALLER
11320                              <1> S3:
11321 00003708 664E                <1>       DEC    SI                  ; CHECK FOR 320/360K IN 1.2M DRIVE
11322 0000370A 7509                <1>       JNZ    short NOT_320_12    ; BYPASS IF NOT
11323 0000370C 808F[B5520100]70    <1>       OR     byte [DSK_STATE+eDI], MED_DET+DBL_STEP+RATE_300 ; SET STATE
11324 00003713 EB2D                <1>       JMP    SHORT S0
11325                              <1>
11326                              <1> NOT_320_12:
11327 00003715 664E                <1>       DEC    SI                  ; CHECK FOR 1.2M MEDIA IN 1.2M DRIVE
11328 00003717 7509                <1>       JNZ    short NOT_12        ; BYPASS IF NOT
```

```
11329 00003719 808F[B5520100]10    <1>       OR    byte [DSK_STATE+eDI], MED_DET+RATE_500 ; SET STATE VARIABLE
11330 00003720 EB20                <1>       JMP   SHORT S0            ; RETURN TO CALLER
11331                              <1>
11332                              <1> NOT_12:
11333 00003722 664E                <1>       DEC   SI                  ; CHECK FOR SET DASD TYPE 04
11334 00003724 752B                <1>       JNZ   short FS_ERR        ; BAD COMMAND EXIT IF NOT VALID TYPE
11335                              <1>
11336 00003726 F687[B5520100]04    <1>       TEST  byte [DSK_STATE+eDI], DRV_DET ; DRIVE DETERMINED ?
11337 0000372D 740B                <1>       JZ    short ASSUME        ; IF STILL NOT DETERMINED ASSUME
11338 0000372F B050                <1>       MOV   AL,MED_DET+RATE_300
11339 00003731 F687[B5520100]02    <1>         TEST    byte [DSK_STATE+eDI], FMT_CAPA ; MULTIPLE FORMAT CAPABILITY ?
11340 00003738 7502                <1>       JNZ   short OR_IT_IN            ; IF 1.2 M THEN DATA RATE 300
11341                              <1>
11342                              <1> ASSUME:
11343 0000373A B090                <1>       MOV   AL,MED_DET+RATE_250 ; SET UP
11344                              <1>
11345                              <1> OR_IT_IN:
11346 0000373C 0887[B5520100]      <1>       OR    [DSK_STATE+eDI], AL ; OR IN THE CORRECT STATE
11347                              <1> S0:
11348 00003742 E84A010000          <1>       CALL  XLAT_OLD            ; TRANSLATE STATE TO COMPATIBLE MODE
11349 00003747 E884060000          <1>       CALL  SETUP_END           ; VARIOUS CLEANUPS
11350 0000374C 665B                <1>       POP   BX                  ; GET SAVED AL TO BL
11351 0000374E 88D8                <1>       MOV   AL,BL               ; PUT BACK FOR RETURN
11352 00003750 C3                  <1>       RETn
11353                              <1>
11354                              <1> FS_ERR:
11355 00003751 C605[A8520100]01    <1>       MOV   byte [DSKETTE_STATUS], BAD_CMD ; UNKNOWN STATE,BAD COMMAND
11356 00003758 EBE8                <1>       JMP   SHORT S0
11357                              <1>
11358                              <1> ;--------------------------------------------------------------------------------
11359                              <1> ; SET_MEDIA  (AH = 18H)
11360                              <1> ;     THIS ROUTINE SETS THE TYPE OF MEDIA AND DATA RATE
11361                              <1> ;     TO BE USED FOR THE FOLLOWING FORMAT OPERATION.
11362                              <1> ;
11363                              <1> ; ON ENTRY:
11364                              <1> ;     [BP]   = SECTOR PER TRACK
11365                              <1> ;     [BP+1] = TRACK #
11366                              <1> ;     DI     = DRIVE #
11367                              <1> ;
11368                              <1> ; ON EXIT:
11369                              <1> ;     @DSKETTE_STATUS REFLECTS STATUS
11370                              <1> ;     IF NO ERROR:
11371                              <1> ;           AH = 0
11372                              <1> ;           CY = 0
11373                              <1> ;           ES = SEGMENT OF MEDIA/DRIVE PARAMETER TABLE
11374                              <1> ;           DI/[BP+6] = OFFSET OF MEDIA/DRIVE PARAMETER TABLE
11375                              <1> ;     IF ERROR:
11376                              <1> ;           AH = @DSKETTE_STATUS
11377                              <1> ;           CY = 1
11378                              <1> ;--------------------------------------------------------------------------------
11379                              <1> SET_MEDIA:
11380 0000375A E801010000          <1>       CALL  XLAT_NEW            ; TRANSLATE STATE TO PRESENT ARCH.
11381 0000375F F687[B5520100]01    <1>         TEST    byte [DSK_STATE+eDI], TRK_CAPA ; CHECK FOR CHANGE LINE AVAILABLE
11382 00003766 7415                <1>       JZ    short SM_CMOS       ; JUMP IF 40 TRACK DRIVE
11383 00003768 E848030000          <1>       CALL  MED_CHANGE          ; RESET CHANGE LINE
11384 0000376D 803D[A8520100]80    <1>       CMP   byte [DSKETTE_STATUS], TIME_OUT ; IF TIME OUT TELL CALLER
11385 00003774 746B                <1>       JE    short SM_RTN
11386 00003776 C605[A8520100]00    <1>       MOV   byte [DSKETTE_STATUS], 0 ; CLEAR STATUS
11387                              <1> SM_CMOS:
11388 0000377D E819070000          <1>       CALL  CMOS_TYPE           ; RETURN DRIVE TYPE IN (AL)
11389                              <1>       ;;20/02/2015
11390                              <1>       ;;JC  short MD_NOT_FND    ; ERROR IN CMOS
11391                              <1>       ;;OR  AL,AL               ; TEST FOR NO DRIVE
11392 00003782 745D                <1>       JZ    short SM_RTN        ; RETURN IF SO
11393 00003784 E863000000          <1>       CALL  DR_TYPE_CHECK       ; RTN CS:BX = MEDIA/DRIVE PARAM TBL
11394 00003789 7231                <1>       JC    short MD_NOT_FND    ; TYPE NOT IN TABLE (BAD CMOS)
11395 0000378B 57                  <1>       PUSH  eDI                 ; SAVE REG.
11396 0000378C 31DB                <1>       XOR   eBX,eBX                   ; BX = INDEX TO DR. TYPE TABLE
11397 0000378E B906000000          <1>       MOV   eCX,DR_CNT          ; CX = LOOP COUNT
11398                              <1> DR_SEARCH:
11399 00003793 8AA3[705C0000]      <1>       MOV   AH, [DR_TYPE+eBX]   ; GET DRIVE TYPE
11400 00003799 80E47F              <1>       AND   AH,BIT7OFF          ; MASK OUT MSB
11401 0000379C 38E0                <1>       CMP   AL,AH               ; DRIVE TYPE MATCH ?
11402 0000379E 7516                <1>       JNE   short NXT_MD        ; NO, CHECK NEXT DRIVE TYPE
11403                              <1> DR_FND:
11404 000037A0 8BBB[715C0000]      <1>       MOV   eDI, [DR_TYPE+eBX+1]      ; DI = MEDIA/DRIVE PARAM TABLE
11405                              <1> MD_SEARCH:
11406 000037A6 8A6704              <1>         MOV     AH, [eDI+MD.SEC_TRK]    ; GET SECTOR/TRACK
11407 000037A9 386500              <1>       CMP   [eBP],AH            ; MATCH?
11408 000037AC 7508                <1>       JNE   short NXT_MD        ; NO, CHECK NEXT MEDIA
11409 000037AE 8A670B              <1>         MOV     AH, [eDI+MD.MAX_TRK]    ; GET MAX. TRACK #
11410 000037B1 386501              <1>       CMP   [eBP+1],AH          ; MATCH?
11411 000037B4 740F                <1>       JE    short MD_FND        ; YES, GO GET RATE
11412                              <1> NXT_MD:
11413                              <1>       ;ADD  BX,3                ; CHECK NEXT DRIVE TYPE
11414 000037B6 83C305              <1>         add   ebx, 5 ; 18/02/2015
11415 000037B9 E2D8                <1>       LOOP    DR_SEARCH
11416 000037BB 5F                  <1>       POP   eDI                 ; RESTORE REG.
11417                              <1> MD_NOT_FND:
11418 000037BC C605[A8520100]0C    <1>       MOV   byte [DSKETTE_STATUS], MED_NOT_FND ; ERROR, MEDIA TYPE NOT FOUND
11419 000037C3 EB1C                <1>       JMP   SHORT SM_RTN        ; RETURN
11420                              <1> MD_FND:
11421 000037C5 8A470C              <1>         MOV     AL, [eDI+MD.RATE]       ; GET RATE
11422 000037C8 3C40                <1>       CMP   AL,RATE_300         ; DOUBLE STEP REQUIRED FOR RATE 300
11423 000037CA 7502                <1>       JNE   short MD_SET
11424 000037CC 0C20                <1>       OR    AL,DBL_STEP
11425                              <1> MD_SET:
11426                              <1>       ;MOV  [BP+6],DI           ; SAVE TABLE POINTER IN STACK
11427 000037CE 897D0C              <1>       mov   [ebp+12], edi ; 18/02/2015
11428 000037D1 0C10                <1>       OR    AL,MED_DET          ; SET MEDIA ESTABLISHED
11429 000037D3 5F                  <1>       POP   eDI
11430 000037D4 80A7[B5520100]0F    <1>       AND   byte [DSK_STATE+eDI], ~(MED_DET+DBL_STEP+RATE_MSK) ; CLEAR STATE
11431 000037DB 0887[B5520100]      <1>       OR    [DSK_STATE+eDI], AL
```

```
11432                                  <1>       ;MOV   AX, CS             ; SEGMENT OF MEDIA/DRIVE PARAMETER TABLE
11433                                  <1>       ;MOV   ES, AX             ; ES IS SEGMENT OF TABLE
11434                                  <1> SM_RTN:
11435 000037E1 E8AB000000             <1>       CALL   XLAT_OLD           ; TRANSLATE STATE TO COMPATIBLE MODE
11436 000037E6 E8E5050000             <1>       CALL   SETUP_END          ; VARIOUS CLEANUPS
11437 000037EB C3                     <1>       RETn
11438                                  <1>
11439                                  <1> ;--------------------------------------------------------------
11440                                  <1> ; DR_TYPE_CHECK                                          :
11441                                  <1> ;     CHECK IF THE GIVEN DRIVE TYPE IN REGISTER (AL)           :
11442                                  <1> ;     IS SUPPORTED IN BIOS DRIVE TYPE TABLE              :
11443                                  <1> ; ON ENTRY:                                      :
11444                                  <1> ;     AL = DRIVE TYPE                                    :
11445                                  <1> ; ON EXIT:                                       :
11446                                  <1> ;     CS = SEGMENT MEDIA/DRIVE PARAMETER TABLE (CODE)          :
11447                                  <1> ;     CY = 0       DRIVE TYPE SUPPORTED                        :
11448                                  <1> ;          BX = OFFSET TO MEDIA/DRIVE PARAMETER TABLE          :
11449                                  <1> ;     CY = 1 DRIVE TYPE NOT SUPPORTED                  :
11450                                  <1> ; REGISTERS ALTERED: eBX                                 :
11451                                  <1> ;--------------------------------------------------------------
11452                                  <1> DR_TYPE_CHECK:
11453 000037EC 6650                   <1>       PUSH   AX
11454 000037EE 51                     <1>       PUSH   eCX
11455 000037EF 31DB                   <1>       XOR    eBX,eBX                 ; BX = INDEX TO DR_TYPE TABLE
11456 000037F1 B906000000             <1>       MOV    eCX,DR_CNT         ; CX = LOOP COUNT
11457                                  <1> TYPE_CHK:
11458 000037F6 8AA3[705C0000]         <1>       MOV    AH,[DR_TYPE+eBX]   ; GET DRIVE TYPE
11459 000037FC 38E0                   <1>       CMP    AL,AH              ; DRIVE TYPE MATCH?
11460 000037FE 740D                   <1>       JE     short DR_TYPE_VALID ; YES, RETURN WITH CARRY RESET
11461                                  <1>       ;ADD   BX,3               ; CHECK NEXT DRIVE TYPE
11462 00003800 83C305                 <1>         add  ebx, 5 ; 16/02/2015 (32 bit address modification)
11463 00003803 E2F1                   <1>       LOOP    TYPE_CHK
11464                                  <1>       ;
11465 00003805 BB[CF5C0000]           <1>       mov    ebx, MD_TBL6       ; 1.44MB fd parameter table
11466                                  <1>                                 ; Default for GET_PARM (11/12/2014)
11467                                  <1>       ;
11468 0000380A F9                     <1>       STC                       ; DRIVE TYPE NOT FOUND IN TABLE
11469 0000380B EB06                   <1>       JMP    SHORT TYPE_RTN
11470                                  <1> DR_TYPE_VALID:
11471 0000380D 8B9B[715C0000]         <1>       MOV    eBX,[DR_TYPE+eBX+1]       ; BX = MEDIA TABLE
11472                                  <1> TYPE_RTN:
11473 00003813 59                     <1>       POP    eCX
11474 00003814 6658                   <1>       POP    AX
11475 00003816 C3                     <1>       RETn
11476                                  <1>
11477                                  <1> ;--------------------------------------------------------------
11478                                  <1> ; SEND_SPEC                                          :
11479                                  <1> ;     SEND THE SPECIFY COMMAND TO CONTROLLER USING DATA FROM     :
11480                                  <1> ;     THE DRIVE PARAMETER TABLE POINTED BY @DISK_POINTER   :
11481                                  <1> ; ON ENTRY:  @DISK_POINTER = DRIVE PARAMETER TABLE         :
11482                                  <1> ; ON EXIT:   NONE                                    :
11483                                  <1> ; REGISTERS ALTERED: CX, DX                              :
11484                                  <1> ;--------------------------------------------------------------
11485                                  <1> SEND_SPEC:
11486 00003817 50                     <1>       PUSH   eAX                ; SAVE AX
11487 00003818 B8[3E380000]           <1>       MOV    eAX, SPECBAC       ; LOAD ERROR ADDRESS
11488 0000381D 50                     <1>       PUSH   eAX                ; PUSH NEC_OUT ERROR RETURN
11489 0000381E B403                   <1>       MOV    AH,03H             ; SPECIFY COMMAND
11490 00003820 E885070000             <1>       CALL   NEC_OUTPUT         ; OUTPUT THE COMMAND
11491 00003825 28D2                   <1>       SUB    DL,DL              ; FIRST SPECIFY BYTE
11492 00003827 E878060000             <1>       CALL   GET_PARM           ; GET PARAMETER TO AH
11493 0000382C E879070000             <1>       CALL   NEC_OUTPUT         ; OUTPUT THE COMMAND
11494 00003831 B201                   <1>       MOV    DL,1               ; SECOND SPECIFY BYTE
11495 00003833 E86C060000             <1>       CALL   GET_PARM           ; GET PARAMETER TO AH
11496 00003838 E86D070000             <1>       CALL   NEC_OUTPUT         ; OUTPUT THE COMMAND
11497 0000383D 58                     <1>       POP    eAX                ; POP ERROR RETURN
11498                                  <1> SPECBAC:
11499 0000383E 58                     <1>       POP    eAX                ; RESTORE ORIGINAL AX VALUE
11500 0000383F C3                     <1>       RETn
11501                                  <1>
11502                                  <1> ;--------------------------------------------------------------
11503                                  <1> ; SEND_SPEC_MD                                           :
11504                                  <1> ;     SEND THE SPECIFY COMMAND TO CONTROLLER USING DATA FROM     :
11505                                  <1> ;     THE MEDIA/DRIVE PARAMETER TABLE POINTED BY (CS:BX)   :
11506                                  <1> ; ON ENTRY:  CS:BX = MEDIA/DRIVE PARAMETER TABLE           :
11507                                  <1> ; ON EXIT:   NONE                                    :
11508                                  <1> ; REGISTERS ALTERED: AX                                  :
11509                                  <1> ;--------------------------------------------------------------
11510                                  <1> SEND_SPEC_MD:
11511 00003840 50                     <1>       PUSH   eAX                ; SAVE RATE DATA
11512 00003841 B8[5E380000]           <1>       MOV    eAX, SPEC_ESBAC          ; LOAD ERROR ADDRESS
11513 00003846 50                     <1>       PUSH   eAX                ; PUSH NEC_OUT ERROR RETURN
11514 00003847 B403                   <1>       MOV    AH,03H             ; SPECIFY COMMAND
11515 00003849 E85C070000             <1>       CALL   NEC_OUTPUT         ; OUTPUT THE COMMAND
11516 0000384E 8A23                   <1>         MOV     AH, [eBX+MD.SPEC1]      ; GET 1ST SPECIFY BYTE
11517 00003850 E855070000             <1>       CALL   NEC_OUTPUT         ; OUTPUT THE COMMAND
11518 00003855 8A6301                 <1>         MOV     AH, [eBX+MD.SPEC2]      ; GET SECOND SPECIFY BYTE
11519 00003858 E84D070000             <1>       CALL   NEC_OUTPUT         ; OUTPUT THE COMMAND
11520 0000385D 58                     <1>       POP    eAX                ; POP ERROR RETURN
11521                                  <1> SPEC_ESBAC:
11522 0000385E 58                     <1>       POP    eAX                ; RESTORE ORIGINAL AX VALUE
11523 0000385F C3                     <1>       RETn
11524                                  <1>
11525                                  <1> ;--------------------------------------------------------------------------
11526                                  <1> ; XLAT_NEW
11527                                  <1> ;     TRANSLATES DISKETTE STATE LOCATIONS FROM COMPATIBLE
11528                                  <1> ;     MODE TO NEW ARCHITECTURE.
11529                                  <1> ;
11530                                  <1> ; ON ENTRY:  DI = DRIVE #
11531                                  <1> ;--------------------------------------------------------------------------
11532                                  <1> XLAT_NEW:
11533 00003860 83FF01                 <1>       CMP    eDI,1                   ; VALID DRIVE
11534 00003863 7725                   <1>       JA     short XN_OUT            ; IF INVALID BACK
```

```
11535 00003865 80BF[B5520100]00    <1>       CMP    byte [DSK_STATE+eDI], 0          ; NO DRIVE ?
11536 0000386C 741D                <1>       JZ     short DO_DET              ; IF NO DRIVE ATTEMPT DETERMINE
11537 0000386E 6689F9              <1>       MOV    CX,DI                     ; CX = DRIVE NUMBER
11538 00003871 C0E102              <1>       SHL    CL,2                      ; CL = SHIFT COUNT, A=0, B=4
11539 00003874 A0[B4520100]        <1>       MOV    AL, [HF_CNTRL]                  ; DRIVE INFORMATION
11540 00003879 D2C8                <1>       ROR    AL,CL                     ; TO LOW NIBBLE
11541 0000387B 2407                <1>       AND    AL,DRV_DET+FMT_CAPA+TRK_CAPA     ; KEEP DRIVE BITS
11542 0000387D 80A7[B5520100]F8    <1>         AND     byte [DSK_STATE+eDI], ~(DRV_DET+FMT_CAPA+TRK_CAPA)
11543 00003884 0887[B5520100]      <1>       OR     [DSK_STATE+eDI], AL       ; UPDATE DRIVE STATE
11544                              <1> XN_OUT:
11545 0000388A C3                  <1>       RETn
11546                              <1> DO_DET:
11547 0000388B E8BF080000          <1>       CALL   DRIVE_DET                 ; TRY TO DETERMINE
11548 00003890 C3                  <1>       RETn
11549                              <1>
11550                              <1> ;--------------------------------------------------------------------------
11551                              <1> ; XLAT_OLD
11552                              <1> ;     TRANSLATES DISKETTE STATE LOCATIONS FROM NEW
11553                              <1> ;     ARCHITECTURE TO COMPATIBLE MODE.
11554                              <1> ;
11555                              <1> ; ON ENTRY:  DI = DRIVE
11556                              <1> ;--------------------------------------------------------------------------
11557                              <1> XLAT_OLD:
11558 00003891 83FF01              <1>       CMP    eDI,1               ; VALID DRIVE ?
11559                              <1>         ;JA     short XO_OUT            ; IF INVALID BACK
11560 00003894 0F8786000000        <1>         ja      XO_OUT
11561 0000389A 80BF[B5520100]00    <1>         CMP  byte [DSK_STATE+eDI],0    ; NO DRIVE ?
11562 000038A1 747D                <1>       JZ     short XO_OUT        ; IF NO DRIVE TRANSLATE DONE
11563                              <1>
11564                              <1> ;-----       TEST FOR SAVED DRIVE INFORMATION ALREADY SET
11565                              <1>
11566 000038A3 6689F9              <1>       MOV    CX,DI               ; CX = DRIVE NUMBER
11567 000038A6 C0E102              <1>       SHL    CL,2                ; CL = SHIFT COUNT, A=0, B=4
11568 000038A9 B402                <1>       MOV    AH,FMT_CAPA         ; LOAD MULTIPLE DATA RATE BIT MASK
11569 000038AB D2CC                <1>       ROR    AH,CL               ; ROTATE BY MASK
11570 000038AD 8425[B4520100]      <1>       TEST   [HF_CNTRL], AH            ; MULTIPLE-DATA RATE DETERMINED ?
11571 000038B3 751C                <1>       JNZ    short SAVE_SET            ; IF SO, NO NEED TO RE-SAVE
11572                              <1>
11573                              <1> ;-----       ERASE DRIVE BITS IN @HF_CNTRL FOR THIS DRIVE
11574                              <1>
11575 000038B5 B407                <1>       MOV    AH,DRV_DET+FMT_CAPA+TRK_CAPA ; MASK TO KEEP
11576 000038B7 D2CC                <1>       ROR    AH,CL               ; FIX MASK TO KEEP
11577 000038B9 F6D4                <1>       NOT    AH                  ; TRANSLATE MASK
11578 000038BB 2025[B4520100]      <1>       AND    [HF_CNTRL], AH            ; KEEP BITS FROM OTHER DRIVE INTACT
11579                              <1>
11580                              <1> ;-----       ACCESS CURRENT DRIVE BITS AND STORE IN @HF_CNTRL
11581                              <1>
11582 000038C1 8A87[B5520100]      <1>       MOV    AL, [DSK_STATE+eDI] ; ACCESS STATE
11583 000038C7 2407                <1>       AND    AL,DRV_DET+FMT_CAPA+TRK_CAPA ; KEEP DRIVE BITS
11584 000038C9 D2C8                <1>       ROR    AL,CL               ; FIX FOR THIS DRIVE
11585 000038CB 0805[B4520100]      <1>       OR     [HF_CNTRL], AL            ; UPDATE SAVED DRIVE STATE
11586                              <1>
11587                              <1> ;-----       TRANSLATE TO COMPATIBILITY MODE
11588                              <1>
11589                              <1> SAVE_SET:
11590 000038D1 8AA7[B5520100]      <1>       MOV    AH, [DSK_STATE+eDI] ; ACCESS STATE
11591 000038D7 88E7                <1>       MOV    BH,AH               ; TO BH FOR LATER
11592 000038D9 80E4C0              <1>       AND    AH,RATE_MSK         ; KEEP ONLY RATE
11593 000038DC 80FC00              <1>       CMP    AH,RATE_500         ; RATE 500 ?
11594 000038DF 7410                <1>       JZ     short CHK_144       ; YES 1.2/1.2 OR 1.44/1.44
11595 000038E1 B001                <1>       MOV    AL,M3D1U            ; AL = 360 IN 1.2 UNESTABLISHED
11596 000038E3 80FC40              <1>       CMP    AH,RATE_300         ; RATE 300 ?
11597 000038E6 7518                <1>       JNZ    short CHK_250       ; NO, 360/360, 720/720 OR 720/1.44
11598 000038E8 F6C720              <1>       TEST   BH,DBL_STEP         ; CHECK FOR DOUBLE STEP
11599 000038EB 751F                <1>       JNZ    short TST_DET       ; MUST BE 360 IN 1.2
11600                              <1> UNKNO:
11601 000038ED B007                <1>       MOV    AL,MED_UNK          ; NONE OF THE ABOVE
11602 000038EF EB22                <1>       JMP    SHORT AL_SET        ; PROCESS COMPLETE
11603                              <1> CHK_144:
11604 000038F1 E8A5050000          <1>       CALL   CMOS_TYPE           ; RETURN DRIVE TYPE IN (AL)
11605                              <1>       ;;20/02/2015
11606                              <1>       ;;JC   short UNKNO         ; ERROR, SET 'NONE OF ABOVE'
11607 000038F6 74F5                <1>       jz     short UNKNO ;; 20/02/2015
11608 000038F8 3C02                <1>       CMP    AL,2                ; 1.2MB DRIVE ?
11609 000038FA 75F1                <1>       JNE    short UNKNO         ; NO, GO SET 'NONE OF ABOVE'
11610 000038FC B002                <1>       MOV    AL,M1D1U            ; AL = 1.2 IN 1.2 UNESTABLISHED
11611 000038FE EB0C                <1>       JMP    SHORT TST_DET
11612                              <1> CHK_250:
11613 00003900 B000                <1>       MOV    AL,M3D3U            ; AL = 360 IN 360 UNESTABLISHED
11614 00003902 80FC80              <1>       CMP    AH,RATE_250         ; RATE 250 ?
11615 00003905 75E6                <1>       JNZ    short UNKNO         ; IF SO FALL IHRU
11616 00003907 F6C701              <1>       TEST   BH,TRK_CAPA         ; 80 TRACK CAPABILITY ?
11617 0000390A 75E1                <1>       JNZ    short UNKNO         ; IF SO JUMP, FALL THRU TEST DET
11618                              <1> TST_DET:
11619 0000390C F6C710              <1>       TEST   BH,MED_DET          ; DETERMINED ?
11620 0000390F 7402                <1>       JZ     short AL_SET        ; IF NOT THEN SET
11621 00003911 0403                <1>       ADD    AL,3                ; MAKE DETERMINED/ESTABLISHED
11622                              <1> AL_SET:
11623 00003913 80A7[B5520100]F8    <1>       AND    byte [DSK_STATE+eDI], ~(DRV_DET+FMT_CAPA+TRK_CAPA) ; CLEAR DRIVE
11624 0000391A 0887[B5520100]      <1>       OR     [DSK_STATE+eDI], AL ; REPLACE WITH COMPATIBLE MODE
11625                              <1> XO_OUT:
11626 00003920 C3                  <1>       RETn
11627                              <1>
11628                              <1> ;--------------------------------------------------------------------------
11629                              <1> ; RD_WR_VF
11630                              <1> ;     COMMON READ, WRITE AND VERIFY:
11631                              <1> ;     MAIN LOOP FOR STATE RETRIES.
11632                              <1> ;
11633                              <1> ; ON ENTRY:  AH = READ/WRITE/VERIFY NEC PARAMETER
11634                              <1> ;            AL = READ/WRITE/VERIFY DMA PARAMETER
11635                              <1> ;
11636                              <1> ; ON EXIT:   @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION
11637                              <1> ;--------------------------------------------------------------------------
```

```
11638                                  <1> RD_WR_VF:
11639 00003921 6650                    <1>         PUSH   AX                   ; SAVE DMA, NEC PARAMETERS
11640 00003923 E838FFFFFF              <1>         CALL   XLAT_NEW             ; TRANSLATE STATE TO PRESENT ARCH.
11641 00003928 E8F3000000              <1>         CALL   SETUP_STATE          ; INITIALIZE START AND END RATE
11642 0000392D 6658                    <1>         POP    AX                   ; RESTORE READ/WRITE/VERIFY
11643                                  <1> DO_AGAIN:
11644 0000392F 6650                    <1>         PUSH   AX                   ; SAVE READ/WRITE/VERIFY PARAMETER
11645 00003931 E87F010000              <1>         CALL   MED_CHANGE           ; MEDIA CHANGE AND RESET IF CHANGED
11646 00003936 6658                    <1>         POP    AX                   ; RESTORE READ/WRITE/VERIFY
11647 00003938 0F82C9000000            <1>           JC     RWV_END                ; MEDIA CHANGE ERROR OR TIME-OUT
11648                                  <1> RWV:
11649 0000393E 6650                    <1>         PUSH   AX                   ; SAVE READ/WRITE/VERIFY PARAMETER
11650 00003940 8AB7[B5520100]          <1>         MOV    DH, [DSK_STATE+eDI] ; GET RATE STATE OF THIS DRIVE
11651 00003946 80E6C0                  <1>         AND    DH,RATE_MSK          ; KEEP ONLY RATE
11652 00003949 E84D050000              <1>         CALL   CMOS_TYPE            ; RETURN DRIVE TYPE IN AL (AL)
11653                                  <1>         ;;20/02/2015
11654                                  <1>         ;;JC   short RWV_ASSUME    ; ERROR IN CMOS
11655 0000394E 7451                    <1>         jz     short RWV_ASSUME ; 20/02/2015
11656 00003950 3C01                    <1>         CMP    AL,1                 ; 40 TRACK DRIVE?
11657 00003952 750D                    <1>         JNE    short RWV_1          ; NO, BYPASS CMOS VALIDITY CHECK
11658 00003954 F687[B5520100]01        <1>         TEST   byte [DSK_STATE+eDI], TRK_CAPA ; CHECK FOR 40 TRACK DRIVE
11659 0000395B 7413                    <1>         JZ     short RWV_2          ; YES, CMOS IS CORRECT
11660 0000395D B002                    <1>         MOV    AL,2                 ; CHANGE TO 1.2M
11661 0000395F EB0F                    <1>         JMP    SHORT RWV_2
11662                                  <1> RWV_1:
11663 00003961 720D                    <1>         JB     short RWV_2          ; NO DRIVE SPECIFIED, CONTINUE
11664 00003963 F687[B5520100]01        <1>         TEST    byte [DSK_STATE+eDI], TRK_CAPA ; IS IT REALLY 40 TRACK?
11665 0000396A 7504                    <1>         JNZ    short RWV_2          ; NO, 80 TRACK
11666 0000396C B001                    <1>         MOV    AL,1                 ; IT IS 40 TRACK, FIX CMOS VALUE
11667 0000396E EB04                    <1>         jmp    short rwv_3
11668                                  <1> RWV_2:
11669 00003970 08C0                    <1>         OR     AL,AL                ; TEST FOR NO DRIVE
11670 00003972 742D                    <1>         JZ     short RWV_ASSUME    ; ASSUME TYPE, USE MAX TRACK
11671                                  <1> rwv_3:
11672 00003974 E873FEFFFF              <1>         CALL   DR_TYPE_CHECK        ; RTN CS:BX = MEDIA/DRIVE PARAM TBL.
11673 00003979 7226                    <1>         JC     short RWV_ASSUME    ; TYPE NOT IN TABLE (BAD CMOS)
11674                                  <1>
11675                                  <1> ;-----      SEARCH FOR MEDIA/DRIVE PARAMETER TABLE
11676                                  <1>
11677 0000397B 57                      <1>         PUSH   eDI                  ; SAVE DRIVE #
11678 0000397C 31DB                    <1>         XOR    eBX,eBX                   ; BX = INDEX TO DR_TYPE TABLE
11679 0000397E B906000000              <1>         MOV    eCX,DR_CNT           ; CX = LOOP COUNT
11680                                  <1> RWV_DR_SEARCH:
11681 00003983 8AA3[705C0000]          <1>         MOV    AH, [DR_TYPE+eBX]   ; GET DRIVE TYPE
11682 00003989 80E47F                  <1>         AND    AH,BIT7OFF           ; MASK OUT MSB
11683 0000398C 38E0                    <1>         CMP    AL,AH                ; DRIVE TYPE MATCH?
11684 0000398E 750B                    <1>         JNE    short RWV_NXT_MD    ; NO, CHECK NEXT DRIVE TYPE
11685                                  <1> RWV_DR_FND:
11686 00003990 8BBB[715C0000]          <1>         MOV    eDI, [DR_TYPE+eBX+1]      ; DI = MEDIA/DRIVE PARAMETER TABLE
11687                                  <1> RWV_MD_SEARH:
11688 00003996 3A770C                  <1>           CMP     DH, [eDI+MD.RATE]       ; MATCH?
11689 00003999 741B                    <1>         JE     short RWV_MD_FND    ; YES, GO GET 1ST SPECIFY BYTE
11690                                  <1> RWV_NXT_MD:
11691                                  <1>         ;ADD   BX,3                 ; CHECK NEXT DRIVE TYPE
11692 0000399B 83C305                  <1>         add    eBX, 5
11693 0000399E E2E3                    <1>         LOOP   RWV_DR_SEARCH
11694 000039A0 5F                      <1>         POP    eDI                  ; RESTORE DRIVE #
11695                                  <1>
11696                                  <1> ;-----      ASSUME PRIMARY DRIVE IS INSTALLED AS SHIPPED
11697                                  <1>
11698                                  <1> RWV_ASSUME:
11699 000039A1 BB[8E5C0000]            <1>         MOV    eBX, MD_TBL1         ; POINT TO 40 TRACK 250 KBS
11700 000039A6 F687[B5520100]01        <1>         TEST   byte [DSK_STATE+eDI], TRK_CAPA ; TEST FOR 80 TRACK
11701 000039AD 740A                    <1>         JZ     short RWV_MD_FND1   ; MUST BE 40 TRACK
11702 000039AF BB[A85C0000]            <1>         MOV    eBX, MD_TBL3         ; POINT TO 80 TRACK 500 KBS
11703 000039B4 EB03                    <1>         JMP    short RWV_MD_FND1   ; GO SPECIFY PARAMTERS
11704                                  <1>
11705                                  <1> ;-----      CS:BX POINTS TO MEDIA/DRIVE PARAMETER TABLE
11706                                  <1>
11707                                  <1> RWV_MD_FND:
11708 000039B6 89FB                    <1>         MOV    eBX,eDI                   ; BX = MEDIA/DRIVE PARAMETER TABLE
11709 000039B8 5F                      <1>         POP    eDI                  ; RESTORE DRIVE #
11710                                  <1>
11711                                  <1> ;-----      SEND THE SPECIFY COMMAND TO THE CONTROLLER
11712                                  <1>
11713                                  <1> RWV_MD_FND1:
11714 000039B9 E882FEFFFF              <1>         CALL   SEND_SPEC_MD
11715 000039BE E864010000              <1>         CALL   CHK_LASTRATE         ; ZF=1 ATTEMP RATE IS SAME AS LAST RATE
11716 000039C3 7405                    <1>         JZ     short RWV_DBL        ; YES,SKIP SEND RATE COMMAND
11717 000039C5 E83B010000              <1>         CALL   SEND_RATE            ; SEND DATA RATE TO NEC
11718                                  <1> RWV_DBL:
11719 000039CA 53                      <1>         PUSH   eBX                  ; SAVE MEDIA/DRIVE PARAM TBL ADDRESS
11720 000039CB E822040000              <1>         CALL   SETUP_DBL            ; CHECK FOR DOUBLE STEP
11721 000039D0 5B                      <1>         POP    eBX                  ; RESTORE ADDRESS
11722 000039D1 7226                    <1>         JC     short CHK_RET        ; ERROR FROM READ ID, POSSIBLE RETRY
11723 000039D3 6658                    <1>         POP    AX                   ; RESTORE NEC, DMA COMMAND
11724 000039D5 6650                    <1>         PUSH   AX                   ; SAVE NEC COMMAND
11725 000039D7 53                      <1>         PUSH   eBX                  ; SAVE MEDIA/DRIVE PARAM TBL ADDRESS
11726 000039D8 E861010000              <1>         CALL   DMA_SETUP            ; SET UP THE DMA
11727 000039DD 5B                      <1>         POP    eBX
11728 000039DE 6658                    <1>         POP    AX                   ; RESTORE NEC COMMAND
11729 000039E0 722F                    <1>         JC     short RWV_BAC        ; CHECK FOR DMA BOUNDARY ERROR
11730 000039E2 6650                    <1>         PUSH   AX                   ; SAVE NEC COMMAND
11731 000039E4 53                      <1>         PUSH   eBX                  ; SAVE MEDIA/DRIVE PARAM TBL ADDRESS
11732 000039E5 E83C020000              <1>         CALL   NEC_INIT             ; INITIALIZE NEC
11733 000039EA 5B                      <1>         POP    eBX                  ; RESTORE ADDRESS
11734 000039EB 720C                    <1>         JC     short CHK_RET        ; ERROR - EXIT
11735 000039ED E866020000              <1>         CALL   RWV_COM                   ; OP CODE COMMON TO READ/WRITE/VERIFY
11736 000039F2 7205                    <1>         JC     short CHK_RET        ; ERROR - EXIT
11737 000039F4 E8AB020000              <1>         CALL   NEC_TERM             ; TERMINATE, GET STATUS, ETC.
11738                                  <1> CHK_RET:
11739 000039F9 E84A030000              <1>         CALL   RETRY                ; CHECK FOR, SETUP RETRY
11740 000039FE 6658                    <1>         POP    AX                   ; RESTORE READ/WRITE/VERIFY PARAMETER
```

```
11741 00003A00 7305              <1>       JNC    short RWV_END       ; CY = 0 NO RETRY
11742 00003A02 E928FFFFFF        <1>         JMP     DO_AGAIN                ; CY = 1 MEANS RETRY
11743                            <1> RWV_END:
11744 00003A07 E8F4020000        <1>       CALL   DSTATE              ; ESTABLISH STATE IF SUCCESSFUL
11745 00003A0C E887030000        <1>       CALL   NUM_TRANS           ; AL = NUMBER TRANSFERRED
11746                            <1> RWV_BAC:                         ; BAD DMA ERROR ENTRY
11747 00003A11 6650              <1>       PUSH   AX                  ; SAVE NUMBER TRANSFERRED
11748 00003A13 E879FEFFFF        <1>       CALL   XLAT_OLD            ; TRANSLATE STATE TO COMPATIBLE MODE
11749 00003A18 6658              <1>       POP    AX                  ; RESTORE NUMBER TRANSFERRED
11750 00003A1A E8B1030000        <1>       CALL   SETUP_END           ; VARIOUS CLEANUPS
11751 00003A1F C3                <1>       RETn
11752                            <1>
11753                            <1> ;--------------------------------------------------------------------------
11754                            <1> ; SETUP_STATE:     INITIALIZES START AND END RATES.
11755                            <1> ;--------------------------------------------------------------------------
11756                            <1> SETUP_STATE:
11757 00003A20 F687[B5520100]10  <1>       TEST   byte [DSK_STATE+eDI], MED_DET ; MEDIA DETERMINED ?
11758 00003A27 7537              <1>       JNZ    short J1C           ; NO STATES IF DETERMINED
11759 00003A29 66B84000          <1>         MOV     AX,(RATE_500*256)+RATE_300  ; AH = START RATE, AL = END RATE
11760 00003A2D F687[B5520100]04  <1>       TEST   byte [DSK_STATE+eDI],DRV_DET ; DRIVE ?
11761 00003A34 740D              <1>       JZ     short AX_SET        ; DO NOT KNOW DRIVE
11762 00003A36 F687[B5520100]02  <1>       TEST   byte [DSK_STATE+eDI], FMT_CAPA ; MULTI-RATE?
11763 00003A3D 7504              <1>       JNZ    short AX_SET        ; JUMP IF YES
11764 00003A3F 66B88080          <1>         MOV     AX,RATE_250*257         ; START A END RATE 250 FOR 360 DRIVE
11765                            <1> AX_SET:
11766 00003A43 80A7[B5520100]1F  <1>       AND    byte [DSK_STATE+eDI], ~(RATE_MSK+DBL_STEP) ; TURN OFF THE RATE
11767 00003A4A 08A7[B5520100]    <1>       OR     [DSK_STATE+eDI], AH ; RATE FIRST TO TRY
11768 00003A50 8025[B0520100]F3  <1>       AND    byte [LASTRATE], ~STRT_MSK ; ERASE LAST TO TRY RATE BITS
11769 00003A57 C0C804            <1>       ROR    AL,4                ; TO OPERATION LAST RATE LOCATION
11770 00003A5A 0805[B0520100]    <1>       OR     [LASTRATE], AL            ; LAST RATE
11771                            <1> J1C:
11772 00003A60 C3                <1>       RETn
11773                            <1>
11774                            <1> ;--------------------------------------------------------------------------
11775                            <1> ;  FMT_INIT: ESTABLISH STATE IF UNESTABLISHED AT FORMAT TIME.
11776                            <1> ;--------------------------------------------------------------------------
11777                            <1> FMT_INIT:
11778 00003A61 F687[B5520100]10  <1>       TEST   byte [DSK_STATE+eDI], MED_DET ; IS MEDIA ESTABLISHED
11779 00003A68 7546              <1>       JNZ    short F1_OUT        ; IF SO RETURN
11780 00003A6A E82C040000        <1>       CALL   CMOS_TYPE           ; RETURN DRIVE TYPE IN AL
11781                            <1>       ;; 20/02/2015
11782                            <1>       ;;JC   short CL_DRV        ; ERROR IN CMOS ASSUME NO DRIVE
11783 00003A6F 7440              <1>       jz     short CL_DRV ;; 20/02/2015
11784 00003A71 FEC8              <1>       DEC    AL                  ; MAKE ZERO ORIGIN
11785                            <1>       ;;JS   short CL_DRV        ; NO DRIVE IF AL 0
11786 00003A73 8AA7[B5520100]    <1>       MOV    AH, [DSK_STATE+eDI] ; AH = CURRENT STATE
11787 00003A79 80E40F            <1>       AND    AH, ~(MED_DET+DBL_STEP+RATE_MSK) ; CLEAR
11788 00003A7C 08C0              <1>       OR     AL,AL               ; CHECK FOR 360
11789 00003A7E 7505              <1>       JNZ    short N_360         ; IF 360 WILL BE 0
11790 00003A80 80CC90            <1>       OR     AH,MED_DET+RATE_250 ; ESTABLISH MEDIA
11791 00003A83 EB25              <1>       JMP    SHORT SKP_STATE           ; SKIP OTHER STATE PROCESSING
11792                            <1> N_360:
11793 00003A85 FEC8              <1>       DEC    AL                  ; 1.2 M DRIVE
11794 00003A87 7505              <1>       JNZ    short N_12          ; JUMP IF NOT
11795                            <1> F1_RATE:
11796 00003A89 80CC10            <1>       OR     AH,MED_DET+RATE_500 ; SET FORMAT RATE
11797 00003A8C EB1C              <1>       JMP    SHORT SKP_STATE           ; SKIP OTHER STATE PROCESSING
11798                            <1> N_12:
11799 00003A8E FEC8              <1>       DEC    AL                  ; CHECK FOR TYPE 3
11800 00003A90 750F              <1>       JNZ    short N_720         ; JUMP IF NOT
11801 00003A92 F6C404            <1>       TEST   AH,DRV_DET          ; IS DRIVE DETERMINED
11802 00003A95 7410              <1>       JZ     short ISNT_12       ; TREAT AS NON 1.2 DRIVE
11803 00003A97 F6C402            <1>       TEST   AH,FMT_CAPA         ; IS 1.2M
11804 00003A9A 740B              <1>       JZ     short ISNT_12       ; JUMP IF NOT
11805 00003A9C 80CC50            <1>       OR     AH,MED_DET+RATE_300 ; RATE 300
11806 00003A9F EB09              <1>       JMP    SHORT SKP_STATE           ; CONTINUE
11807                            <1> N_720:
11808 00003AA1 FEC8              <1>       DEC    AL                  ; CHECK FOR TYPE 4
11809 00003AA3 750C              <1>       JNZ    short CL_DRV        ; NO DRIVE, CMOS BAD
11810 00003AA5 EBE2              <1>       JMP    SHORT F1_RATE
11811                            <1> ISNT_12:
11812 00003AA7 80CC90            <1>       OR     AH,MED_DET+RATE_250 ; MUST BE RATE 250
11813                            <1>
11814                            <1> SKP_STATE:
11815 00003AAA 88A7[B5520100]    <1>       MOV    [DSK_STATE+eDI], AH ; STORE AWAY
11816                            <1> F1_OUT:
11817 00003AB0 C3                <1>       RETn
11818                            <1> CL_DRV:
11819 00003AB1 30E4              <1>       XOR    AH,AH               ; CLEAR STATE
11820 00003AB3 EBF5              <1>       JMP    SHORT SKP_STATE           ; SAVE IT
11821                            <1>
11822                            <1> ;--------------------------------------------------------------------------
11823                            <1> ; MED_CHANGE
11824                            <1> ;     CHECKS FOR MEDIA CHANGE, RESETS MEDIA CHANGE,
11825                            <1> ;     CHECKS MEDIA CHANGE AGAIN.
11826                            <1> ;
11827                            <1> ; ON EXIT:  CY = 1 MEANS MEDIA CHANGE OR TIMEOUT
11828                            <1> ;           @DSKETTE_STATUS = ERROR CODE
11829                            <1> ;--------------------------------------------------------------------------
11830                            <1> MED_CHANGE:
11831 00003AB5 E888060000        <1>       CALL   READ_DSKCHNG        ; READ DISK CHANCE LINE STATE
11832 00003ABA 7447              <1>       JZ     short MC_OUT        ; BYPASS HANDLING DISK CHANGE LINE
11833 00003ABC 80A7[B5520100]EF  <1>       AND    byte [DSK_STATE+eDI], ~MED_DET ; CLEAR STATE FOR THIS DRIVE
11834                            <1>
11835                            <1> ;     THIS SEQUENCE ENSURES WHENEVER A DISKETTE IS CHANGED THAT
11836                            <1> ;     ON THE NEXT OPERATION THE REQUIRED MOTOR START UP TIME WILL
11837                            <1> ;     BE WAITED. (DRIVE MOTOR MAY GO OFF UPON DOOR OPENING).
11838                            <1>
11839 00003AC3 6689F9            <1>       MOV    CX,DI               ; CL = DRIVE 0
11840 00003AC6 B001              <1>       MOV    AL,1                ; MOTOR ON BIT MASK
11841 00003AC8 D2E0              <1>       SHL    AL,CL               ; TO APPROPRIATE POSITION
11842 00003ACA F6D0              <1>       NOT    AL                  ; KEEP ALL BUT MOTOR ON
11843 00003ACC FA                <1>       CLI                        ; NO INTERRUPTS
```

```
11844 00003ACD 2005[A6520100]      <1>       AND    [MOTOR_STATUS], AL  ; TURN MOTOR OFF INDICATOR
11845 00003AD3 FB                  <1>       STI                        ; INTERRUPTS ENABLED
11846 00003AD4 E810040000          <1>       CALL   MOTOR_ON            ; TURN MOTOR ON
11847                              <1>
11848                              <1> ;-----      THIS SEQUENCE OF SEEKS IS USED TO RESET DISKETTE CHANGE SIGNAL
11849                              <1>
11850 00003AD9 E86DF9FFFF          <1>       CALL   DSK_RESET           ; RESET NEC
11851 00003ADE B501                <1>       MOV    CH,01H              ; MOVE TO CYLINDER 1
11852 00003AE0 E8FF040000          <1>       CALL   SEEK                ; ISSUE SEEK
11853 00003AE5 30ED                <1>       XOR    CH,CH               ; MOVE TO CYLINDER 0
11854 00003AE7 E8F8040000          <1>       CALL   SEEK                ; ISSUE SEEK
11855 00003AEC C605[A8520100]06    <1>       MOV    byte [DSKETTE_STATUS], MEDIA_CHANGE ; STORE IN STATUS
11856                              <1> OK1:
11857 00003AF3 E84A060000          <1>       CALL   READ_DSKCHNG        ; CHECK MEDIA CHANGED AGAIN
11858 00003AF8 7407                <1>       JZ     short OK2           ; IF ACTIVE, NO DISKETTE, TIMEOUT
11859                              <1> OK4:
11860 00003AFA C605[A8520100]80    <1>       MOV    byte [DSKETTE_STATUS], TIME_OUT ; TIMEOUT IF DRIVE EMPTY
11861                              <1> OK2:
11862 00003B01 F9                  <1>       STC                        ; MEDIA CHANGED, SET CY
11863 00003B02 C3                  <1>       RETn
11864                              <1> MC_OUT:
11865 00003B03 F8                  <1>       CLC                        ; NO MEDIA CHANGED, CLEAR CY
11866 00003B04 C3                  <1>       RETn
11867                              <1>
11868                              <1> ;--------------------------------------------------------------------------
11869                              <1> ; SEND_RATE
11870                              <1> ;     SENDS DATA RATE COMMAND TO NEC
11871                              <1> ; ON ENTRY:  DI = DRIVE #
11872                              <1> ; ON EXIT:   NONE
11873                              <1> ; REGISTERS ALTERED: DX
11874                              <1> ;--------------------------------------------------------------------------
11875                              <1> SEND_RATE:
11876 00003B05 6650                <1>       PUSH   AX                  ; SAVE REG.
11877 00003B07 8025[B0520100]3F    <1>       AND    byte [LASTRATE], ~SEND_MSK ; ELSE CLEAR LAST RATE ATTEMPTED
11878 00003B0E 8A87[B5520100]      <1>       MOV    AL, [DSK_STATE+eDI] ; GET RATE STATE OF THIS DRIVE
11879 00003B14 24C0                <1>       AND    AL,SEND_MSK         ; KEEP ONLY RATE BITS
11880 00003B16 0805[B0520100]      <1>       OR     [LASTRATE], AL            ; SAVE NEW RATE FOR NEXT CHECK
11881 00003B1C C0C002              <1>       ROL    AL,2                ; MOVE TO BIT OUTPUT POSITIONS
11882 00003B1F 66BAF703            <1>       MOV    DX,03F7H            ; OUTPUT NEW DATA RATE
11883 00003B23 EE                  <1>       OUT    DX,AL
11884 00003B24 6658                <1>       POP    AX                  ; RESTORE REG.
11885 00003B26 C3                  <1>       RETn
11886                              <1>
11887                              <1> ;--------------------------------------------------------------------------
11888                              <1> ; CHK_LASTRATE
11889                              <1> ;     CHECK PREVIOUS DATE RATE SNT TO THE CONTROLLER.
11890                              <1> ; ON ENTRY:
11891                              <1> ;     DI = DRIVE #
11892                              <1> ; ON EXIT:
11893                              <1> ;     ZF =  1 DATA RATE IS THE SAME AS THE LAST RATE SENT TO NEC
11894                              <1> ;     ZF =  0 DATA RATE IS DIFFERENT FROM LAST RATE
11895                              <1> ; REGISTERS ALTERED: DX
11896                              <1> ;--------------------------------------------------------------------------
11897                              <1> CHK_LASTRATE:
11898 00003B27 6650                <1>       PUSH   AX                  ; SAVE REG
11899 00003B29 2225[B0520100]      <1>       AND    AH, [LASTRATE]            ; GET LAST DATA RATE SELECTED
11900 00003B2F 8A87[B5520100]      <1>       MOV    AL, [DSK_STATE+eDI] ; GET RATE STATE OF THIS DRIVE
11901 00003B35 6625C0C0            <1>         AND     AX, SEND_MSK*257        ; KEEP ONLY RATE BITS OF BOTH
11902 00003B39 38E0                <1>       CMP    AL, AH              ; COMPARE TO PREVIOUSLY TRIED
11903                              <1>                                  ; ZF = 1 RATE IS THE SAME
11904 00003B3B 6658                <1>       POP    AX                  ; RESTORE REG.
11905 00003B3D C3                  <1>       RETn
11906                              <1>
11907                              <1> ;--------------------------------------------------------------------------
11908                              <1> ; DMA_SETUP
11909                              <1> ;     THIS ROUTINE SETS UP THE DMA FOR READ/WRITE/VERIFY OPERATIONS.
11910                              <1> ;
11911                              <1> ; ON ENTRY:  AL = DMA COMMAND
11912                              <1> ;
11913                              <1> ; ON EXIT:   @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION
11914                              <1> ;--------------------------------------------------------------------------
11915                              <1>
11916                              <1> ; SI = Head #, # of Sectors or DASD Type
11917                              <1>
11918                              <1> ; 22/08/2015
11919                              <1> ; 08/02/2015 - Protected Mode Modification
11920                              <1> ; 06/02/2015 - 07/02/2015
11921                              <1> ; NOTE: Buffer address must be in 1st 16MB of Physical Memory (24 bit limit).
11922                              <1> ; (DMA Addres = Physical Address)
11923                              <1> ; (Retro UNIX 386 v1 Kernel/System Mode Virtual Address = Physical Address)
11924                              <1> ;
11925                              <1>
11926                              <1>
11927                              <1> ; 04/02/2016 (clc)
11928                              <1> ; 20/02/2015 modification (source: AWARD BIOS 1999, DMA_SETUP)
11929                              <1> ; 16/12/2014 (IODELAY)
11930                              <1>
11931                              <1> DMA_SETUP:
11932                              <1>
11933                              <1> ;; 20/02/2015
11934 00003B3E 8B5504              <1>       mov    edx, [ebp+4]        ; Buffer address
11935 00003B41 F7C2000000FF        <1>       test   edx, 0FF000000h           ; 16 MB limit (22/08/2015, bugfix)
11936 00003B47 756E                <1>       jnz    short dma_bnd_err_stc
11937                              <1>       ;
11938 00003B49 6650                <1>       push   ax                  ; DMA command
11939 00003B4B 52                  <1>       push   edx                 ; *
11940 00003B4C B203                <1>       mov    dl, 3               ; GET BYTES/SECTOR PARAMETER
11941 00003B4E E851030000          <1>       call   GET_PARM            ;
11942 00003B53 88E1                <1>       mov    cl, ah                   ; SHIFT COUNT (0=128, 1=256, 2=512 ETC)
11943 00003B55 6689F0              <1>       mov    ax, si              ; Sector count
11944 00003B58 88C4                <1>       mov    ah, al              ; AH =  # OF SECTORS
11945 00003B5A 28C0                <1>       sub    al, al              ; AL = 0, AX = # SECTORS * 256
11946 00003B5C 66D1E8              <1>       shr    ax, 1               ; AX = # SECTORS * 128
```

```
11947 00003B5F 66D3E0              <1>       shl    ax, cl              ; SHIFT BY PARAMETER VALUE
11948 00003B62 6648                <1>       dec    ax                  ; -1 FOR DMA VALUE
11949 00003B64 6689C1              <1>       mov    cx, ax
11950 00003B67 5A                  <1>       pop    edx                 ; *
11951 00003B68 6658                <1>       pop    ax
11952 00003B6A 3C42                <1>       cmp    al, 42h
11953 00003B6C 7507                <1>         jne     short NOT_VERF
11954 00003B6E BA0000FF00          <1>       mov    edx, 0FF0000h
11955 00003B73 EB08                <1>       jmp    short J33
11956                              <1> NOT_VERF:
11957 00003B75 6601CA              <1>       add    dx, cx              ; check for overflow
11958 00003B78 723E                <1>       jc     short dma_bnd_err
11959                              <1>       ;
11960 00003B7A 6629CA              <1>       sub    dx, cx              ; Restore start address
11961                              <1> J33:
11962 00003B7D FA                  <1>       CLI                        ; DISABLE INTERRUPTS DURING DMA SET-UP
11963 00003B7E E60C                <1>       OUT    DMA+12,AL           ; SET THE FIRST/LA5T F/F
11964                              <1>       IODELAY                          ; WAIT FOR I/O
11965 00003B80 EB00                <2>  jmp short $+2
11966 00003B82 EB00                <2>  jmp short $+2
11967 00003B84 E60B                <1>       OUT    DMA+11,AL           ; OUTPUT THE MODE BYTE
11968 00003B86 89D0                <1>       mov    eax, edx            ; Buffer address
11969 00003B88 E604                <1>       OUT    DMA+4,AL            ; OUTPUT LOW ADDRESS
11970                              <1>       IODELAY                          ; WAIT FOR I/O
11971 00003B8A EB00                <2>  jmp short $+2
11972 00003B8C EB00                <2>  jmp short $+2
11973 00003B8E 88E0                <1>       MOV    AL,AH
11974 00003B90 E604                <1>       OUT    DMA+4,AL            ; OUTPUT HIGH ADDRESS
11975 00003B92 C1E810              <1>       shr    eax, 16
11976                              <1>       IODELAY                          ; I/O WAIT STATE
11977 00003B95 EB00                <2>  jmp short $+2
11978 00003B97 EB00                <2>  jmp short $+2
11979 00003B99 E681                <1>       OUT    081H,AL                  ; OUTPUT highest BITS TO PAGE REGISTER
11980                              <1>       IODELAY
11981 00003B9B EB00                <2>  jmp short $+2
11982 00003B9D EB00                <2>  jmp short $+2
11983 00003B9F 6689C8              <1>       mov    ax, cx              ; Byte count - 1
11984 00003BA2 E605                <1>       OUT    DMA+5,AL            ; LOW BYTE OF COUNT
11985                              <1>       IODELAY                          ; WAIT FOR I/O
11986 00003BA4 EB00                <2>  jmp short $+2
11987 00003BA6 EB00                <2>  jmp short $+2
11988 00003BA8 88E0                <1>       MOV    AL, AH
11989 00003BAA E605                <1>       OUT    DMA+5,AL            ; HIGH BYTE OF COUNT
11990                              <1>       IODELAY
11991 00003BAC EB00                <2>  jmp short $+2
11992 00003BAE EB00                <2>  jmp short $+2
11993 00003BB0 FB                  <1>       STI                        ; RE-ENABLE INTERRUPTS
11994 00003BB1 B002                <1>       MOV    AL, 2               ; MODE FOR 8237
11995 00003BB3 E60A                <1>       OUT    DMA+10, AL          ; INITIALIZE THE DISKETTE CHANNEL
11996                              <1>
11997 00003BB5 F8                  <1>       clc    ; 04/02/2016
11998 00003BB6 C3                  <1>       retn
11999                              <1>
12000                              <1> dma_bnd_err_stc:
12001 00003BB7 F9                  <1>       stc
12002                              <1> dma_bnd_err:
12003 00003BB8 C605[A8520100]09    <1>       MOV    byte [DSKETTE_STATUS], DMA_BOUNDARY ; SET ERROR
12004 00003BBF C3                  <1>       RETn                       ; CY SET BY ABOVE IF ERROR
12005                              <1>
12006                              <1> ;; 16/12/2014
12007                              <1> ;;    CLI                        ; DISABLE INTERRUPTS DURING DMA SET-UP
12008                              <1> ;;    OUT    DMA+12,AL           ; SET THE FIRST/LA5T F/F
12009                              <1> ;;    ;JMP   $+2                 ; WAIT FOR I/O
12010                              <1> ;;    IODELAY
12011                              <1> ;;    OUT    DMA+11,AL           ; OUTPUT THE MODE BYTE
12012                              <1> ;;    ;SIODELAY
12013                              <1> ;;      ;CMP AL, 42H                   ; DMA VERIFY COMMAND
12014                              <1> ;;      ;JNE short NOT_VERF            ; NO
12015                              <1> ;;      ;XOR AX, AX              ; START ADDRESS
12016                              <1> ;;      ;JMP SHORT J33
12017                              <1> ;;;NOT_VERF:
12018                              <1> ;;    ;MOV   AX,ES               ; GET THE ES VALUE
12019                              <1> ;;    ;ROL   AX,4                ; ROTATE LEFT
12020                              <1> ;;    ;MOV   CH,AL               ; GET HIGHEST NIBBLE OF ES TO CH
12021                              <1> ;;    ;AND   AL,11110000B        ; ZERO THE LOW NIBBLE FROM SEGMENT
12022                              <1> ;;    ;ADD   AX,[BP+2]           ; TEST FOR CARRY FROM ADDITION
12023                              <1> ;;    mov    eax, [ebp+4] ; 06/02/2015
12024                              <1> ;;    ;JNC   short J33
12025                              <1> ;;    ;INC   CH                  ; CARRY MEANS HIGH 4 BITS MUST BE INC
12026                              <1> ;;;J33:
12027                              <1> ;;    PUSH   eAX                 ; SAVE START ADDRESS
12028                              <1> ;;    OUT    DMA+4,AL            ; OUTPUT LOW ADDRESS
12029                              <1> ;;    ;JMP   $+2                 ; WAIT FOR I/O
12030                              <1> ;;    IODELAY
12031                              <1> ;;    MOV    AL,AH
12032                              <1> ;;    OUT    DMA+4,AL            ; OUTPUT HIGH ADDRESS
12033                              <1> ;;    shr    eax, 16           ; 07/02/2015
12034                              <1> ;;    ;MOV   AL,CH               ; GET HIGH 4 BITS
12035                              <1> ;;    ;JMP   $+2                 ; I/O WAIT STATE
12036                              <1> ;;    IODELAY
12037                              <1> ;;    ;AND   AL,00001111B
12038                              <1> ;;    OUT    081H,AL                  ; OUTPUT HIGH 4 BITS TO PAGE REGISTER
12039                              <1> ;;    ;SIODELAY
12040                              <1> ;;
12041                              <1> ;;;----- DETERMINE COUNT
12042                              <1> ;;    sub    eax, eax ; 08/02/2015
12043                              <1> ;;    MOV    AX, SI              ; AL =  # OF SECTORS
12044                              <1> ;;    XCHG   AL, AH              ; AH =  # OF SECTORS
12045                              <1> ;;    SUB    AL, AL              ; AL = 0, AX = # SECTORS * 256
12046                              <1> ;;    SHR    AX, 1               ; AX = # SECTORS * 128
12047                              <1> ;;    PUSH   AX                  ; SAVE # OF SECTORS * 128
12048                              <1> ;;    MOV    DL, 3               ; GET BYTES/SECTOR PARAMETER
12049                              <1> ;;    CALL   GET_PARM            ; "
```

```
12050                                  <1> ;;    MOV    CL,AH               ; SHIFT COUNT (0=128, 1=256, 2=512 ETC)
12051                                  <1> ;;    POP    AX                  ; AX = # SECTORS * 128
12052                                  <1> ;;    SHL    AX,CL               ; SHIFT BY PARAMETER VALUE
12053                                  <1> ;;    DEC    AX                  ; -1 FOR DMA VALUE
12054                                  <1> ;;    PUSH   eAX  ; 08/02/2015   ; SAVE COUNT VALUE
12055                                  <1> ;;    OUT    DMA+5,AL            ; LOW BYTE OF COUNT
12056                                  <1> ;;    ;JMP   $+2                 ; WAIT FOR I/O
12057                                  <1> ;;    IODELAY
12058                                  <1> ;;    MOV    AL, AH
12059                                  <1> ;;    OUT    DMA+5,AL            ; HIGH BYTE OF COUNT
12060                                  <1> ;;    ;IODELAY
12061                                  <1> ;;    STI                        ; RE-ENABLE INTERRUPTS
12062                                  <1> ;;    POP    eCX  ; 08/02/2015   ; RECOVER COUNT VALUE
12063                                  <1> ;;    POP    eAX  ; 08/02/2015   ; RECOVER ADDRESS VALUE
12064                                  <1> ;;    ;ADD   AX, CX              ; ADD, TEST FOR 64K OVERFLOW
12065                                  <1> ;;    add    ecx, eax ; 08/02/2015
12066                                  <1> ;;    MOV    AL, 2               ; MODE FOR 8237
12067                                  <1> ;;    ;JMP   $+2                 ; WAIT FOR I/O
12068                                  <1> ;;    SIODELAY
12069                                  <1> ;;    OUT    DMA+10, AL          ; INITIALIZE THE DISKETTE CHANNEL
12070                                  <1> ;;    ;JNC   short NO_BAD        ; CHECK FOR ERROR
12071                                  <1> ;;    jc     short dma_bnd_err ; 08/02/2015
12072                                  <1> ;;    and    ecx, 0FFF00000h ; 16 MB limit
12073                                  <1> ;;    jz     short NO_BAD
12074                                  <1> ;;dma_bnd_err:
12075                                  <1> ;;    MOV    byte [DSKETTE_STATUS], DMA_BOUNDARY ; SET ERROR
12076                                  <1> ;;NO_BAD:
12077                                  <1> ;;    RETn                       ; CY SET BY ABOVE IF ERROR
12078                                  <1>
12079                                  <1> ;--------------------------------------------------------------------------
12080                                  <1> ; FMTDMA_SET
12081                                  <1> ;     THIS ROUTINE SETS UP THE DMA CONTROLLER FOR A FORMAT OPERATION.
12082                                  <1> ;
12083                                  <1> ; ON ENTRY:  NOTHING REQUIRED
12084                                  <1> ;
12085                                  <1> ; ON EXIT:  @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION
12086                                  <1> ;--------------------------------------------------------------------------
12087                                  <1>
12088                                  <1> FMTDMA_SET:
12089                                  <1> ;; 20/02/2015 modification
12090 00003BC0 8B5504                 <1>       mov    edx, [ebp+4]        ; Buffer address
12091 00003BC3 F7C20000F0FF           <1>       test   edx, 0FFF00000h           ; 16 MB limit
12092 00003BC9 75EC                   <1>       jnz    short dma_bnd_err_stc
12093                                  <1>       ;
12094 00003BCB 6652                   <1>       push   dx                  ; *
12095 00003BCD B204                   <1>       mov    DL, 4               ; SECTORS/TRACK VALUE IN PARM TABLE
12096 00003BCF E8D0020000             <1>       call   GET_PARM            ; "
12097 00003BD4 88E0                   <1>       mov    al, ah              ; AL = SECTORS/TRACK VALUE
12098 00003BD6 28E4                   <1>       sub    ah, ah              ; AX = SECTORS/TRACK VALUE
12099 00003BD8 66C1E002               <1>       shl    ax, 2               ; AX = SEC/TRK * 4 (OFFSET C,H,R,N)
12100 00003BDC 6648                   <1>       dec    ax                  ; -1 FOR DMA VALUE
12101 00003BDE 6689C1                 <1>       mov    cx, ax
12102 00003BE1 665A                   <1>       pop    dx                  ; *
12103 00003BE3 6601CA                 <1>       add    dx, cx              ; check for overflow
12104 00003BE6 72D0                   <1>       jc     short dma_bnd_err
12105                                  <1>       ;
12106 00003BE8 6629CA                 <1>       sub    dx, cx              ; Restore start address
12107                                  <1>       ;
12108 00003BEB B04A                   <1>       MOV    AL, 04AH            ; WILL WRITE TO THE DISKETTE
12109 00003BED FA                     <1>       CLI                        ; DISABLE INTERRUPTS DURING DMA SET-UP
12110 00003BEE E60C                   <1>       OUT    DMA+12,AL           ; SET THE FIRST/LA5T F/F
12111                                  <1>       IODELAY                          ; WAIT FOR I/O
12112 00003BF0 EB00                   <2>  jmp short $+2
12113 00003BF2 EB00                   <2>  jmp short $+2
12114 00003BF4 E60B                   <1>       OUT    DMA+11,AL           ; OUTPUT THE MODE BYTE
12115 00003BF6 89D0                   <1>       mov    eax, edx            ; Buffer address
12116 00003BF8 E604                   <1>       OUT    DMA+4,AL            ; OUTPUT LOW ADDRESS
12117                                  <1>       IODELAY                          ; WAIT FOR I/O
12118 00003BFA EB00                   <2>  jmp short $+2
12119 00003BFC EB00                   <2>  jmp short $+2
12120 00003BFE 88E0                   <1>       MOV    AL,AH
12121 00003C00 E604                   <1>       OUT    DMA+4,AL            ; OUTPUT HIGH ADDRESS
12122 00003C02 C1E810                 <1>       shr    eax, 16
12123                                  <1>       IODELAY                          ; I/O WAIT STATE
12124 00003C05 EB00                   <2>  jmp short $+2
12125 00003C07 EB00                   <2>  jmp short $+2
12126 00003C09 E681                   <1>       OUT    081H,AL                   ; OUTPUT highest BITS TO PAGE REGISTER
12127                                  <1>       IODELAY
12128 00003C0B EB00                   <2>  jmp short $+2
12129 00003C0D EB00                   <2>  jmp short $+2
12130 00003C0F 6689C8                 <1>       mov    ax, cx              ; Byte count - 1
12131 00003C12 E605                   <1>       OUT    DMA+5,AL            ; LOW BYTE OF COUNT
12132                                  <1>       IODELAY                          ; WAIT FOR I/O
12133 00003C14 EB00                   <2>  jmp short $+2
12134 00003C16 EB00                   <2>  jmp short $+2
12135 00003C18 88E0                   <1>       MOV    AL, AH
12136 00003C1A E605                   <1>       OUT    DMA+5,AL            ; HIGH BYTE OF COUNT
12137                                  <1>       IODELAY
12138 00003C1C EB00                   <2>  jmp short $+2
12139 00003C1E EB00                   <2>  jmp short $+2
12140 00003C20 FB                     <1>       STI                        ; RE-ENABLE INTERRUPTS
12141 00003C21 B002                   <1>       MOV    AL, 2               ; MODE FOR 8237
12142 00003C23 E60A                   <1>       OUT    DMA+10, AL          ; INITIALIZE THE DISKETTE CHANNEL
12143 00003C25 C3                     <1>       retn
12144                                  <1>
12145                                  <1> ;; 08/02/2015 - Protected Mode Modification
12146                                  <1> ;;    MOV    AL, 04AH            ; WILL WRITE TO THE DISKETTE
12147                                  <1> ;;    CLI                        ; DISABLE INTERRUPTS DURING DMA SET-UP
12148                                  <1> ;;    OUT    DMA+12,AL           ; SET THE FIRST/LA5T F/F
12149                                  <1> ;;    ;JMP   $+2                 ; WAIT FOR I/O
12150                                  <1> ;;    IODELAY
12151                                  <1> ;;    OUT    DMA+11,AL           ; OUTPUT THE MODE BYTE
12152                                  <1> ;;    ;MOV   AX,ES               ; GET THE ES VALUE
```

```
12153                           <1> ;;    ;ROL   AX,4               ; ROTATE LEFT
12154                           <1> ;;    ;MOV   CH,AL              ; GET HIGHEST NIBBLE OF ES TO CH
12155                           <1> ;;    ;AND   AL,11110000B       ; ZERO THE LOW NIBBLE FROM SEGMENT
12156                           <1> ;;    ;ADD   AX,[BP+2]          ; TEST FOR CARRY FROM ADDITION
12157                           <1> ;;    ;JNC   short J33A
12158                           <1> ;;    ;INC   CH                 ; CARRY MEANS HIGH 4 BITS MUST BE INC
12159                           <1> ;;    mov    eax, [ebp+4] ; 08/02/2015
12160                           <1> ;;;J33A:
12161                           <1> ;;    PUSH   eAX ; 08/02/2015   ; SAVE START ADDRESS
12162                           <1> ;;    OUT    DMA+4,AL           ; OUTPUT LOW ADDRESS
12163                           <1> ;;    ;JMP   $+2                ; WAIT FOR I/O
12164                           <1> ;;    IODELAY
12165                           <1> ;;    MOV    AL,AH
12166                           <1> ;;    OUT    DMA+4,AL           ; OUTPUT HIGH ADDRESS
12167                           <1> ;;    shr    eax, 16 ; 08/02/2015
12168                           <1> ;;    ;MOV   AL,CH              ; GET HIGH 4 BITS
12169                           <1> ;;    ;JMP   $+2                ; I/O WAIT STATE
12170                           <1> ;;    IODELAY
12171                           <1> ;;    ;AND   AL,00001111B
12172                           <1> ;;    OUT    081H,AL                  ; OUTPUT HIGH 4 BITS TO PAGE REGISTER
12173                           <1> ;;
12174                           <1> ;;;----- DETERMINE COUNT
12175                           <1> ;;    sub    eax, eax ; 08/02/2015
12176                           <1> ;;    MOV    DL, 4              ; SECTORS/TRACK VALUE IN PARM TABLE
12177                           <1> ;;    CALL   GET_PARM           ; "
12178                           <1> ;;    XCHG   AL, AH             ; AL = SECTORS/TRACK VALUE
12179                           <1> ;;    SUB    AH, AH             ; AX = SECTORS/TRACK VALUE
12180                           <1> ;;    SHL    AX, 2              ; AX = SEC/TRK * 4 (OFFSET C,H,R,N)
12181                           <1> ;;    DEC    AX                 ; -1 FOR DMA VALUE
12182                           <1> ;;    PUSH   eAX    ; 08/02/2015 ; SAVE # OF BYTES TO BE TRANSFERED
12183                           <1> ;;    OUT    DMA+5,AL           ; LOW BYTE OF COUNT
12184                           <1> ;;    ;JMP   $+2                ; WAIT FOR I/O
12185                           <1> ;;    IODELAY
12186                           <1> ;;    MOV    AL, AH
12187                           <1> ;;    OUT    DMA+5,AL           ; HIGH BYTE OF COUNT
12188                           <1> ;;    STI                       ; RE-ENABLE INTERRUPTS
12189                           <1> ;;    POP    eCX    ; 08/02/2015 ; RECOVER COUNT VALUE
12190                           <1> ;;    POP    eAX    ; 08/02/2015 ; RECOVER ADDRESS VALUE
12191                           <1> ;;    ;ADD   AX, CX             ; ADD, TEST FOR 64K OVERFLOW
12192                           <1> ;;    add    ecx, eax ; 08/02/2015
12193                           <1> ;;    MOV    AL, 2              ; MODE FOR 8237
12194                           <1> ;;    ;JMP   $+2                ; WAIT FOR I/O
12195                           <1> ;;    SIODELAY
12196                           <1> ;;    OUT    DMA+10, AL         ; INITIALIZE THE DISKETTE CHANNEL
12197                           <1> ;;    ;JNC   short FMTDMA_OK           ; CHECK FOR ERROR
12198                           <1> ;;    jc     short fmtdma_bnd_err ; 08/02/2015
12199                           <1> ;;    and    ecx, 0FFF00000h  ; 16 MB limit
12200                           <1> ;;    jz     short FMTDMA_OK
12201                           <1> ;;    stc    ; 20/02/2015
12202                           <1> ;;fmtdma_bnd_err:
12203                           <1> ;;    MOV    byte [DSKETTE_STATUS], DMA_BOUNDARY ; SET ERROR
12204                           <1> ;;FMTDMA_OK:
12205                           <1> ;;    RETn                      ; CY SET BY ABOVE IF ERROR
12206                           <1>
12207                           <1> ;--------------------------------------------------------------------------
12208                           <1> ; NEC_INIT
12209                           <1> ;     THIS ROUTINE SEEKS TO THE REQUESTED TRACK AND INITIALIZES
12210                           <1> ;     THE NEC FOR THE READ/WRITE/VERIFY/FORMAT OPERATION.
12211                           <1> ;
12212                           <1> ; ON ENTRY:  AH = NEC COMMAND TO BE PERFORMED
12213                           <1> ;
12214                           <1> ; ON EXIT:   @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION
12215                           <1> ;--------------------------------------------------------------------------
12216                           <1> NEC_INIT:
12217 00003C26 6650             <1>       PUSH   AX                 ; SAVE NEC COMMAND
12218 00003C28 E8BC020000       <1>       CALL   MOTOR_ON           ; TURN MOTOR ON FOR SPECIFIC DRIVE
12219                           <1>
12220                           <1> ;-----      DO THE SEEK OPERATION
12221                           <1>
12222 00003C2D 8A6D01           <1>       MOV    CH,[eBP+1]         ; CH = TRACK #
12223 00003C30 E8AF030000       <1>       CALL   SEEK               ; MOVE TO CORRECT TRACK
12224 00003C35 6658             <1>       POP    AX                 ; RECOVER COMMAND
12225 00003C37 721E             <1>       JC     short ER_1         ; ERROR ON SEEK
12226 00003C39 BB[573C0000]     <1>       MOV    eBX, ER_1          ; LOAD ERROR ADDRESS
12227 00003C3E 53               <1>       PUSH   eBX                ; PUSH NEC_OUT ERROR RETURN
12228                           <1>
12229                           <1> ;-----      SEND OUT THE PARAMETERS TO THE CONTROLLER
12230                           <1>
12231 00003C3F E866030000       <1>       CALL   NEC_OUTPUT         ; OUTPUT THE OPERATION COMMAND
12232 00003C44 6689F0           <1>       MOV    AX,SI              ; AH = HEAD #
12233 00003C47 89FB             <1>       MOV    eBX,eDI                 ; BL = DRIVE #
12234 00003C49 C0E402           <1>       SAL    AH,2               ; MOVE IT TO BIT 2
12235 00003C4C 80E404           <1>       AND    AH,00000100B       ; ISOLATE THAT BIT
12236 00003C4F 08DC             <1>       OR     AH,BL              ; OR IN THE DRIVE NUMBER
12237 00003C51 E854030000       <1>       CALL   NEC_OUTPUT         ; FALL THRU CY SET IF ERROR
12238 00003C56 5B               <1>       POP    eBX                ; THROW AWAY ERROR RETURN
12239                           <1> ER_1:
12240 00003C57 C3               <1>       RETn
12241                           <1>
12242                           <1> ;--------------------------------------------------------------------------
12243                           <1> ; RWV_COM
12244                           <1> ;     THIS ROUTINE SENDS PARAMETERS TO THE NEC SPECIFIC TO THE
12245                           <1> ;     READ/WRITE/VERIFY OPERATIONS.
12246                           <1> ;
12247                           <1> ; ON ENTRY:  CS:BX = ADDRESS OF MEDIA/DRIVE PARAMETER TABLE
12248                           <1> ; ON EXIT:   @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION
12249                           <1> ;--------------------------------------------------------------------------
12250                           <1> RWV_COM:
12251 00003C58 B8[A33C0000]     <1>       MOV    eAX, ER_2          ; LOAD ERROR ADDRESS
12252 00003C5D 50               <1>       PUSH   eAX                ; PUSH NEC_OUT ERROR RETURN
12253 00003C5E 8A6501           <1>       MOV    AH,[eBP+1]         ; OUTPUT TRACK #
12254 00003C61 E844030000       <1>       CALL   NEC_OUTPUT
12255 00003C66 6689F0           <1>       MOV    AX,SI              ; OUTPUT HEAD #
```

```
12256 00003C69 E83C030000          <1>       CALL   NEC_OUTPUT
12257 00003C6E 8A6500              <1>         MOV     AH,[eBP]               ; OUTPUT SECTOR #
12258 00003C71 E834030000          <1>       CALL   NEC_OUTPUT
12259 00003C76 B203                <1>       MOV    DL,3                ; BYTES/SECTOR PARAMETER FROM BLOCK
12260 00003C78 E827020000          <1>       CALL   GET_PARM            ; ... TO THE NEC
12261 00003C7D E828030000          <1>       CALL   NEC_OUTPUT          ; OUTPUT TO CONTROLLER
12262 00003C82 B204                <1>       MOV    DL,4                ; EOT PARAMETER FROM BLOCK
12263 00003C84 E81B020000          <1>       CALL   GET_PARM            ; ... TO THE NEC
12264 00003C89 E81C030000          <1>       CALL   NEC_OUTPUT          ; OUTPUT TO CONTROLLER
12265 00003C8E 8A6305              <1>         MOV     AH, [eBX+MD.GAP]         ; GET GAP LENGTH
12266                              <1> _R15:
12267 00003C91 E814030000          <1>       CALL   NEC_OUTPUT
12268 00003C96 B206                <1>       MOV    DL,6                ; DTL PARAMETER PROM BLOCK
12269 00003C98 E807020000          <1>       CALL   GET_PARM            ;  TO THE NEC
12270 00003C9D E808030000          <1>       CALL   NEC_OUTPUT          ; OUTPUT TO CONTROLLER
12271 00003CA2 58                  <1>       POP    eAX                 ; THROW AWAY ERROR EXIT
12272                              <1> ER_2:
12273 00003CA3 C3                  <1>       RETn
12274                              <1>
12275                              <1> ;--------------------------------------------------------------------------
12276                              <1> ; NEC_TERM
12277                              <1> ;     THIS ROUTINE WAITS FOR THE OPERATION THEN ACCEPTS THE STATUS
12278                              <1> ;     FROM THE NEC FOR THE READ/WRITE/VERIFY/FORWAT OPERATION.
12279                              <1> ;
12280                              <1> ; ON EXIT:  @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION
12281                              <1> ;--------------------------------------------------------------------------
12282                              <1> NEC_TERM:
12283                              <1>
12284                              <1> ;-----      LET THE OPERATION HAPPEN
12285                              <1>
12286 00003CA4 56                  <1>       PUSH   eSI                 ; SAVE HEAD #, # OF SECTORS
12287 00003CA5 E80D040000          <1>       CALL   WAIT_INT            ; WAIT FOR THE INTERRUPT
12288 00003CAA 9C                  <1>       PUSHF
12289 00003CAB E837040000          <1>       CALL   RESULTS                   ; GET THE NEC STATUS
12290 00003CB0 724B                <1>       JC     short SET_END_POP
12291 00003CB2 9D                  <1>       POPF
12292 00003CB3 723E                <1>       JC     short SET_END       ; LOOK FOR ERROR
12293                              <1>
12294                              <1> ;-----      CHECK THE RESULTS RETURNED BY THE CONTROLLER
12295                              <1>
12296 00003CB5 FC                  <1>       CLD                        ; SET THE CORRECT DIRECTION
12297 00003CB6 BE[A9520100]        <1>       MOV    eSI, NEC_STATUS           ; POINT TO STATUS FIELD
12298 00003CBB AC                  <1>       lodsb                      ; GET ST0
12299 00003CBC 24C0                <1>       AND    AL,11000000B        ; TEST FOR NORMAL TERMINATION
12300 00003CBE 7433                <1>       JZ     short SET_END
12301 00003CC0 3C40                <1>       CMP    AL,01000000B        ; TEST FOR ABNORMAL TERMINATION
12302 00003CC2 7527                <1>       JNZ    short J18           ; NOT ABNORMAL, BAD NEC
12303                              <1>
12304                              <1> ;-----      ABNORMAL TERMINATION, FIND OUT WHY
12305                              <1>
12306 00003CC4 AC                  <1>       lodsb                      ; GET ST1
12307 00003CC5 D0E0                <1>       SAL    AL,1                ; TEST FOR EDT FOUND
12308 00003CC7 B404                <1>       MOV    AH,RECORD_NOT_FND
12309 00003CC9 7222                <1>       JC     short J19
12310 00003CCB C0E002              <1>       SAL    AL,2
12311 00003CCE B410                <1>       MOV    AH,BAD_CRC
12312 00003CD0 721B                <1>       JC     short J19
12313 00003CD2 D0E0                <1>       SAL    AL,1                ; TEST FOR DMA OVERRUN
12314 00003CD4 B408                <1>       MOV    AH,BAD_DMA
12315 00003CD6 7215                <1>       JC     short J19
12316 00003CD8 C0E002              <1>       SAL    AL,2                ; TEST FOR RECORD NOT FOUND
12317 00003CDB B404                <1>       MOV    AH,RECORD_NOT_FND
12318 00003CDD 720E                <1>       JC     short J19
12319 00003CDF D0E0                <1>       SAL    AL,1
12320 00003CE1 B403                <1>       MOV    AH,WRITE_PROTECT    ; TEST FOR WRITE_PROTECT
12321 00003CE3 7208                <1>       JC     short J19
12322 00003CE5 D0E0                <1>       SAL    AL,1                ; TEST MISSING ADDRESS MARK
12323 00003CE7 B402                <1>       MOV    AH,BAD_ADDR_MARK
12324 00003CE9 7202                <1>       JC     short J19
12325                              <1>
12326                              <1> ;-----      NEC MUST HAVE FAILED
12327                              <1> J18:
12328 00003CEB B420                <1>       MOV    AH,BAD_NEC
12329                              <1> J19:
12330 00003CED 0825[A8520100]      <1>       OR     [DSKETTE_STATUS], AH
12331                              <1> SET_END:
12332 00003CF3 803D[A8520100]01    <1>       CMP    byte [DSKETTE_STATUS], 1 ; SET ERROR CONDITION
12333 00003CFA F5                  <1>       CMC
12334 00003CFB 5E                  <1>       POP    eSI
12335 00003CFC C3                  <1>       RETn                       ; RESTORE HEAD #, # OF SECTORS
12336                              <1>
12337                              <1> SET_END_POP:
12338 00003CFD 9D                  <1>       POPF
12339 00003CFE EBF3                <1>       JMP    SHORT SET_END
12340                              <1>
12341                              <1> ;--------------------------------------------------------------------------
12342                              <1> ; DSTATE:   ESTABLISH STATE UPON SUCCESSFUL OPERATION.
12343                              <1> ;--------------------------------------------------------------------------
12344                              <1> DSTATE:
12345 00003D00 803D[A8520100]00    <1>       CMP    byte [DSKETTE_STATUS],0   ; CHECK FOR ERROR
12346 00003D07 753E                <1>       JNZ    short SETBAC            ; IF ERROR JUMP
12347 00003D09 808F[B5520100]10    <1>       OR     byte [DSK_STATE+eDI],MED_DET ; NO ERROR, MARK MEDIA AS DETERMINED
12348 00003D10 F687[B5520100]04    <1>       TEST   byte [DSK_STATE+eDI],DRV_DET ; DRIVE DETERMINED ?
12349 00003D17 752E                <1>       JNZ    short SETBAC        ; IF DETERMINED NO TRY TO DETERMINE
12350 00003D19 8A87[B5520100]      <1>       MOV    AL,[DSK_STATE+eDI]  ; LOAD STATE
12351 00003D1F 24C0                <1>       AND    AL,RATE_MSK         ; KEEP ONLY RATE
12352 00003D21 3C80                <1>       CMP    AL,RATE_250         ; RATE 250 ?
12353 00003D23 751B                <1>       JNE    short M_12          ; NO, MUST BE 1.2M OR 1.44M DRIVE
12354                              <1>
12355                              <1> ;-----      CHECK IF IT IS 1.44M
12356                              <1>
12357 00003D25 E871010000          <1>       CALL   CMOS_TYPE           ; RETURN DRIVE TYPE IN (AL)
12358                              <1>       ;;20/02/2015
```

```
12359                                  <1>       ;;JC   short M_12          ; CMOS BAD
12360 00003D2A 7414                    <1>       jz     short M_12 ;; 20/02/2015
12361 00003D2C 3C04                    <1>       CMP    AL, 4               ; 1.44MB DRIVE ?
12362 00003D2E 7410                    <1>       JE     short M_12          ; YES
12363                                  <1> M_720:
12364 00003D30 80A7[B5520100]FD        <1>       AND    byte [DSK_STATE+eDI], ~FMT_CAPA ; TURN OFF FORMAT CAPABILITY
12365 00003D37 808F[B5520100]04        <1>       OR     byte [DSK_STATE+eDI],DRV_DET  ; MARK DRIVE DETERMINED
12366 00003D3E EB07                    <1>       JMP    SHORT SETBAC        ; BACK
12367                                  <1> M_12:
12368 00003D40 808F[B5520100]06        <1>       OR     byte [DSK_STATE+eDI],DRV_DET+FMT_CAPA
12369                                  <1>                                  ; TURN ON DETERMINED & FMT CAPA
12370                                  <1> SETBAC:
12371 00003D47 C3                      <1>       RETn
12372                                  <1>
12373                                  <1> ;--------------------------------------------------------------------------
12374                                  <1> ; RETRY
12375                                  <1> ;     DETERMINES WHETHER A RETRY IS NECESSARY.
12376                                  <1> ;     IF RETRY IS REQUIRED THEN STATE INFORMATION IS UPDATED FOR RETRY.
12377                                  <1> ;
12378                                  <1> ; ON EXIT:  CY = 1 FOR RETRY, CY = 0 FOR NO RETRY
12379                                  <1> ;--------------------------------------------------------------------------
12380                                  <1> RETRY:
12381 00003D48 803D[A8520100]00        <1>       CMP    byte [DSKETTE_STATUS],0   ; GET STATUS OF OPERATION
12382 00003D4F 7445                    <1>       JZ     short NO_RETRY            ; SUCCESSFUL OPERATION
12383 00003D51 803D[A8520100]80        <1>       CMP    byte [DSKETTE_STATUS],TIME_OUT ; IF TIME OUT NO RETRY
12384 00003D58 743C                    <1>       JZ     short NO_RETRY
12385 00003D5A 8AA7[B5520100]          <1>       MOV    AH,[DSK_STATE+eDI]  ; GET MEDIA STATE OF DRIVE
12386 00003D60 F6C410                  <1>       TEST   AH,MED_DET          ; ESTABLISHED/DETERMINED ?
12387 00003D63 7531                    <1>       JNZ    short NO_RETRY            ; IF ESTABLISHED STATE THEN TRUE ERROR
12388 00003D65 80E4C0                  <1>       AND    AH,RATE_MSK         ; ISOLATE RATE
12389 00003D68 8A2D[B0520100]          <1>       MOV    CH,[LASTRATE]       ; GET START OPERATION STATE
12390 00003D6E C0C504                  <1>       ROL    CH,4                ; TO CORRESPONDING BITS
12391 00003D71 80E5C0                  <1>       AND    CH,RATE_MSK         ; ISOLATE RATE BITS
12392 00003D74 38E5                    <1>       CMP    CH,AH               ; ALL RATES TRIED
12393 00003D76 741E                    <1>       JE     short NO_RETRY            ; IF YES, THEN TRUE ERROR
12394                                  <1>
12395                                  <1> ;     SETUP STATE INDICATOR FOR RETRY ATTEMPT TO NEXT RATE
12396                                  <1> ;      00000000B (500) -> 10000000B    (250)
12397                                  <1> ;      10000000B (250) -> 01000000B    (300)
12398                                  <1> ;      01000000B (300) -> 00000000B    (500)
12399                                  <1>
12400 00003D78 80FC01                  <1>       CMP    AH,RATE_500+1       ; SET CY FOR RATE 500
12401 00003D7B D0DC                    <1>       RCR    AH,1                ; TO NEXT STATE
12402 00003D7D 80E4C0                  <1>       AND    AH,RATE_MSK         ; KEEP ONLY RATE BITS
12403 00003D80 80A7[B5520100]1F        <1>       AND    byte [DSK_STATE+eDI], ~(RATE_MSK+DBL_STEP)
12404                                  <1>                                  ; RATE, DBL STEP OFF
12405 00003D87 08A7[B5520100]          <1>       OR     [DSK_STATE+eDI],AH  ; TURN ON NEW RATE
12406 00003D8D C605[A8520100]00        <1>       MOV    byte [DSKETTE_STATUS],0   ; RESET STATUS FOR RETRY
12407 00003D94 F9                      <1>       STC                        ; SET CARRY FOR RETRY
12408 00003D95 C3                      <1>       RETn                       ; RETRY RETURN
12409                                  <1>
12410                                  <1> NO_RETRY:
12411 00003D96 F8                      <1>       CLC                        ; CLEAR CARRY NO RETRY
12412 00003D97 C3                      <1>       RETn                       ; NO RETRY RETURN
12413                                  <1>
12414                                  <1> ;--------------------------------------------------------------------------
12415                                  <1> ; NUM_TRANS
12416                                  <1> ;     THIS ROUTINE CALCULATES THE NUMBER OF SECTORS THAT WERE
12417                                  <1> ;     ACTUALLY TRANSFERRED TO/FROM THE DISKETTE.
12418                                  <1> ;
12419                                  <1> ; ON ENTRY:  [BP+1] = TRACK
12420                                  <1> ;            SI-HI  = HEAD
12421                                  <1> ;            [BP]   = START SECTOR
12422                                  <1> ;
12423                                  <1> ; ON EXIT:   AL = NUMBER ACTUALLY TRANSFERRED
12424                                  <1> ;--------------------------------------------------------------------------
12425                                  <1> NUM_TRANS:
12426 00003D98 30C0                    <1>       XOR    AL,AL               ; CLEAR FOR ERROR
12427 00003D9A 803D[A8520100]00        <1>       CMP    byte [DSKETTE_STATUS],0   ; CHECK FOR ERROR
12428 00003DA1 752C                    <1>       JNZ    NT_OUT              ; IF ERROR 0 TRANSFERRED
12429 00003DA3 B204                    <1>       MOV    DL,4                ; SECTORS/TRACK OFFSET TO DL
12430 00003DA5 E8FA000000              <1>       CALL   GET_PARM            ; AH = SECTORS/TRACK
12431 00003DAA 8A1D[AE520100]          <1>       MOV    BL, [NEC_STATUS+5]  ; GET ENDING SECTOR
12432 00003DB0 6689F1                  <1>       MOV    CX,SI               ; CH = HEAD # STARTED
12433 00003DB3 3A2D[AD520100]          <1>       CMP    CH, [NEC_STATUS+4]  ; GET HEAD ENDED UP ON
12434 00003DB9 750D                    <1>       JNZ    DIF_HD              ; IF ON SAME HEAD, THEN NO ADJUST
12435 00003DBB 8A2D[AC520100]          <1>       MOV    CH, [NEC_STATUS+3]  ; GET TRACK ENDED UP ON
12436 00003DC1 3A6D01                  <1>       CMP    CH,[eBP+1]          ; IS IT ASKED FOR TRACK
12437 00003DC4 7404                    <1>       JZ     short SAME_TRK            ; IF SAME TRACK NO INCREASE
12438 00003DC6 00E3                    <1>       ADD    BL,AH               ; ADD SECTORS/TRACK
12439                                  <1> DIF_HD:
12440 00003DC8 00E3                    <1>       ADD    BL,AH               ; ADD SECTORS/TRACK
12441                                  <1> SAME_TRK:
12442 00003DCA 2A5D00                  <1>       SUB    BL,[eBP]            ; SUBTRACT START FROM END
12443 00003DCD 88D8                    <1>       MOV    AL,BL               ; TO AL
12444                                  <1> NT_OUT:
12445 00003DCF C3                      <1>       RETn
12446                                  <1>
12447                                  <1> ;--------------------------------------------------------------------------
12448                                  <1> ; SETUP_END
12449                                  <1> ;     RESTORES @MOTOR_COUNT TO PARAMETER PROVIDED IN TABLE
12450                                  <1> ;     AND LOADS @DSKETTE_STATUS TO AH, AND SETS CY.
12451                                  <1> ;
12452                                  <1> ; ON EXIT:
12453                                  <1> ;     AH, @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION
12454                                  <1> ;--------------------------------------------------------------------------
12455                                  <1> SETUP_END:
12456 00003DD0 B202                    <1>       MOV    DL,2                ; GET THE MOTOR WAIT PARAMETER
12457 00003DD2 6650                    <1>       PUSH   AX                  ; SAVE NUMBER TRANSFERRED
12458 00003DD4 E8CB000000              <1>       CALL   GET_PARM
12459 00003DD9 8825[A7520100]          <1>       MOV    [MOTOR_COUNT],AH    ; STORE UPON RETURN
12460 00003DDF 6658                    <1>       POP    AX                  ; RESTORE NUMBER TRANSFERRED
12461 00003DE1 8A25[A8520100]          <1>       MOV    AH, [DSKETTE_STATUS]      ; GET STATUS OF OPERATION
```

```
12462 00003DE7 08E4                <1>       OR     AH,AH               ; CHECK FOR ERROR
12463 00003DE9 7402                <1>       JZ     short NUN_ERR       ; NO ERROR
12464 00003DEB 30C0                <1>       XOR    AL,AL               ; CLEAR NUMBER RETURNED
12465                              <1> NUN_ERR:
12466 00003DED 80FC01              <1>       CMP    AH,1                ; SET THE CARRY FLAG TO INDICATE
12467 00003DF0 F5                  <1>       CMC                        ; SUCCESS OR FAILURE
12468 00003DF1 C3                  <1>       RETn
12469                              <1>
12470                              <1> ;--------------------------------------------------------------------------
12471                              <1> ; SETUP_DBL
12472                              <1> ;     CHECK DOUBLE STEP.
12473                              <1> ;
12474                              <1> ; ON ENTRY : DI = DRIVE
12475                              <1> ;
12476                              <1> ; ON EXIT :  CY = 1 MEANS ERROR
12477                              <1> ;--------------------------------------------------------------------------
12478                              <1> SETUP_DBL:
12479 00003DF2 8AA7[B5520100]      <1>       MOV    AH, [DSK_STATE+eDI] ; ACCESS STATE
12480 00003DF8 F6C410              <1>       TEST   AH,MED_DET          ; ESTABLISHED STATE ?
12481 00003DFB 757E                <1>       JNZ    short NO_DBL              ; IF ESTABLISHED THEN DOUBLE DONE
12482                              <1>
12483                              <1> ;-----       CHECK FOR TRACK 0 TO SPEED UP ACKNOWLEDGE OF UNFORMATTED DISKETTE
12484                              <1>
12485 00003DFD C605[A5520100]00    <1>       MOV    byte [SEEK_STATUS],0       ; SET RECALIBRATE REQUIRED ON ALL DRIVES
12486 00003E04 E8E0000000          <1>       CALL   MOTOR_ON           ; ENSURE MOTOR STAY ON
12487 00003E09 B500                <1>       MOV    CH,0               ; LOAD TRACK 0
12488 00003E0B E8D4010000          <1>       CALL   SEEK               ; SEEK TO TRACK 0
12489 00003E10 E868000000          <1>       CALL   READ_ID                 ; READ ID FUNCTION
12490 00003E15 7249                <1>       JC     short SD_ERR       ; IF ERROR NO TRACK 0
12491                              <1>
12492                              <1> ;-----       INITIALIZE START AND MAX TRACKS (TIMES 2 FOR BOTH HEADS)
12493                              <1>
12494 00003E17 66B95004            <1>       MOV    CX,0450H           ; START, MAX TRACKS
12495 00003E1B F687[B5520100]01    <1>       TEST   byte [DSK_STATE+eDI],TRK_CAPA ; TEST FOR 80 TRACK CAPABILITY
12496 00003E22 7402                <1>       JZ     short CNT_OK       ; IF NOT COUNT IS SETUP
12497 00003E24 B1A0                <1>       MOV    CL,0A0H                 ; MAXIMUM TRACK 1.2 MB
12498                              <1>
12499                              <1> ;     ATTEMPT READ ID OF ALL TRACKS, ALL HEADS UNTIL SUCCESS; UPON SUCCESS,
12500                              <1> ;     MUST SEE IF ASKED FOR TRACK IN SINGLE STEP MODE = TRACK ID READ; IF NOT
12501                              <1> ;     THEN SET DOUBLE STEP ON.
12502                              <1>
12503                              <1> CNT_OK:
12504 00003E26 C605[A7520100]FF    <1>         MOV     byte [MOTOR_COUNT], 0FFH ; ENSURE MOTOR STAYS ON FOR OPERATION
12505 00003E2D 6651                <1>       PUSH   CX                  ; SAVE TRACK, COUNT
12506 00003E2F C605[A8520100]00    <1>       MOV    byte [DSKETTE_STATUS],0   ; CLEAR STATUS, EXPECT ERRORS
12507 00003E36 6631C0              <1>       XOR    AX,AX               ; CLEAR AX
12508 00003E39 D0ED                <1>       SHR    CH,1                ; HALVE TRACK, CY = HEAD
12509 00003E3B C0D003              <1>       RCL    AL,3                ; AX = HEAD IN CORRECT BIT
12510 00003E3E 6650                <1>       PUSH   AX                  ; SAVE HEAD
12511 00003E40 E89F010000          <1>       CALL   SEEK                ; SEEK TO TRACK
12512 00003E45 6658                <1>       POP    AX                  ; RESTORE HEAD
12513 00003E47 6609C7              <1>       OR     DI,AX               ; DI = HEAD OR'ED DRIVE
12514 00003E4A E82E000000          <1>       CALL   READ_ID                 ; READ ID HEAD 0
12515 00003E4F 9C                  <1>       PUSHF                      ; SAVE RETURN FROM READ_ID
12516 00003E50 6681E7FB00          <1>       AND    DI,11111011B        ; TURN OFF HEAD 1 BIT
12517 00003E55 9D                  <1>       POPF                       ; RESTORE ERROR RETURN
12518 00003E56 6659                <1>       POP    CX                  ; RESTORE COUNT
12519 00003E58 7308                <1>       JNC    short DO_CHK        ; IF OK, ASKED = RETURNED TRACK ?
12520 00003E5A FEC5                <1>       INC    CH                  ; INC FOR NEXT TRACK
12521 00003E5C 38CD                <1>       CMP    CH,CL               ; REACHED MAXIMUM YET
12522 00003E5E 75C6                <1>       JNZ    short CNT_OK        ; CONTINUE TILL ALL TRIED
12523                              <1>
12524                              <1> ;-----       FALL THRU, READ ID FAILED FOR ALL TRACKS
12525                              <1>
12526                              <1> SD_ERR:
12527 00003E60 F9                  <1>       STC                        ; SET CARRY FOR ERROR
12528 00003E61 C3                  <1>       RETn                       ; SETUP_DBL ERROR EXIT
12529                              <1>
12530                              <1> DO_CHK:
12531 00003E62 8A0D[AC520100]      <1>       MOV    CL, [NEC_STATUS+3]  ; LOAD RETURNED TRACK
12532 00003E68 888F[B9520100]      <1>       MOV    [DSK_TRK+eDI], CL   ; STORE TRACK NUMBER
12533 00003E6E D0ED                <1>       SHR    CH,1                ; HALVE TRACK
12534 00003E70 38CD                <1>       CMP    CH,CL               ; IS IT THE SAME AS ASKED FOR TRACK
12535 00003E72 7407                <1>       JZ     short NO_DBL        ; IF SAME THEN NO DOUBLE STEP
12536 00003E74 808F[B5520100]20    <1>       OR     byte [DSK_STATE+eDI],DBL_STEP ; TURN ON DOUBLE STEP REQUIRED
12537                              <1> NO_DBL:
12538 00003E7B F8                  <1>       CLC                        ; CLEAR ERROR FLAG
12539 00003E7C C3                  <1>       RETn
12540                              <1>
12541                              <1> ;--------------------------------------------------------------------------
12542                              <1> ; READ_ID
12543                              <1> ;     READ ID FUNCTION.
12544                              <1> ;
12545                              <1> ; ON ENTRY:  DI : BIT 2 = HEAD; BITS 1,0 = DRIVE
12546                              <1> ;
12547                              <1> ; ON EXIT:   DI : BIT 2 IS RESET, BITS 1,0 = DRIVE
12548                              <1> ;            @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION
12549                              <1> ;--------------------------------------------------------------------------
12550                              <1> READ_ID:
12551 00003E7D B8[9A3E0000]        <1>       MOV    eAX, ER_3           ; MOVE NEC OUTPUT ERROR ADDRESS
12552 00003E82 50                  <1>       PUSH   eAX
12553 00003E83 B44A                <1>       MOV    AH,4AH              ; READ ID COMMAND
12554 00003E85 E820010000          <1>       CALL   NEC_OUTPUT          ; TO CONTROLLER
12555 00003E8A 6689F8              <1>       MOV    AX,DI               ; DRIVE # TO AH, HEAD 0
12556 00003E8D 88C4                <1>       MOV    AH,AL
12557 00003E8F E816010000          <1>       CALL   NEC_OUTPUT          ; TO CONTROLLER
12558 00003E94 E80BFEFFFF          <1>       CALL   NEC_TERM            ; WAIT FOR OPERATION, GET STATUS
12559 00003E99 58                  <1>       POP    eAX                 ; THROW AWAY ERROR ADDRESS
12560                              <1> ER_3:
12561 00003E9A C3                  <1>       RETn
12562                              <1>
12563                              <1> ;--------------------------------------------------------------------------
12564                              <1> ; CMOS_TYPE
```

```
12565                                  <1> ;     RETURNS DISKETTE TYPE FROM CMOS
12566                                  <1> ;
12567                                  <1> ; ON ENTRY:  DI = DRIVE #
12568                                  <1> ;
12569                                  <1> ; ON EXIT:   AL = TYPE; CY REFLECTS STATUS
12570                                  <1> ;--------------------------------------------------------------------------
12571                                  <1>
12572                                  <1> CMOS_TYPE: ; 11/12/2014
12573 00003E9B 8A87[F65C0000]          <1> mov   al, [eDI+fd0_type]
12574 00003EA1 20C0                    <1> and   al, al ; 18/12/2014
12575 00003EA3 C3                      <1> retn
12576                                  <1>
12577                                  <1> ;CMOS_TYPE:
12578                                  <1> ;     MOV    AL, CMOS_DIAG       ; CMOS DIAGNOSTIC STATUS BYTE ADDRESS
12579                                  <1> ;     CALL   CMOS_READ           ; GET CMOS STATUS
12580                                  <1> ;     TEST   AL,BAD_BAT+BAD_CKSUM      ; BATTERY GOOD AND CHECKSUM VALID
12581                                  <1> ;     STC                        ; SET CY = 1 INDICATING ERROR FOR RETURN
12582                                  <1> ;     JNZ    short BAD_CM        ; ERROR IF EITHER BIT ON
12583                                  <1> ;     MOV    AL,CMOS_DISKETTE    ; ADDRESS OF DISKETTE BYTE IN CMOS
12584                                  <1> ;     CALL   CMOS_READ           ; GET DISKETTE BYTE
12585                                  <1> ;     OR     DI,DI               ; SEE WHICH DRIVE IN QUESTION
12586                                  <1> ;     JNZ    short TB            ; IF DRIVE 1, DATA IN LOW NIBBLE
12587                                  <1> ;     ROR    AL,4                ; EXCHANGE NIBBLES IF SECOND DRIVE
12588                                  <1> ;TB:
12589                                  <1> ;     AND    AL,0FH              ; KEEP ONLY DRIVE DATA, RESET CY, 0
12590                                  <1> ;BAD_CM:
12591                                  <1> ;     RETn                       ; CY, STATUS OF READ
12592                                  <1>
12593                                  <1> ;--------------------------------------------------------------------------
12594                                  <1> ; GET_PARM
12595                                  <1> ;     THIS ROUTINE FETCHES THE INDEXED POINTER FROM THE DISK_BASE
12596                                  <1> ;     BLOCK POINTED TO BY THE DATA VARIABLE @DISK_POINTER. A BYTE FROM
12597                                  <1> ;     THAT TABLE IS THEN MOVED INTO AH, THE INDEX OF THAT BYTE BEING
12598                                  <1> ;     THE PARAMETER IN DL.
12599                                  <1> ;
12600                                  <1> ; ON ENTRY:  DL = INDEX OF BYTE TO BE FETCHED
12601                                  <1> ;
12602                                  <1> ; ON EXIT:   AH = THAT BYTE FROM BLOCK
12603                                  <1> ;            AL,DH DESTROYED
12604                                  <1> ;--------------------------------------------------------------------------
12605                                  <1> GET_PARM:
12606                                  <1>       ;PUSH  DS
12607 00003EA4 56                      <1>       PUSH   eSI
12608                                  <1>       ;SUB   AX,AX               ; DS = 0, BIOS DATA AREA
12609                                  <1>       ;MOV   DS,AX
12610                                  <1>       ;;mov  ax, cs
12611                                  <1>       ;;mov  ds, ax
12612                                  <1>       ; 08/02/2015 (protected mode modifications, bx -> ebx)
12613 00003EA5 87D3                    <1>       XCHG   eDX,eBX                   ; BL = INDEX
12614                                  <1>       ;SUB   BH,BH               ; BX = INDEX
12615 00003EA7 81E3FF000000            <1>       and    ebx, 0FFh
12616                                  <1>       ;LDS   SI, [DISK_POINTER]  ; POINT TO BLOCK
12617                                  <1>       ;
12618                                  <1>       ; 17/12/2014
12619 00003EAD 66A1[E55C0000]          <1>       mov    ax, [cfd] ; current (AL) and previous fd (AH)
12620 00003EB3 38E0                    <1>       cmp    al, ah
12621 00003EB5 7425                    <1>       je     short gpndc
12622 00003EB7 A2[E65C0000]            <1>       mov    [pfd], al ; current drive -> previous drive
12623 00003EBC 53                      <1>       push   ebx ; 08/02/2015
12624 00003EBD 88C3                    <1>       mov    bl, al
12625                                  <1>       ; 11/12/2014
12626 00003EBF 8A83[F65C0000]          <1>       mov    al, [eBX+fd0_type]  ; Drive type (0,1,2,3,4)
12627                                  <1>       ; 18/12/2014
12628 00003EC5 20C0                    <1>       and    al, al
12629 00003EC7 7507                    <1>       jnz    short gpdtc
12630 00003EC9 BB[CF5C0000]            <1>       mov    ebx, MD_TBL6        ; 1.44 MB param. tbl. (default)
12631 00003ECE EB05                    <1>         jmp     short gpdpu
12632                                  <1> gpdtc:
12633 00003ED0 E817F9FFFF              <1>       call   DR_TYPE_CHECK
12634                                  <1>       ; cf = 1 -> eBX points to 1.44MB fd parameter table (default)
12635                                  <1> gpdpu:
12636 00003ED5 891D[6C5C0000]          <1>       mov    [DISK_POINTER], ebx
12637 00003EDB 5B                      <1>       pop    ebx
12638                                  <1> gpndc:
12639 00003EDC 8B35[6C5C0000]          <1>       mov    esi, [DISK_POINTER] ; 08/02/2015, si -> esi
12640 00003EE2 8A241E                  <1>       MOV    AH, [eSI+eBX]       ; GET THE WORD
12641 00003EE5 87D3                    <1>       XCHG   eDX,eBX                   ; RESTORE BX
12642 00003EE7 5E                      <1>       POP    eSI
12643                                  <1>       ;POP   DS
12644 00003EE8 C3                      <1>       RETn
12645                                  <1>
12646                                  <1> ;--------------------------------------------------------------------------
12647                                  <1> ; MOTOR_ON
12648                                  <1> ;     TURN MOTOR ON AND WAIT FOR MOTOR START UP TIME. THE @MOTOR_COUNT
12649                                  <1> ;     IS REPLACED WITH A SUFFICIENTLY HIGH NUMBER (0FFH) TO ENSURE
12650                                  <1> ;     THAT THE MOTOR DOES NOT GO OFF DURING THE OPERATION. IF THE
12651                                  <1> ;     MOTOR NEEDED TO BE TURNED ON, THE MULTI-TASKING HOOK FUNCTION
12652                                  <1> ;     (AX=90FDH, INT 15) IS CALLED TELLING THE OPERATING SYSTEM
12653                                  <1> ;     THAT THE BIOS IS ABOUT TO WAIT FOR MOTOR START UP. IF THIS
12654                                  <1> ;     FUNCTION RETURNS WITH CY = 1, IT MEANS THAT THE MINIMUM WAIT
12655                                  <1> ;     HAS BEEN COMPLETED. AT THIS POINT A CHECK IS MADE TO ENSURE
12656                                  <1> ;     THAT THE MOTOR WASN'T TURNED OFF BY THE TIMER. IF THE HOOK DID
12657                                  <1> ;     NOT WAIT, THE WAIT FUNCTION (AH=086H) IS CALLED TO WAIT THE
12658                                  <1> ;     PRESCRIBED AMOUNT OF TIME. IF THE CARRY FLAG IS SET ON RETURN,
12659                                  <1> ;     IT MEANS THAT THE FUNCTION IS IN USE AND DID NOT PERFORM THE
12660                                  <1> ;     WAIT. A TIMER 1 WAIT LOOP WILL THEN DO THE WAIT.
12661                                  <1> ;
12662                                  <1> ; ON ENTRY:  DI = DRIVE #
12663                                  <1> ; ON EXIT:   AX,CX,DX DESTROYED
12664                                  <1> ;--------------------------------------------------------------------------
12665                                  <1> MOTOR_ON:
12666 00003EE9 53                      <1>       PUSH   eBX                 ; SAVE REG.
12667 00003EEA E82A000000              <1>       CALL   TURN_ON                   ; TURN ON MOTOR
```

```
12668 00003EEF 7226              <1>       JC     short MOT_IS_ON           ; IF CY=1 NO WAIT
12669 00003EF1 E89BF9FFFF        <1>       CALL   XLAT_OLD           ; TRANSLATE STATE TO COMPATIBLE MODE
12670 00003EF6 E865F9FFFF        <1>       CALL   XLAT_NEW           ; TRANSLATE STATE TO PRESENT ARCH,
12671                            <1>       ;CALL  TURN_ON            ; CHECK AGAIN IF MOTOR ON
12672                            <1>       ;JC    MOT_IS_ON          ; IF NO WAIT MEANS IT IS ON
12673                            <1> M_WAIT:
12674 00003EFB B20A              <1>       MOV    DL,10              ; GET THE MOTOR WAIT PARAMETER
12675 00003EFD E8A2FFFFFF        <1>       CALL   GET_PARM
12676                            <1>       ;MOV   AL,AH              ; AL = MOTOR WAIT PARAMETER
12677                            <1>       ;XOR   AH,AH              ; AX = MOTOR WAIT PARAMETER
12678                            <1>       ;CMP   AL,8               ; SEE IF AT LEAST A SECOND IS SPECIFIED
12679 00003F02 80FC08            <1>       cmp    ah, 8
12680                            <1>       ;JAE   short GP2          ; IF YES, CONTINUE
12681 00003F05 7702              <1>       ja     short J13
12682                            <1>       ;MOV   AL,8               ; ONE SECOND WAIT FOR MOTOR START UP
12683 00003F07 B408              <1>       mov    ah, 8
12684                            <1>
12685                            <1> ;-----       AS CONTAINS NUMBER OF 1/8 SECONDS (125000 MICROSECONDS) TO WAIT
12686                            <1> GP2:
12687                            <1> ;-----       FOLLOWING LOOPS REQUIRED WHEN RTC WAIT FUNCTION IS ALREADY IN USE
12688                            <1> J13:                           ; WAIT FOR 1/8 SECOND PER (AL)
12689 00003F09 B95E200000        <1>       MOV    eCX,8286           ; COUNT FOR 1/8 SECOND AT 15.085737 US
12690 00003F0E E8DADEFFFF        <1>       CALL   WAITF              ; GO TO FIXED WAIT ROUTINE
12691                            <1>       ;DEC   AL                 ; DECREMENT TIME VALUE
12692 00003F13 FECC              <1>       dec    ah
12693 00003F15 75F2              <1>       JNZ    short J13          ; ARE WE DONE YET
12694                            <1> MOT_IS_ON:
12695 00003F17 5B                <1>       POP    eBX                ; RESTORE REG.
12696 00003F18 C3                <1>       RETn
12697                            <1>
12698                            <1> ;--------------------------------------------------------------------------------
12699                            <1> ; TURN_ON
12700                            <1> ;     TURN MOTOR ON AND RETURN WAIT STATE.
12701                            <1> ;
12702                            <1> ; ON ENTRY:  DI = DRIVE #
12703                            <1> ;
12704                            <1> ; ON EXIT:   CY = 0 MEANS WAIT REQUIRED
12705                            <1> ;            CY = 1 MEANS NO WAIT REQUIRED
12706                            <1> ;            AX,BX,CX,DX DESTROYED
12707                            <1> ;--------------------------------------------------------------------------------
12708                            <1> TURN_ON:
12709 00003F19 89FB              <1>       MOV    eBX,eDI                   ; BX = DRIVE #
12710 00003F1B 88D9              <1>       MOV    CL,BL              ; CL = DRIVE #
12711 00003F1D C0C304            <1>       ROL    BL,4               ; BL = DRIVE SELECT
12712 00003F20 FA                <1>       CLI                       ; NO INTERRUPTS WHILE DETERMINING STATUS
12713 00003F21 C605[A7520100]FF  <1>       MOV    byte [MOTOR_COUNT],0FFH   ; ENSURE MOTOR STAYS ON FOR OPERATION
12714 00003F28 A0[A6520100]      <1>       MOV    AL, [MOTOR_STATUS]  ; GET DIGITAL OUTPUT REGISTER REFLECTION
12715 00003F2D 2430              <1>       AND    AL,00110000B       ; KEEP ONLY DRIVE SELECT BITS
12716 00003F2F B401              <1>       MOV    AH,1               ; MASK FOR DETERMINING MOTOR BIT
12717 00003F31 D2E4              <1>       SHL    AH,CL              ; AH = MOTOR ON, A=00000001, B=00000010
12718                            <1>
12719                            <1> ;  AL = DRIVE SELECT FROM @MOTOR_STATUS
12720                            <1> ;  BL = DRIVE SELECT DESIRED
12721                            <1> ;  AH = MOTOR ON MASK DESIRED
12722                            <1>
12723 00003F33 38D8              <1>       CMP    AL,BL              ; REQUESTED DRIVE ALREADY SELECTED ?
12724 00003F35 7508              <1>       JNZ    short TURN_IT_ON   ; IF NOT SELECTED JUMP
12725 00003F37 8425[A6520100]    <1>       TEST   AH, [MOTOR_STATUS]  ; TEST MOTOR ON BIT
12726 00003F3D 7535              <1>       JNZ    short NO_MOT_WAIT  ; JUMP IF MOTOR ON AND SELECTED
12727                            <1>
12728                            <1> TURN_IT_ON:
12729 00003F3F 08DC              <1>       OR     AH,BL              ; AH = DRIVE SELECT AND MOTOR ON
12730 00003F41 8A3D[A6520100]    <1>       MOV    BH,[MOTOR_STATUS]  ; SAVE COPY OF @MOTOR_STATUS BEFORE
12731 00003F47 80E70F            <1>       AND    BH,00001111B       ; KEEP ONLY MOTOR BITS
12732 00003F4A 8025[A6520100]CF  <1>       AND    byte [MOTOR_STATUS],11001111B ; CLEAR OUT DRIVE SELECT
12733 00003F51 0825[A6520100]    <1>       OR     [MOTOR_STATUS],AH  ; OR IN DRIVE SELECTED AND MOTOR ON
12734 00003F57 A0[A6520100]      <1>       MOV    AL,[MOTOR_STATUS]  ; GET DIGITAL OUTPUT REGISTER REFLECTION
12735 00003F5C 88C3              <1>       MOV    BL,AL              ; BL=@MOTOR_STATUS AFTER, BH=BEFORE
12736 00003F5E 80E30F            <1>       AND    BL,00001111B       ; KEEP ONLY MOTOR BITS
12737 00003F61 FB                <1>       STI                       ; ENABLE INTERRUPTS AGAIN
12738 00003F62 243F              <1>       AND    AL,00111111B       ; STRIP AWAY UNWANTED BITS
12739 00003F64 C0C004            <1>       ROL    AL,4               ; PUT BITS IN DESIRED POSITIONS
12740 00003F67 0C0C              <1>       OR     AL,00001100B       ; NO RESET, ENABLE DMA/INTERRUPT
12741 00003F69 66BAF203          <1>       MOV    DX,03F2H           ; SELECT DRIVE AND TURN ON MOTOR
12742 00003F6D EE                <1>       OUT    DX,AL
12743 00003F6E 38FB              <1>       CMP    BL,BH              ; NEW MOTOR TURNED ON ?
12744                            <1>       ;JZ    short NO_MOT_WAIT  ; NO WAIT REQUIRED IF JUST SELECT
12745 00003F70 7403              <1>       je     short no_mot_w1 ; 27/02/2015
12746 00003F72 F8                <1>       CLC                       ; (re)SET CARRY MEANING WAIT
12747 00003F73 C3                <1>       RETn
12748                            <1>
12749                            <1> NO_MOT_WAIT:
12750 00003F74 FB                <1>       sti
12751                            <1> no_mot_w1: ; 27/02/2015
12752 00003F75 F9                <1>       STC                       ; SET NO WAIT REQUIRED
12753                            <1>       ;STI                      ; INTERRUPTS BACK ON
12754 00003F76 C3                <1>       RETn
12755                            <1>
12756                            <1> ;--------------------------------------------------------------------------------
12757                            <1> ; HD_WAIT
12758                            <1> ;     WAIT FOR HEAD SETTLE TIME.
12759                            <1> ;
12760                            <1> ; ON ENTRY:  DI = DRIVE #
12761                            <1> ;
12762                            <1> ; ON EXIT:   AX,BX,CX,DX DESTROYED
12763                            <1> ;--------------------------------------------------------------------------------
12764                            <1> HD_WAIT:
12765 00003F77 B209              <1>       MOV    DL,9               ; GET HEAD SETTLE PARAMETER
12766 00003F79 E826FFFFFF        <1>       CALL   GET_PARM
12767 00003F7E 08E4              <1>       or     ah, ah ; 17/12/2014
12768 00003F80 7519              <1>       jnz    short DO_WAT
12769 00003F82 F605[A6520100]80  <1>         TEST    byte [MOTOR_STATUS],10000000B ; SEE IF A WRITE OPERATION
12770                            <1>       ;JZ    short ISNT_WRITE   ; IF NOT, DO NOT ENFORCE ANY VALUES
```

```
12771                                  <1>       ;OR    AH,AH               ; CHECK FOR ANY WAIT?
12772                                  <1>       ;JNZ   short DO_WAT        ; IF THERE DO NOT ENFORCE
12773 00003F89 741E                    <1>       jz     short HW_DONE
12774 00003F8B B40F                    <1>       MOV    AH,HD12_SETTLE            ; LOAD 1.2M HEAD SETTLE MINIMUM
12775 00003F8D 8A87[B5520100]          <1>       MOV    AL,[DSK_STATE+eDI]  ; LOAD STATE
12776 00003F93 24C0                    <1>       AND    AL,RATE_MSK         ; KEEP ONLY RATE
12777 00003F95 3C80                    <1>       CMP    AL,RATE_250         ; 1.2 M DRIVE ?
12778 00003F97 7502                    <1>       JNZ    short DO_WAT        ; DEFAULT HEAD SETTLE LOADED
12779                                  <1> ;GP3:
12780 00003F99 B414                    <1>       MOV    AH,HD320_SETTLE           ; USE 320/360 HEAD SETTLE
12781                                  <1> ;     JMP    SHORT DO_WAT
12782                                  <1>
12783                                  <1> ;ISNT_WRITE:
12784                                  <1> ;     OR     AH,AH               ; CHECK FOR NO WAIT
12785                                  <1> ;     JZ     short HW_DONE       ; IF NOT WRITE AND 0 ITS OK
12786                                  <1>
12787                                  <1> ;-----       AH CONTAINS NUMBER OF MILLISECONDS TO WAIT
12788                                  <1> DO_WAT:
12789                                  <1> ;     MOV    AL,AH               ; AL = # MILLISECONDS
12790                                  <1> ;     ;XOR   AH,AH               ; AX = # MILLISECONDS
12791                                  <1> J29:                             ;      1 MILLISECOND LOOP
12792                                  <1>       ;mov   cx, WAIT_FDU_HEAD_SETTLE ; 33 ; 1 ms in 30 micro units.
12793 00003F9B B942000000              <1>       MOV    eCX,66              ; COUNT AT 15.085737 US PER COUNT
12794 00003FA0 E848DEFFFF              <1>       CALL   WAITF               ; DELAY FOR 1 MILLISECOND
12795                                  <1>       ;DEC   AL                  ; DECREMENT THE COUNT
12796 00003FA5 FECC                    <1>       dec    ah
12797 00003FA7 75F2                    <1>       JNZ    short J29           ; DO AL MILLISECOND # OF TIMES
12798                                  <1> HW_DONE:
12799 00003FA9 C3                      <1>       RETn
12800                                  <1>
12801                                  <1> ;--------------------------------------------------------------------------
12802                                  <1> ; NEC_OUTPUT
12803                                  <1> ;     THIS ROUTINE SENDS A BYTE TO THE NEC CONTROLLER AFTER TESTING
12804                                  <1> ;     FOR CORRECT DIRECTION AND CONTROLLER READY THIS ROUTINE WILL
12805                                  <1> ;     TIME OUT IF THE BYTE IS NOT ACCEPTED WITHIN A REASONABLE AMOUNT
12806                                  <1> ;     OF TIME, SETTING THE DISKETTE STATUS ON COMPLETION.
12807                                  <1> ;
12808                                  <1> ; ON ENTRY:  AH = BYTE TO BE OUTPUT
12809                                  <1> ;
12810                                  <1> ; ON EXIT:   CY = 0  SUCCESS
12811                                  <1> ;            CY = 1  FAILURE -- DISKETTE STATUS UPDATED
12812                                  <1> ;                    IF A FAILURE HAS OCCURRED, THE RETURN IS MADE ONE LEVEL
12813                                  <1> ;                    HIGHER THAN THE CALLER OF NEC OUTPUT. THIS REMOVES THE
12814                                  <1> ;                    REQUIREMENT OF TESTING AFTER EVERY CALL OF NEC_OUTPUT.
12815                                  <1> ;            AX,CX,DX DESTROYED
12816                                  <1> ;--------------------------------------------------------------------------
12817                                  <1>
12818                                  <1> ; 09/12/2014 [Erdogan Tan]
12819                                  <1> ;     (from 'PS2 Hardware Interface Tech. Ref. May 88', Page 09-05.)
12820                                  <1> ; Diskette Drive Controller Status Register (3F4h)
12821                                  <1> ;     This read only register facilitates the transfer of data between
12822                                  <1> ;     the system microprocessor and the controller.
12823                                  <1> ; Bit 7 - When set to 1, the Data register is ready to transfer data
12824                                  <1> ;       with the system micrprocessor.
12825                                  <1> ; Bit 6 - The direction of data transfer. If this bit is set to 0,
12826                                  <1> ;       the transfer is to the controller.
12827                                  <1> ; Bit 5 - When this bit is set to 1, the controller is in the non-DMA mode.
12828                                  <1> ; Bit 4 - When this bit is set to 1, a Read or Write command is being executed.
12829                                  <1> ; Bit 3 - Reserved.
12830                                  <1> ; Bit 2 - Reserved.
12831                                  <1> ; Bit 1 - When this bit is set to 1, dskette drive 1 is in the seek mode.
12832                                  <1> ; Bit 0 - When this bit is set to 1, dskette drive 1 is in the seek mode.
12833                                  <1>
12834                                  <1> ; Data Register (3F5h)
12835                                  <1> ; This read/write register passes data, commands and parameters, and provides
12836                                  <1> ; diskette status information.
12837                                  <1>
12838                                  <1> NEC_OUTPUT:
12839                                  <1>       ;PUSH  BX                  ; SAVE REG.
12840 00003FAA 66BAF403               <1>       MOV    DX,03F4H            ; STATUS PORT
12841                                  <1>       ;MOV   BL,2                ; HIGH ORDER COUNTER
12842                                  <1>       ;XOR   CX,CX               ; COUNT FOR TIME OUT
12843                                  <1>       ; 16/12/2014
12844                                  <1>       ; waiting for (max.) 0.5 seconds
12845                                  <1>         ;;mov     byte [wait_count], 0 ;; 27/02/2015
12846                                  <1>       ;
12847                                  <1>       ; 17/12/2014
12848                                  <1>       ; Modified from AWARD BIOS 1999 - ADISK.ASM - SEND_COMMAND
12849                                  <1>       ;
12850                                  <1>       ;WAIT_FOR_PORT:    Waits for a bit at a port pointed to by DX to
12851                                  <1>       ;            go on.
12852                                  <1>       ;INPUT:
12853                                  <1>       ;     AH=Mask for isolation bits.
12854                                  <1>       ;     AL=pattern to look for.
12855                                  <1>       ;     DX=Port to test for
12856                                  <1>       ;     BH:CX=Number of memory refresh periods to delay.
12857                                  <1>       ;          (normally 30 microseconds per period.)
12858                                  <1>       ;
12859                                  <1>       ;WFP_SHORT:
12860                                  <1>       ;     Wait for port if refresh cycle is short (15-80 Us range).
12861                                  <1>       ;
12862                                  <1>
12863                                  <1> ;     mov    bl, WAIT_FDU_SEND_HI+1    ; 0+1
12864                                  <1> ;     mov    cx, WAIT_FDU_SEND_LO      ; 16667
12865 00003FAE B91B410000              <1>       mov    ecx, WAIT_FDU_SEND_LH   ; 16667 (27/02/2015)
12866                                  <1> ;
12867                                  <1> ;WFPS_OUTER_LP:
12868                                  <1> ;     ;
12869                                  <1> ;WFPS_CHECK_PORT:
12870                                  <1> J23:
12871 00003FB3 EC                     <1>       IN     AL,DX               ; GET STATUS
12872 00003FB4 24C0                    <1>       AND    AL,11000000B        ; KEEP STATUS AND DIRECTION
12873 00003FB6 3C80                    <1>       CMP    AL,10000000B        ; STATUS 1 AND DIRECTION 0 ?
```

```
12874 00003FB8 7418              <1>       JZ     short J27           ; STATUS AND DIRECTION OK
12875                            <1> WFPS_HI:
12876 00003FBA E461              <1>       IN     AL, PORT_B   ;061h  ; SYS1 ; wait for hi to lo
12877 00003FBC A810              <1>       TEST   AL,010H                   ; transition on memory
12878 00003FBE 75FA              <1>       JNZ    SHORT WFPS_HI       ; refresh.
12879                            <1> WFPS_LO:
12880 00003FC0 E461              <1>       IN     AL, PORT_B          ; SYS1
12881 00003FC2 A810              <1>       TEST   AL,010H
12882 00003FC4 74FA              <1>       JZ     SHORT WFPS_LO
12883                            <1>       ;LOOP  SHORT WFPS_CHECK_PORT
12884 00003FC6 E2EB              <1>       loop   J23    ; 27/02/2015
12885                            <1> ;     ;
12886                            <1> ;     dec    bl
12887                            <1> ;     jnz    short WFPS_OUTER_LP
12888                            <1> ;     jmp    short WFPS_TIMEOUT  ; fail
12889                            <1> ;J23:
12890                            <1> ;     IN     AL,DX               ; GET STATUS
12891                            <1> ;     AND    AL,11000000B        ; KEEP STATUS AND DIRECTION
12892                            <1> ;     CMP    AL,10000000B        ; STATUS 1 AND DIRECTION 0 ?
12893                            <1> ;     JZ     short J27           ; STATUS AND DIRECTION OK
12894                            <1>       ;LOOP  J23                 ; CONTINUE TILL CX EXHAUSTED
12895                            <1>       ;DEC   BL                  ; DECREMENT COUNTER
12896                            <1>       ;JNZ   short J23           ; REPEAT TILL DELAY FINISHED, CX = 0
12897                            <1>
12898                            <1>       ;;27/02/2015
12899                            <1>       ;16/12/2014
12900                            <1>         ;;cmp     byte [wait_count], 10   ; (10/18.2 seconds)
12901                            <1>       ;;jb   short J23
12902                            <1>
12903                            <1> ;WFPS_TIMEOUT:
12904                            <1>
12905                            <1> ;-----       FALL THRU TO ERROR RETURN
12906                            <1>
12907 00003FC8 800D[A8520100]80  <1>       OR     byte [DSKETTE_STATUS],TIME_OUT
12908                            <1>       ;POP   BX                  ; RESTORE REG.
12909 00003FCF 58                <1>       POP    eAX ; 08/02/2015    ; DISCARD THE RETURN ADDRESS
12910 00003FD0 F9                <1>       STC                        ; INDICATE ERROR TO CALLER
12911 00003FD1 C3                <1>       RETn
12912                            <1>
12913                            <1> ;-----       DIRECTION AND STATUS OK; OUTPUT BYTE
12914                            <1>
12915                            <1> J27:
12916 00003FD2 88E0              <1>       MOV    AL,AH               ; GET BYTE TO OUTPUT
12917 00003FD4 6642              <1>       INC    DX                  ; DATA PORT = STATUS PORT + 1
12918 00003FD6 EE                <1>       OUT    DX,AL               ; OUTPUT THE BYTE
12919                            <1>       ;;NEWIODELAY  ;; 27/02/2015
12920                            <1>       ; 27/02/2015
12921 00003FD7 9C                <1>       PUSHF                      ; SAVE FLAGS
12922 00003FD8 B903000000        <1>       MOV    eCX, 3              ; 30 TO 45 MICROSECONDS WAIT FOR
12923 00003FDD E80BDEFFFF        <1>       CALL   WAITF               ; NEC FLAGS UPDATE CYCLE
12924 00003FE2 9D                <1>       POPF                       ; RESTORE FLAGS FOR EXIT
12925                            <1>       ;POP   BX                  ; RESTORE REG
12926 00003FE3 C3                <1>       RETn                       ; CY = 0 FROM TEST INSTRUCTION
12927                            <1>
12928                            <1> ;--------------------------------------------------------------------------
12929                            <1> ; SEEK
12930                            <1> ;     THIS ROUTINE WILL MOVE THE HEAD ON THE NAMED DRIVE TO THE NAMED
12931                            <1> ;     TRACK. IF THE DRIVE HAS NOT BEEN ACCESSED SINCE THE DRIVE
12932                            <1> ;     RESET COMMAND WAS ISSUED, THE DRIVE WILL BE RECALIBRATED.
12933                            <1> ;
12934                            <1> ; ON ENTRY:  DI = DRIVE #
12935                            <1> ;            CH = TRACK #
12936                            <1> ;
12937                            <1> ; ON EXIT:   @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION.
12938                            <1> ;            AX,BX,CX DX DESTROYED
12939                            <1> ;--------------------------------------------------------------------------
12940                            <1> SEEK:
12941 00003FE4 89FB              <1>       MOV    eBX,eDI                  ; BX = DRIVE #
12942 00003FE6 B001              <1>       MOV    AL,1                ; ESTABLISH MASK FOR RECALIBRATE TEST
12943 00003FE8 86CB              <1>       XCHG   CL,BL               ; SET DRIVE VALULE INTO CL
12944 00003FEA D2C0              <1>       ROL    AL,CL               ; SHIFT MASK BY THE DRIVE VALUE
12945 00003FEC 86CB              <1>       XCHG   CL,BL               ; RECOVER TRACK VALUE
12946 00003FEE 8405[A5520100]    <1>       TEST   AL,[SEEK_STATUS]    ; TEST FOR RECALIBRATE REQUIRED
12947 00003FF4 7526              <1>       JNZ    short J28A          ; JUMP IF RECALIBRATE NOT REQUIRED
12948                            <1>
12949 00003FF6 0805[A5520100]    <1>       OR     [SEEK_STATUS],AL    ; TURN ON THE NO RECALIBRATE BIT IN FLAG
12950 00003FFC E862000000        <1>       CALL   RECAL               ; RECALIBRATE DRIVE
12951 00004001 730E              <1>       JNC    short AFT_RECAL          ; RECALIBRATE DONE
12952                            <1>
12953                            <1> ;-----       ISSUE RECALIBRATE FOR 80 TRACK DISKETTES
12954                            <1>
12955 00004003 C605[A8520100]00  <1>       MOV    byte [DSKETTE_STATUS],0   ; CLEAR OUT INVALID STATUS
12956 0000400A E854000000        <1>       CALL   RECAL               ; RECALIBRATE DRIVE
12957 0000400F 7251              <1>       JC     short RB            ; IF RECALIBRATE FAILS TWICE THEN ERROR
12958                            <1>
12959                            <1> AFT_RECAL:
12960 00004011 C687[B9520100]00  <1>         MOV     byte [DSK_TRK+eDI],0    ; SAVE NEW CYLINDER AS PRESENT POSITION
12961 00004018 08ED              <1>       OR     CH,CH               ; CHECK FOR SEEK TO TRACK 0
12962 0000401A 743F              <1>       JZ     short DO_WAIT       ; HEAD SETTLE, CY = 0 IF JUMP
12963                            <1>
12964                            <1> ;-----       DRIVE IS IN SYNCHRONIZATION WITH CONTROLLER, SEEK TO TRACK
12965                            <1>
12966 0000401C F687[B5520100]20  <1> J28A: TEST   byte [DSK_STATE+eDI],DBL_STEP ; CHECK FOR DOUBLE STEP REQUIRED
12967 00004023 7402              <1>       JZ     short _R7           ; SINGLE STEP REQUIRED BYPASS DOUBLE
12968 00004025 D0E5              <1>       SHL    CH,1                ; DOUBLE NUMBER OF STEP TO TAKE
12969                            <1>
12970 00004027 3AAF[B9520100]    <1> _R7:  CMP    CH, [DSK_TRK+eDI]   ; SEE IF ALREADY AT THE DESIRED TRACK
12971 0000402D 7433              <1>       JE     short RB            ; IF YES, DO NOT NEED TO SEEK
12972                            <1>
12973 0000402F BA[62400000]      <1>       MOV    eDX, NEC_ERR        ; LOAD RETURN ADDRESS
12974 00004034 52                <1>       PUSH   eDX ; (*)           ; ON STACK FOR NEC OUTPUT ERROR
12975 00004035 88AF[B9520100]    <1>       MOV    [DSK_TRK+eDI],CH    ; SAVE NEW CYLINDER AS PRESENT POSITION
12976 0000403B B40F              <1>       MOV    AH,0FH              ; SEEK COMMAND TO NEC
```

```
12977 0000403D E868FFFFFF          <1>       CALL   NEC_OUTPUT
12978 00004042 89FB                <1>       MOV    eBX,eDI                  ; BX = DRIVE #
12979 00004044 88DC                <1>       MOV    AH,BL                ; OUTPUT DRIVE NUMBER
12980 00004046 E85FFFFFFF          <1>       CALL   NEC_OUTPUT
12981 0000404B 8AA7[B9520100]      <1>       MOV    AH, [DSK_TRK+eDI]   ; GET CYLINDER NUMBER
12982 00004051 E854FFFFFF          <1>       CALL   NEC_OUTPUT
12983 00004056 E829000000          <1>       CALL   CHK_STAT_2          ; ENDING INTERRUPT AND SENSE STATUS
12984                              <1>
12985                              <1> ;-----      WAIT FOR HEAD SETTLE
12986                              <1>
12987                              <1> DO_WAIT:
12988 0000405B 9C                  <1>       PUSHF                      ; SAVE STATUS
12989 0000405C E816FFFFFF          <1>       CALL   HD_WAIT                  ; WAIT FOR HEAD SETTLE TIME
12990 00004061 9D                  <1>       POPF                       ; RESTORE STATUS
12991                              <1> RB:
12992                              <1> NEC_ERR:
12993                              <1>       ; 08/02/2015 (code trick here from original IBM PC/AT DISKETTE.ASM)
12994                              <1>       ; (*) nec_err -> retn (push edx -> pop edx) -> nec_err -> retn
12995 00004062 C3                  <1>       RETn                       ; RETURN TO CALLER
12996                              <1>
12997                              <1> ;--------------------------------------------------------------------------
12998                              <1> ; RECAL
12999                              <1> ;     RECALIBRATE DRIVE
13000                              <1> ;
13001                              <1> ; ON ENTRY:  DI = DRIVE #
13002                              <1> ;
13003                              <1> ; ON EXIT:   CY REFLECTS STATUS OF OPERATION.
13004                              <1> ;--------------------------------------------------------------------------
13005                              <1> RECAL:
13006 00004063 6651                <1>       PUSH   CX
13007 00004065 B8[81400000]        <1>       MOV    eAX, RC_BACK        ; LOAD NEC_OUTPUT ERROR
13008 0000406A 50                  <1>       PUSH   eAX
13009 0000406B B407                <1>       MOV    AH,07H              ; RECALIBRATE COMMAND
13010 0000406D E838FFFFFF          <1>       CALL   NEC_OUTPUT
13011 00004072 89FB                <1>       MOV    eBX,eDI                  ; BX = DRIVE #
13012 00004074 88DC                <1>       MOV    AH,BL
13013 00004076 E82FFFFFFF          <1>       CALL   NEC_OUTPUT          ; OUTPUT THE DRIVE NUMBER
13014 0000407B E804000000          <1>       CALL   CHK_STAT_2          ; GET THE INTERRUPT AND SENSE INT STATUS
13015 00004080 58                  <1>       POP    eAX                 ; THROW AWAY ERROR
13016                              <1> RC_BACK:
13017 00004081 6659                <1>       POP    CX
13018 00004083 C3                  <1>       RETn
13019                              <1>
13020                              <1> ;--------------------------------------------------------------------------
13021                              <1> ; CHK_STAT_2
13022                              <1> ;     THIS ROUTINE HANDLES THE INTERRUPT RECEIVED AFTER RECALIBRATE,
13023                              <1> ;     OR SEEK TO THE ADAPTER. THE INTERRUPT IS WAITED FOR, THE
13024                              <1> ;     INTERRUPT STATUS SENSED, AND THE RESULT RETURNED TO THE CALLER.
13025                              <1> ;
13026                              <1> ; ON EXIT:  @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION.
13027                              <1> ;--------------------------------------------------------------------------
13028                              <1> CHK_STAT_2:
13029 00004084 B8[AC400000]        <1>         MOV     eAX, CS_BACK            ; LOAD NEC_OUTPUT ERROR ADDRESS
13030 00004089 50                  <1>       PUSH   eAX
13031 0000408A E828000000          <1>       CALL   WAIT_INT            ; WAIT FOR THE INTERRUPT
13032 0000408F 721A                <1>       JC     short J34           ; IF ERROR, RETURN IT
13033 00004091 B408                <1>       MOV    AH,08H              ; SENSE INTERRUPT STATUS COMMAND
13034 00004093 E812FFFFFF          <1>       CALL   NEC_OUTPUT
13035 00004098 E84A000000          <1>       CALL   RESULTS                  ; READ IN THE RESULTS
13036 0000409D 720C                <1>       JC     short J34
13037 0000409F A0[A9520100]        <1>       MOV    AL,[NEC_STATUS]          ; GET THE FIRST STATUS BYTE
13038 000040A4 2460                <1>       AND    AL,01100000B        ; ISOLATE THE BITS
13039 000040A6 3C60                <1>       CMP    AL,01100000B        ; TEST FOR CORRECT VALUE
13040 000040A8 7403                <1>       JZ     short J35           ; IF ERROR, GO MARK IT
13041 000040AA F8                  <1>       CLC                        ; GOOD RETURN
13042                              <1> J34:
13043 000040AB 58                  <1>       POP    eAX                 ; THROW AWAY ERROR RETURN
13044                              <1> CS_BACK:
13045 000040AC C3                  <1>       RETn
13046                              <1> J35:
13047 000040AD 800D[A8520100]40    <1>       OR     byte [DSKETTE_STATUS], BAD_SEEK
13048 000040B4 F9                  <1>       STC                        ; ERROR RETURN CODE
13049 000040B5 EBF4                <1>       JMP    SHORT J34
13050                              <1>
13051                              <1> ;--------------------------------------------------------------------------
13052                              <1> ; WAIT_INT
13053                              <1> ;     THIS ROUTINE WAITS FOR AN INTERRUPT TO OCCUR A TIME OUT ROUTINE
13054                              <1> ;     TAKES PLACE DURING THE WAIT, SO THAT AN ERROR MAY BE RETURNED
13055                              <1> ;     IF THE DRIVE IS NOT READY.
13056                              <1> ;
13057                              <1> ; ON EXIT:  @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION.
13058                              <1> ;--------------------------------------------------------------------------
13059                              <1>
13060                              <1> ; 17/12/2014
13061                              <1> ; 2.5 seconds waiting !
13062                              <1> ;(AWARD BIOS - 1999, WAIT_FDU_INT_LOW, WAIT_FDU_INT_HI)
13063                              <1> ; amount of time to wait for completion interrupt from NEC.
13064                              <1>
13065                              <1>
13066                              <1> WAIT_INT:
13067 000040B7 FB                  <1>       STI                        ; TURN ON INTERRUPTS, JUST IN CASE
13068 000040B8 F8                  <1>       CLC                        ; CLEAR TIMEOUT INDICATOR
13069                              <1>        ;MOV  BL,10               ; CLEAR THE COUNTERS
13070                              <1>        ;XOR  CX,CX               ; FOR 2 SECOND WAIT
13071                              <1>
13072                              <1>       ; Modification from AWARD BIOS - 1999 (ATORGS.ASM, WAIT
13073                              <1>       ;
13074                              <1>       ;WAIT_FOR_MEM:
13075                              <1>       ;     Waits for a bit at a specified memory location pointed
13076                              <1>       ;     to by ES:[DI] to become set.
13077                              <1>       ;INPUT:
13078                              <1>       ;     AH=Mask to test with.
13079                              <1>       ;     ES:[DI] = memory location to watch.
```

```
13080                                   <1>       ;     BH:CX=Number of memory refresh periods to delay.
13081                                   <1>       ;          (normally 30 microseconds per period.)
13082                                   <1>
13083                                   <1>       ; waiting for (max.) 2.5 secs in 30 micro units.
13084                                   <1> ;     mov    cx, WAIT_FDU_INT_LO       ; 017798
13085                                   <1> ;;    mov    bl, WAIT_FDU_INT_HI
13086                                   <1> ;     mov    bl, WAIT_FDU_INT_HI + 1
13087                                   <1>       ; 27/02/2015
13088 000040B9 B986450100              <1>       mov    ecx, WAIT_FDU_INT_LH      ; 83334 (2.5 seconds)
13089                                   <1> WFMS_CHECK_MEM:
13090 000040BE F605[A5520100]80         <1>       test   byte [SEEK_STATUS],INT_FLAG ; TEST FOR INTERRUPT OCCURRING
13091 000040C5 7516                     <1>         jnz     short J37
13092                                   <1> WFMS_HI:
13093 000040C7 E461                     <1>       IN     AL,PORT_B  ; 061h   ; SYS1, wait for lo to hi
13094 000040C9 A810                     <1>       TEST   AL,010H                   ; transition on memory
13095 000040CB 75FA                     <1>       JNZ    SHORT WFMS_HI       ; refresh.
13096                                   <1> WFMS_LO:
13097 000040CD E461                     <1>       IN     AL,PORT_B           ;SYS1
13098 000040CF A810                     <1>       TEST   AL,010H
13099 000040D1 74FA                     <1>       JZ     SHORT WFMS_LO
13100 000040D3 E2E9                     <1>         LOOP    WFMS_CHECK_MEM
13101                                   <1> ;WFMS_OUTER_LP:
13102                                   <1> ;;    or     bl, bl              ; check outer counter
13103                                   <1> ;;    jz     short J36A          ; WFMS_TIMEOUT
13104                                   <1> ;     dec    bl
13105                                   <1> ;     jz     short J36A
13106                                   <1> ;     jmp    short WFMS_CHECK_MEM
13107                                   <1>
13108                                   <1>       ;17/12/2014
13109                                   <1>       ;16/12/2014
13110                                   <1> ;        mov     byte [wait_count], 0    ; Reset (INT 08H) counter
13111                                   <1> ;J36:
13112                                   <1> ;     TEST   byte [SEEK_STATUS],INT_FLAG ; TEST FOR INTERRUPT OCCURRING
13113                                   <1> ;     JNZ    short J37
13114                                   <1>       ;16/12/2014
13115                                   <1>       ;LOOP  J36                 ; COUNT DOWN WHILE WAITING
13116                                   <1>       ;DEC   BL                  ; SECOND LEVEL COUNTER
13117                                   <1>       ;JNZ   short J36
13118                                   <1> ;       cmp     byte [wait_count], 46   ; (46/18.2 seconds)
13119                                   <1> ;     jb     short J36
13120                                   <1>
13121                                   <1> ;WFMS_TIMEOUT:
13122                                   <1> ;J36A:
13123 000040D5 800D[A8520100]80         <1>       OR     byte [DSKETTE_STATUS], TIME_OUT ; NOTHING HAPPENED
13124 000040DC F9                       <1>       STC                        ; ERROR RETURN
13125                                   <1> J37:
13126 000040DD 9C                       <1>       PUSHF                      ; SAVE CURRENT CARRY
13127 000040DE 8025[A5520100]7F         <1>       AND    byte [SEEK_STATUS], ~INT_FLAG ; TURN OFF INTERRUPT FLAG
13128 000040E5 9D                       <1>       POPF                       ; RECOVER CARRY
13129 000040E6 C3                       <1>       RETn                       ; GOOD RETURN CODE
13130                                   <1>
13131                                   <1> ;--------------------------------------------------------------------------
13132                                   <1> ; RESULTS
13133                                   <1> ;     THIS ROUTINE WILL READ ANYTHING THAT THE NEC CONTROLLER RETURNS
13134                                   <1> ;     FOLLOWING AN INTERRUPT.
13135                                   <1> ;
13136                                   <1> ; ON EXIT:  @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION.
13137                                   <1> ;           AX,BX,CX,DX DESTROYED
13138                                   <1> ;--------------------------------------------------------------------------
13139                                   <1> RESULTS:
13140 000040E7 57                       <1>       PUSH   eDI
13141 000040E8 BF[A9520100]             <1>       MOV    eDI, NEC_STATUS           ; POINTER TO DATA AREA
13142 000040ED B307                     <1>       MOV    BL,7                ; MAX STATUS BYTES
13143 000040EF 66BAF403                 <1>       MOV    DX,03F4H            ; STATUS PORT
13144                                   <1>
13145                                   <1> ;-----      WAIT FOR REQUEST FOR MASTER
13146                                   <1>
13147                                   <1> _R10:
13148                                   <1>       ; 16/12/2014
13149                                   <1>       ; wait for (max) 0.5 seconds
13150                                   <1>       ;MOV   BH,2                ; HIGH ORDER COUNTER
13151                                   <1>       ;XOR   CX,CX               ; COUNTER
13152                                   <1>
13153                                   <1>       ;Time to wait while waiting for each byte of NEC results = .5
13154                                   <1>       ;seconds.  .5 seconds = 500,000 micros.  500,000/30 = 16,667.
13155                                   <1>       ; 27/02/2015
13156 000040F3 B91B410000              <1>       mov    ecx, WAIT_FDU_RESULTS_LH ; 16667
13157                                   <1>       ;mov   cx, WAIT_FDU_RESULTS_LO  ; 16667
13158                                   <1>       ;mov   bh, WAIT_FDU_RESULTS_HI+1 ; 0+1
13159                                   <1>
13160                                   <1> WFPSR_OUTER_LP:
13161                                   <1>       ;
13162                                   <1> WFPSR_CHECK_PORT:
13163                                   <1> J39:                             ; WAIT FOR MASTER
13164 000040F8 EC                       <1>       IN     AL,DX               ; GET STATUS
13165 000040F9 24C0                     <1>       AND    AL,11000000B        ; KEEP ONLY STATUS AND DIRECTION
13166 000040FB 3CC0                     <1>       CMP    AL,11000000B        ; STATUS 1 AND DIRECTION 1 ?
13167 000040FD 7418                     <1>       JZ     short J42           ; STATUS AND DIRECTION OK
13168                                   <1> WFPSR_HI:
13169 000040FF E461                     <1>       IN     AL, PORT_B   ;061h  ; SYS1 ; wait for hi to lo
13170 00004101 A810                     <1>       TEST   AL,010H                   ; transition on memory
13171 00004103 75FA                     <1>       JNZ    SHORT WFPSR_HI            ; refresh.
13172                                   <1> WFPSR_LO:
13173 00004105 E461                     <1>       IN     AL, PORT_B          ; SYS1
13174 00004107 A810                     <1>       TEST   AL,010H
13175 00004109 74FA                     <1>       JZ     SHORT WFPSR_LO
13176 0000410B E2EB                     <1>         LOOP    WFPSR_CHECK_PORT
13177                                   <1>       ;; 27/02/2015
13178                                   <1>       ;;dec  bh
13179                                   <1>       ;;jnz  short WFPSR_OUTER_LP
13180                                   <1>       ;jmp   short WFPSR_TIMEOUT ; fail
13181                                   <1>
13182                                   <1>       ;;mov  byte [wait_count], 0
```

```
13183                                  <1> ;J39:                           ; WAIT FOR MASTER
13184                                  <1> ;     IN    AL,DX               ; GET STATUS
13185                                  <1> ;     AND   AL,11000000B        ; KEEP ONLY STATUS AND DIRECTION
13186                                  <1> ;     CMP   AL,11000000B        ; STATUS 1 AND DIRECTION 1 ?
13187                                  <1> ;     JZ    short J42           ; STATUS AND DIRECTION OK
13188                                  <1>       ;LOOP J39                 ; LOOP TILL TIMEOUT
13189                                  <1>       ;DEC  BH                  ; DECREMENT HIGH ORDER COUNTER
13190                                  <1>       ;JNZ  short J39           ; REPEAT TILL DELAY DONE
13191                                  <1>       ;
13192                                  <1>       ;;cmp byte [wait_count], 10  ; (10/18.2 seconds)
13193                                  <1>       ;;jb  short J39
13194                                  <1>
13195                                  <1> ;WFPSR_TIMEOUT:
13196 0000410D 800D[A8520100]80       <1>       OR    byte [DSKETTE_STATUS],TIME_OUT
13197 00004114 F9                     <1>       STC                         ; SET ERROR RETURN
13198 00004115 EB29                   <1>       JMP   SHORT POPRES        ; POP REGISTERS AND RETURN
13199                                  <1>
13200                                  <1> ;-----      READ IN THE STATUS
13201                                  <1>
13202                                  <1> J42:
13203 00004117 EB00                   <1>       JMP   $+2                 ; I/O DELAY
13204 00004119 6642                   <1>       INC   DX                  ; POINT AT DATA PORT
13205 0000411B EC                     <1>       IN    AL,DX               ; GET THE DATA
13206                                  <1>       ; 16/12/2014
13207                                  <1>       NEWIODELAY
13208 0000411C E6EB                   <2>  out 0ebh,al
13209 0000411E 8807                   <1>         MOV     [eDI],AL              ; STORE THE BYTE
13210 00004120 47                     <1>       INC   eDI                 ; INCREMENT THE POINTER
13211                                  <1>       ; 16/12/2014
13212                                  <1> ;     push  cx
13213                                  <1> ;     mov   cx, 30
13214                                  <1> ;wdw2:
13215                                  <1> ;     NEWIODELAY
13216                                  <1> ;     loop  wdw2
13217                                  <1> ;     pop   cx
13218                                  <1>
13219 00004121 B903000000             <1>       MOV   eCX,3               ; MINIMUM 24 MICROSECONDS FOR NEC
13220 00004126 E8C2DCFFFF             <1>       CALL  WAITF               ; WAIT 30 TO 45 MICROSECONDS
13221 0000412B 664A                   <1>       DEC   DX                  ; POINT AT STATUS PORT
13222 0000412D EC                     <1>       IN    AL,DX               ; GET STATUS
13223                                  <1>       ; 16/12/2014
13224                                  <1>       NEWIODELAY
13225 0000412E E6EB                   <2>  out 0ebh,al
13226                                  <1>       ;
13227 00004130 A810                   <1>       TEST  AL,00010000B        ; TEST FOR NEC STILL BUSY
13228 00004132 740C                   <1>       JZ    short POPRES        ; RESULTS DONE ?
13229                                  <1>
13230 00004134 FECB                   <1>       DEC   BL                  ; DECREMENT THE STATUS COUNTER
13231 00004136 75BB                   <1>         JNZ     short _R10              ; GO BACK FOR MORE
13232 00004138 800D[A8520100]20       <1>       OR    byte [DSKETTE_STATUS],BAD_NEC ; TOO MANY STATUS BYTES
13233 0000413F F9                     <1>       STC                         ; SET ERROR FLAG
13234                                  <1>
13235                                  <1> ;-----      RESULT OPERATION IS DONE
13236                                  <1> POPRES:
13237 00004140 5F                     <1>       POP   eDI
13238 00004141 C3                     <1>       RETn                        ; RETURN WITH CARRY SET
13239                                  <1>
13240                                  <1> ;--------------------------------------------------------------------------
13241                                  <1> ; READ_DSKCHNG
13242                                  <1> ;     READS THE STATE OF THE DISK CHANGE LINE.
13243                                  <1> ;
13244                                  <1> ; ON ENTRY:  DI = DRIVE #
13245                                  <1> ;
13246                                  <1> ; ON EXIT:   DI = DRIVE #
13247                                  <1> ;            ZF = 0 : DISK CHANGE LINE INACTIVE
13248                                  <1> ;            ZF = 1 : DISK CHANGE LINE ACTIVE
13249                                  <1> ;            AX,CX,DX DESTROYED
13250                                  <1> ;--------------------------------------------------------------------------
13251                                  <1> READ_DSKCHNG:
13252 00004142 E8A2FDFFFF             <1>       CALL  MOTOR_ON            ; TURN ON THE MOTOR IF OFF
13253 00004147 66BAF703               <1>       MOV   DX,03F7H            ; ADDRESS DIGITAL INPUT REGISTER
13254 0000414B EC                     <1>       IN    AL,DX               ; INPUT DIGITAL INPUT REGISTER
13255 0000414C A880                   <1>       TEST  AL,DSK_CHG          ; CHECK FOR DISK CHANGE LINE ACTIVE
13256 0000414E C3                     <1>       RETn                        ; RETURN TO CALLER WITH ZERO FLAG SET
13257                                  <1>
13258                                  <1> ;--------------------------------------------------------------------------
13259                                  <1> ; DRIVE_DET
13260                                  <1> ;     DETERMINES WHETHER DRIVE IS 80 OR 40 TRACKS AND
13261                                  <1> ;     UPDATES STATE INFORMATION ACCORDINGLY.
13262                                  <1> ; ON ENTRY:  DI = DRIVE #
13263                                  <1> ;--------------------------------------------------------------------------
13264                                  <1> DRIVE_DET:
13265 0000414F E895FDFFFF             <1>       CALL  MOTOR_ON            ; TURN ON MOTOR IF NOT ALREADY ON
13266 00004154 E80AFFFFFF             <1>       CALL  RECAL               ; RECALIBRATE DRIVE
13267 00004159 7251                   <1>       JC    short DD_BAC        ; ASSUME NO DRIVE PRESENT
13268 0000415B B530                   <1>       MOV   CH,TRK_SLAP         ; SEEK TO TRACK 48
13269 0000415D E882FEFFFF             <1>       CALL  SEEK
13270 00004162 7248                   <1>       JC    short DD_BAC        ; ERROR NO DRIVE
13271 00004164 B50B                   <1>       MOV   CH,QUIET_SEEK+1           ; SEEK TO TRACK 10
13272                                  <1> SK_GIN:
13273 00004166 FECD                   <1>       DEC   CH                  ; DECREMENT TO NEXT TRACK
13274 00004168 6651                   <1>       PUSH  CX                  ; SAVE TRACK
13275 0000416A E875FEFFFF             <1>       CALL  SEEK
13276 0000416F 723C                   <1>       JC    short POP_BAC       ; POP AND RETURN
13277 00004171 B8[AD410000]           <1>       MOV   eAX, POP_BAC        ; LOAD NEC OUTPUT ERROR ADDRESS
13278 00004176 50                     <1>       PUSH  eAX
13279 00004177 B404                   <1>       MOV   AH,SENSE_DRV_ST           ; SENSE DRIVE STATUS COMMAND BYTE
13280 00004179 E82CFEFFFF             <1>       CALL  NEC_OUTPUT          ; OUTPUT TO NEC
13281 0000417E 6689F8                 <1>       MOV   AX,DI               ; AL = DRIVE
13282 00004181 88C4                   <1>       MOV   AH,AL               ; AH = DRIVE
13283 00004183 E822FEFFFF             <1>       CALL  NEC_OUTPUT          ; OUTPUT TO NEC
13284 00004188 E85AFFFFFF             <1>       CALL  RESULTS                   ; GO GET STATUS
13285 0000418D 58                     <1>       POP   eAX                 ; THROW AWAY ERROR ADDRESS
```

```
13286 0000418E 6659                <1>       POP    CX                  ; RESTORE TRACK
13287 00004190 F605[A9520100]10    <1>       TEST   byte [NEC_STATUS], HOME   ; TRACK 0 ?
13288 00004197 74CD                <1>       JZ     short SK_GIN         ; GO TILL TRACK 0
13289 00004199 08ED                <1>       OR     CH,CH                ; IS HOME AT TRACK 0
13290 0000419B 7408                <1>       JZ     short IS_80          ; MUST BE 80 TRACK DRIVE
13291                              <1>
13292                              <1> ;     DRIVE IS A 360; SET DRIVE TO DETERMINED;
13293                              <1> ;     SET MEDIA TO DETERMINED AT RATE 250.
13294                              <1>
13295 0000419D 808F[B5520100]94    <1>       OR     byte [DSK_STATE+eDI], DRV_DET+MED_DET+RATE_250
13296 000041A4 C3                  <1>       RETn                        ; ALL INFORMATION SET
13297                              <1> IS_80:
13298 000041A5 808F[B5520100]01    <1>       OR     byte [DSK_STATE+eDI], TRK_CAPA ; SETUP 80 TRACK CAPABILITY
13299                              <1> DD_BAC:
13300 000041AC C3                  <1>       RETn
13301                              <1> POP_BAC:
13302 000041AD 6659                <1>       POP    CX                  ; THROW AWAY
13303 000041AF C3                  <1>       RETn
13304                              <1>
13305                              <1> fdc_int:
13306                              <1>         ; 30/07/2015
13307                              <1>         ; 16/02/2015
13308                              <1> ;int_0Eh: ; 11/12/2014
13309                              <1>
13310                              <1> ;--- HARDWARE INT 0EH -- ( IRQ LEVEL  6 ) -------------------------------------
13311                              <1> ; DISK_INT
13312                              <1> ;     THIS ROUTINE HANDLES THE DISKETTE INTERRUPT.
13313                              <1> ;
13314                              <1> ; ON EXIT:  THE INTERRUPT FLAG IS SET IN @SEEK_STATUS.
13315                              <1> ;--------------------------------------------------------------------------------
13316                              <1> DISK_INT_1:
13317                              <1>
13318 000041B0 6650                <1>       PUSH   AX                  ; SAVE WORK REGISTER
13319 000041B2 1E                  <1>       push   ds
13320 000041B3 66B81000            <1>       mov    ax, KDATA
13321 000041B7 8ED8                <1>       mov    ds, ax
13322 000041B9 800D[A5520100]80    <1>         OR      byte [SEEK_STATUS], INT_FLAG ; TURN ON INTERRUPT OCCURRED
13323 000041C0 B020                <1>       MOV     AL,EOI                  ; END OF INTERRUPT MARKER
13324 000041C2 E620                <1>       OUT    INTA00,AL           ; INTERRUPT CONTROL PORT
13325 000041C4 1F                  <1>       pop    ds
13326 000041C5 6658                <1>       POP    AX                  ; RECOVER REGISTER
13327 000041C7 CF                  <1>       IRETd                      ; RETURN FROM INTERRUPT
13328                              <1>
13329                              <1> ;--------------------------------------------------------------------------------
13330                              <1> ; DSKETTE_SETUP
13331                              <1> ;     THIS ROUTINE DOES A PRELIMINARY CHECK TO SEE WHAT TYPE OF
13332                              <1> ;     DISKETTE DRIVES ARE ATTACH TO THE SYSTEM.
13333                              <1> ;--------------------------------------------------------------------------------
13334                              <1>
13335                              <1> ; 29/05/2016 - TRDOS 386 (TRDOS v2.0)
13336                              <1>
13337                              <1> DSKETTE_SETUP:
13338                              <1>       ;PUSH  AX                  ; SAVE REGISTERS
13339                              <1>       ;PUSH  BX
13340                              <1>       ;PUSH  CX
13341 000041C8 52                  <1>       PUSH   eDX
13342                              <1>       ;PUSH  DI
13343                              <1>       ;;PUSH DS
13344                              <1>       ; 14/12/2014
13345                              <1>       ;mov   word [DISK_POINTER], MD_TBL6
13346                              <1>       ;mov   [DISK_POINTER+2], cs
13347                              <1>       ;
13348                              <1>       ;OR    byte [RTC_WAIT_FLAG], 1   ; NO RTC WAIT, FORCE USE OF LOOP
13349 000041C9 31FF                <1>       XOR    eDI,eDI                   ; INITIALIZE DRIVE POINTER
13350 000041CB 66C705[B5520100]00- <1>       MOV    WORD [DSK_STATE],0  ; INITIALIZE STATES
13351 000041D3 00                  <1>
13352 000041D4 8025[B0520100]33    <1>       AND    byte [LASTRATE],~(STRT_MSK+SEND_MSK) ; CLEAR START & SEND
13353 000041DB 800D[B0520100]C0    <1>       OR     byte [LASTRATE],SEND_MSK ; INITIALIZE SENT TO IMPOSSIBLE
13354 000041E2 C605[A5520100]00    <1>       MOV    byte [SEEK_STATUS],0      ; INDICATE RECALIBRATE NEEDED
13355 000041E9 C605[A7520100]00    <1>       MOV    byte [MOTOR_COUNT],0      ; INITIALIZE MOTOR COUNT
13356 000041F0 C605[A6520100]00    <1>       MOV    byte [MOTOR_STATUS],0     ; INITIALIZE DRIVES TO OFF STATE
13357 000041F7 C605[A8520100]00    <1>       MOV    byte [DSKETTE_STATUS],0   ; NO ERRORS
13358                              <1>       ;
13359                              <1>       ; 28/02/2015
13360                              <1>       ;mov   word [cfd], 100h
13361 000041FE E848F2FFFF          <1>       call   DSK_RESET
13362 00004203 5A                  <1>       pop    edx
13363 00004204 F8                  <1>       clc    ; 29/05/2016
13364 00004205 C3                  <1>       retn
13365                              <1>
13366                              <1> ;SUP0:
13367                              <1> ;     CALL   DRIVE_DET          ; DETERMINE DRIVE
13368                              <1> ;     CALL   XLAT_OLD           ; TRANSLATE STATE TO COMPATIBLE MODE
13369                              <1> ;     ; 02/01/2015
13370                              <1> ;     ;INC   DI                 ; POINT TO NEXT DRIVE
13371                              <1> ;     ;CMP   DI,MAX_DRV         ; SEE IF DONE
13372                              <1> ;     ;JNZ   short SUP0         ; REPEAT FOR EACH ORIVE
13373                              <1> ;       cmp     byte [fd1_type], 0
13374                              <1> ;     jna    short sup1
13375                              <1> ;     or     di, di
13376                              <1> ;     jnz    short sup1
13377                              <1> ;     inc    di
13378                              <1> ;       jmp     short SUP0
13379                              <1> ;sup1:
13380                              <1> ;     MOV    byte [SEEK_STATUS],0      ; FORCE RECALIBRATE
13381                              <1> ;     ;AND   byte [RTC_WAIT_FLAG],0FEH ; ALLOW FOR RTC WAIT
13382                              <1> ;     CALL   SETUP_END          ; VARIOUS CLEANUPS
13383                              <1> ;     ;;POP  DS                 ; RESTORE CALLERS REGISTERS
13384                              <1> ;     ;POP   DI
13385                              <1> ;     POP    eDX
13386                              <1> ;     ;POP   CX
13387                              <1> ;     ;POP   BX
13388                              <1> ;     ;POP   AX
```

```
13389                                   <1> ;     RETn
13390                                   <1>
13391                                   <1> ;////////////////////////////////////////////////
13392                                   <1> ;; END OF DISKETTE I/O ;;;;;;;;;;;;;;;;;;;;;;;;;;;;
13393                                   <1> ;
13394                                   <1>
13395                                   <1> int13h: ; 21/02/2015
13396 00004206 9C                       <1>       pushfd
13397 00004207 0E                       <1>       push   cs
13398 00004208 E843010000               <1>       call   DISK_IO
13399 0000420D C3                       <1>       retn
13400                                   <1>
13401                                   <1> ;;;;;; DISK I/O ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;; 21/02/2015 ;;;
13402                                   <1> ;////////////////////////////////////////////////////////////////////
13403                                   <1>
13404                                   <1> ; DISK I/O - Erdogan Tan (Retro UNIX 386 v1 project)
13405                                   <1> ; 18/02/2016
13406                                   <1> ; 17/02/2016
13407                                   <1> ; 23/02/2015
13408                                   <1> ; 21/02/2015 (unix386.s)
13409                                   <1> ; 22/12/2014 - 14/02/2015 (dsectrm2.s)
13410                                   <1> ;
13411                                   <1> ; Original Source Code:
13412                                   <1> ; DISK ----- 09/25/85 FIXED DISK BIOS
13413                                   <1> ; (IBM PC XT Model 286 System BIOS Source Code, 04-21-86)
13414                                   <1> ;
13415                                   <1> ; Modifications: by reference of AWARD BIOS 1999 (D1A0622)
13416                                   <1> ;            Source Code - ATORGS.ASM, AHDSK.ASM
13417                                   <1> ;
13418                                   <1>
13419                                   <1>
13420                                   <1> ;The wait for controller to be not busy is 10 seconds.
13421                                   <1> ;10,000,000 / 30 = 333,333.  333,333 decimal = 051615h
13422                                   <1> ;;WAIT_HDU_CTLR_BUSY_LO   equ    1615h
13423                                   <1> ;;WAIT_HDU_CTLR_BUSY_HI   equ      05h
13424                                   <1> WAIT_HDU_CTRL_BUSY_LH     equ    51615h  ;21/02/2015
13425                                   <1>
13426                                   <1> ;The wait for controller to issue completion interrupt is 10 seconds.
13427                                   <1> ;10,000,000 / 30 = 333,333.  333,333 decimal = 051615h
13428                                   <1> ;;WAIT_HDU_INT_LO  equ    1615h
13429                                   <1> ;;WAIT_HDU_INT_HI  equ      05h
13430                                   <1> WAIT_HDU_INT_LH           equ    51615h ; 21/02/2015
13431                                   <1>
13432                                   <1> ;The wait for Data request on read and write longs is
13433                                   <1> ;2000 us. (?)
13434                                   <1> ;;WAIT_HDU_DRQ_LO  equ    1000   ; 03E8h
13435                                   <1> ;;WAIT_HDU_DRQ_HI  equ    0
13436                                   <1> WAIT_HDU_DRQ_LH           equ    1000   ; 21/02/2015
13437                                   <1>
13438                                   <1> ; Port 61h (PORT_B)
13439                                   <1> SYS1        equ    61h    ; PORT_B  (diskette.inc)
13440                                   <1>
13441                                   <1> ; 23/12/2014
13442                                   <1> %define CMD_BLOCK       eBP-8  ; 21/02/2015
13443                                   <1>
13444                                   <1>
13445                                   <1> ;--- INT 13H ----------------------------------------------------------------
13446                                   <1> ;                                                                       :
13447                                   <1> ; FIXED DISK I/O INTERFACE                                                :
13448                                   <1> ;                                                                       :
13449                                   <1> ;     THIS INTERFACE PROVIDES ACCESS TO 5 1/4" FIXED DISKS THROUGH           :
13450                                   <1> ;     THE IBM FIXED DISK CONTROLLER.                                    :
13451                                   <1> ;                                                                       :
13452                                   <1> ;     THE  BIOS  ROUTINES  ARE  MEANT  TO  BE  ACCESSED  THROUGH        :
13453                                   <1> ;     SOFTWARE  INTERRUPTS  ONLY.    ANY  ADDRESSES  PRESENT     IN          :
13454                                   <1> ;     THESE  LISTINGS  ARE  INCLUDED    ONLY  FOR  COMPLETENESS,        :
13455                                   <1> ;     NOT  FOR  REFERENCE.  APPLICATIONS   WHICH  REFERENCE  ANY        :
13456                                   <1> ;     ABSOLUTE  ADDRESSES  WITHIN  THE  CODE SEGMENTS  OF  BIOS        :
13457                                   <1> ;     VIOLATE  THE  STRUCTURE  AND  DESIGN  OF  BIOS.                   :
13458                                   <1> ;                                                                       :
13459                                   <1> ;-------------------------------------------------------------------------:
13460                                   <1> ;                                                                       :
13461                                   <1> ; INPUT  (AH)= HEX COMMAND VALUE                                        :
13462                                   <1> ;                                                                       :
13463                                   <1> ;     (AH)= 00H  RESET DISK (DL = 80H,81H) / DISKETTE                   :
13464                                   <1> ;     (AH)= 01H  READ THE STATUS OF THE LAST DISK OPERATION INTO (AL)       :
13465                                   <1> ;                NOTE: DL < 80H - DISKETTE                            :
13466                                   <1> ;                      DL > 80H - DISK                                :
13467                                   <1> ;     (AH)= 02H  READ THE DESIRED SECTORS INTO MEMORY                   :
13468                                   <1> ;     (AH)= 03H  WRITE THE DESIRED SECTORS FROM MEMORY                  :
13469                                   <1> ;     (AH)= 04H  VERIFY THE DESIRED SECTORS                             :
13470                                   <1> ;     (AH)= 05H  FORMAT THE DESIRED TRACK                               :
13471                                   <1> ;     (AH)= 06H  UNUSED                                                 :
13472                                   <1> ;     (AH)= 07H  UNUSED                                                 :
13473                                   <1> ;     (AH)= 08H  RETURN THE CURRENT DRIVE PARAMETERS                        :
13474                                   <1> ;     (AH)= 09H  INITIALIZE DRIVE PAIR CHARACTERISTICS                  :
13475                                   <1> ;                INTERRUPT 41 POINTS TO DATA BLOCK FOR DRIVE 0         :
13476                                   <1> ;                INTERRUPT 46 POINTS TO DATA BLOCK FOR DRIVE 1         :
13477                                   <1> ;     (AH)= 0AH  READ LONG                                              :
13478                                   <1> ;     (AH)= 0BH  WRITE LONG  (READ & WRITE LONG ENCOMPASS 512 + 4 BYTES ECC) :
13479                                   <1> ;     (AH)= 0CH  SEEK                                                   :
13480                                   <1> ;     (AH)= 0DH  ALTERNATE DISK RESET (SEE DL)                          :
13481                                   <1> ;     (AH)= 0EH  UNUSED                                                 :
13482                                   <1> ;     (AH)= 0FH  UNUSED                                                 :
13483                                   <1> ;     (AH)= 10H  TEST DRIVE READY                                       :
13484                                   <1> ;     (AH)= 11H  RECALIBRATE                                            :
13485                                   <1> ;     (AH)= 12H  UNUSED                                                 :
13486                                   <1> ;     (AH)= 13H  UNUSED                                                 :
13487                                   <1> ;     (AH)= 14H  CONTROLLER INTERNAL DIAGNOSTIC                         :
13488                                   <1> ;     (AH)= 15H  READ DASD TYPE                                         :
13489                                   <1> ;                                                                       :
13490                                   <1> ;-------------------------------------------------------------------------
13491                                   <1> ;                                                                       :
```

```
13492                                   <1> ;     REGISTERS USED FOR FIXED DISK OPERATIONS                              :
13493                                   <1> ;                                                                           :
13494                                   <1> ;           (DL)   -  DRIVE NUMBER     (80H-81H FOR DISK. VALUE CHECKED)  :
13495                                   <1> ;           (DH)   -  HEAD NUMBER          (0-15 ALLOWED, NOT VALUE CHECKED)  :
13496                                   <1> ;           (CH)   -  CYLINDER NUMBER  (0-1023, NOT VALUE CHECKED)(SEE CL):
13497                                   <1> ;           (CL)   -  SECTOR NUMBER    (1-17, NOT VALUE CHECKED)          :
13498                                   <1> ;                                                                           :
13499                                   <1> ;                     NOTE: HIGH 2 BITS OF CYLINDER NUMBER ARE PLACED      :
13500                                   <1> ;                         IN THE HIGH 2 BITS OF THE CL REGISTER            :
13501                                   <1> ;                         (10 BITS TOTAL)                                  :
13502                                   <1> ;                                                                           :
13503                                   <1> ;           (AL)   -  NUMBER OF SECTORS (MAXIMUM POSSIBLE RANGE 1-80H,     :
13504                                   <1> ;                                       FOR READ/WRITE LONG 1-79H)          :
13505                                   <1> ;                                                                           :
13506                                   <1> ;           (ES:BX) -  ADDRESS OF BUFFER FOR READS AND WRITES,              :
13507                                   <1> ;                     (NOT REQUIRED FOR VERIFY)                             :
13508                                   <1> ;                                                                           :
13509                                   <1> ;           FORMAT (AH=5) ES:BX POINTS TO A 512 BYTE BUFFER. THE FIRST      :
13510                                   <1> ;                     2*(SECTORS/TRACK) BYTES CONTAIN F,N FOR EACH SECTOR.:
13511                                   <1> ;                     F = 00H FOR A GOOD SECTOR                             :
13512                                   <1> ;                         80H FOR A BAD SECTOR                              :
13513                                   <1> ;                     N = SECTOR NUMBER                                     :
13514                                   <1> ;                     FOR AN INTERLEAVE OF 2 AND 17 SECTORS/TRACK                :
13515                                   <1> ;                     THE TABLE SHOULD BE:                                  :
13516                                   <1> ;                                                                           :
13517                                   <1> ;                DB     00H,01H,00H,0AH,00H,02H,00H,0BH,00H,03H,00H,0CH      :
13518                                   <1> ;                DB     00H,04H,00H,0DH,00H,05H,00H,0EH,00H,06H,00H,0FH      :
13519                                   <1> ;                DB     00H,07H,00H,10H,00H,08H,00H,11H,00H,09H              :
13520                                   <1> ;                                                                           :
13521                                   <1> ;-------------------------------------------------------------------------------
13522                                   <1>
13523                                   <1> ;-------------------------------------------------------------------------------
13524                                   <1> ; OUTPUT                                                                    :
13525                                   <1> ;     AH = STATUS OF CURRENT OPERATION                                      :
13526                                   <1> ;          STATUS BITS ARE DEFINED IN THE EQUATES BELOW                        :
13527                                   <1> ;     CY = 0 SUCCESSFUL OPERATION (AH=0 ON RETURN)                          :
13528                                   <1> ;     CY = 1 FAILED OPERATION (AH HAS ERROR REASON)                         :
13529                                   <1> ;                                                                           :
13530                                   <1> ;     NOTE:  ERROR 11H  INDICATES THAT THE DATA READ HAD A RECOVERABLE      :
13531                                   <1> ;            ERROR WHICH WAS CORRECTED BY THE ECC ALGORITHM.  THE DATA      :
13532                                   <1> ;            IS PROBABLY GOOD,   HOWEVER THE BIOS ROUTINE INDICATES AN      :
13533                                   <1> ;            ERROR TO ALLOW THE CONTROLLING PROGRAM A CHANCE TO DECIDE       :
13534                                   <1> ;            FOR ITSELF.  THE  ERROR  MAY  NOT  RECUR  IF  THE DATA IS      :
13535                                   <1> ;            REWRITTEN.                                                     :
13536                                   <1> ;                                                                           :
13537                                   <1> ;     IF DRIVE PARAMETERS WERE REQUESTED (DL >= 80H),                       :
13538                                   <1> ;        INPUT:                                                             :
13539                                   <1> ;          (DL) = DRIVE NUMBER                                              :
13540                                   <1> ;          ; 27/05/2016 - TRDOS 386 (TRDOS v2.0)                                    :

13541                                   <1> ;           EBX = Buffer address for fixed disk parameters table (32 bytes)   :
13542                                   <1> ;        OUTPUT:                                                            :
13543                                   <1> ;          (DL) = NUMBER OF CONSECUTIVE ACKNOWLEDGING DRIVES ATTACHED (1-2)  :
13544                                   <1> ;               (CONTROLLER CARD ZERO TALLY ONLY)                           :
13545                                   <1> ;          (DH) = MAXIMUM USEABLE VALUE FOR HEAD NUMBER                     :
13546                                   <1> ;          (CH) = MAXIMUM USEABLE VALUE FOR CYLINDER NUMBER                    :
13547                                   <1> ;          (CL) = MAXIMUM USEABLE VALUE FOR SECTOR NUMBER                   :
13548                                   <1> ;               AND CYLINDER NUMBER HIGH BITS                               :
13549                                   <1> ;                                                                           :
13550                                   <1> ;     IF READ DASD TYPE WAS REQUESTED,                                      :
13551                                   <1> ;                                                                           :
13552                                   <1> ;     AH = 0 - NOT PRESENT                                                  :
13553                                   <1> ;          1 - DISKETTE - NO CHANGE LINE AVAILABLE                          :
13554                                   <1> ;          2 - DISKETTE - CHANGE LINE AVAILABLE                             :
13555                                   <1> ;          3 - FIXED DISK                                                   :
13556                                   <1> ;                                                                           :
13557                                   <1> ;     CX,DX = NUMBER OF 512 BYTE BLOCKS WHEN AH = 3                         :
13558                                   <1> ;                                                                           :
13559                                   <1> ;     REGISTERS WILL BE PRESERVED EXCEPT WHEN THEY ARE USED TO RETURN        :
13560                                   <1> ;     INFORMATION.                                                          :
13561                                   <1> ;                                                                           :
13562                                   <1> ;     NOTE: IF AN ERROR IS REPORTED BY THE DISK CODE, THE APPROPRIATE        :
13563                                   <1> ;           ACTION IS TO RESET THE DISK, THEN RETRY THE OPERATION.            :
13564                                   <1> ;                                                                           :
13565                                   <1> ;-------------------------------------------------------------------------------
13566                                   <1>
13567                                   <1> SENSE_FAIL   EQU    0FFH          ; NOT IMPLEMENTED
13568                                   <1> NO_ERR       EQU    0E0H          ; STATUS ERROR/ERROR REGISTER=0
13569                                   <1> WRITE_FAULT  EQU    0CCH          ; WRITE FAULT ON SELECTED DRIVE
13570                                   <1> UNDEF_ERR    EQU    0BBH          ; UNDEFINED ERROR OCCURRED
13571                                   <1> NOT_RDY      EQU    0AAH          ; DRIVE NOT READY
13572                                   <1> TIME_OUT     EQU    80H           ; ATTACHMENT FAILED TO RESPOND
13573                                   <1> BAD_SEEK     EQU    40H           ; SEEK OPERATION FAILED
13574                                   <1> BAD_CNTLR    EQU    20H           ; CONTROLLER HAS FAILED
13575                                   <1> DATA_CORRECTED     EQU    11H           ; ECC CORRECTED DATA ERROR
13576                                   <1> BAD_ECC      EQU    10H           ; BAD ECC ON DISK READ
13577                                   <1> BAD_TRACK    EQU    0BH           ; NOT IMPLEMENTED
13578                                   <1> BAD_SECTOR   EQU    0AH           ; BAD SECTOR FLAG DETECTED
13579                                   <1> ;DMA_BOUNDARY      EQU    09H           ; DATA EXTENDS TOO FAR
13580                                   <1> INIT_FAIL    EQU    07H           ; DRIVE PARAMETER ACTIVITY FAILED
13581                                   <1> BAD_RESET    EQU    05H           ; RESET FAILED
13582                                   <1> ;RECORD_NOT_FND    EQU    04H           ; REQUESTED SECTOR NOT FOUND
13583                                   <1> ;BAD_ADDR_MARK     EQU    02H           ; ADDRESS MARK NOT FOUND
13584                                   <1> ;BAD_CMD     EQU    01H           ; BAD COMMAND PASSED TO DISK I/O
13585                                   <1>
13586                                   <1> ;-----------------------------------------------------
13587                                   <1> ;                                                   :
13588                                   <1> ; FIXED DISK PARAMETER TABLE                         :
13589                                   <1> ;  -  THE TABLE IS COMPOSED OF A BLOCK DEFINED AS:  :
13590                                   <1> ;                                                   :
13591                                   <1> ;  +0 (1 WORD) - MAXIMUM NUMBER OF CYLINDERS        :
13592                                   <1> ;  +2 (1 BYTE) - MAXIMUM NUMBER OF HEADS            :
13593                                   <1> ;  +3 (1 WORD) - NOT USED/SEE PC-XT                 :
```

```
13594                                  <1> ;  +5 (1 WORD) - STARTING WRITE PRECOMPENSATION CYL :
13595                                  <1> ;  +7 (1 BYTE) - MAXIMUM ECC DATA BURST LENGTH      :
13596                                  <1> ;  +8 (1 BYTE) - CONTROL BYTE                :
13597                                  <1> ;               BIT   7 DISABLE RETRIES -OR-   :
13598                                  <1> ;               BIT   6 DISABLE RETRIES        :
13599                                  <1> ;               BIT   3 MORE THAN 8 HEADS            :
13600                                  <1> ;  +9 (3 BYTES)- NOT USED/SEE PC-XT              :
13601                                  <1> ; +12 (1 WORD) - LANDING ZONE                :
13602                                  <1> ; +14 (1 BYTE) - NUMBER OF SECTORS/TRACK           :
13603                                  <1> ; +15 (1 BYTE) - RESERVED FOR FUTURE USE           :
13604                                  <1> ;                                            :
13605                                  <1> ;      - TO DYNAMICALLY DEFINE A SET OF PARAMETERS  :
13606                                  <1> ;        BUILD A TABLE FOR UP TO 15 TYPES AND PLACE :
13607                                  <1> ;        THE CORRESPONDING VECTOR INTO INTERRUPT 41 :
13608                                  <1> ;        FOR DRIVE 0 AND INTERRUPT 46 FOR DRIVE 1.  :
13609                                  <1> ;                                            :
13610                                  <1> ;-------------------------------------------------------
13611                                  <1>
13612                                  <1> ;-------------------------------------------------------
13613                                  <1> ;                                            :
13614                                  <1> ; HARDWARE SPECIFIC VALUES                         :
13615                                  <1> ;                                            :
13616                                  <1> ;  -  CONTROLLER I/O PORT                    :
13617                                  <1> ;                                            :
13618                                  <1> ;     > WHEN READ FROM:                      :
13619                                  <1> ;     HF_PORT+0 - READ DATA (FROM CONTROLLER TO CPU)      :
13620                                  <1> ;     HF_PORT+1 - GET ERROR REGISTER               :
13621                                  <1> ;     HF_PORT+2 - GET SECTOR COUNT                 :
13622                                  <1> ;     HF_PORT+3 - GET SECTOR NUMBER                :
13623                                  <1> ;     HF_PORT+4 - GET CYLINDER LOW                 :
13624                                  <1> ;     HF_PORT+5 - GET CYLINDER HIGH (2 BITS)       :
13625                                  <1> ;     HF_PORT+6 - GET SIZE/DRIVE/HEAD        :
13626                                  <1> ;     HF_PORT+7 - GET STATUS REGISTER        :
13627                                  <1> ;                                            :
13628                                  <1> ;     > WHEN WRITTEN TO:                     :
13629                                  <1> ;     HF_PORT+0 - WRITE DATA (FROM CPU TO CONTROLLER) :
13630                                  <1> ;     HF_PORT+1 - SET PRECOMPENSATION CYLINDER      :
13631                                  <1> ;     HF_PORT+2 - SET SECTOR COUNT                 :
13632                                  <1> ;     HF_PORT+3 - SET SECTOR NUMBER                :
13633                                  <1> ;     HF_PORT+4 - SET CYLINDER LOW                 :
13634                                  <1> ;     HF_PORT+5 - SET CYLINDER HIGH (2 BITS)       :
13635                                  <1> ;     HF_PORT+6 - SET SIZE/DRIVE/HEAD        :
13636                                  <1> ;     HF_PORT+7 - SET COMMAND REGISTER       :
13637                                  <1> ;                                            :
13638                                  <1> ;-------------------------------------------------------
13639                                  <1>
13640                                  <1> ;HF_PORT     EQU    01F0H  ; DISK PORT
13641                                  <1> ;HF1_PORT    equ    0170h
13642                                  <1> ;HF_REG_PORT EQU    03F6H
13643                                  <1> ;HF1_REG_PORT      equ    0376h
13644                                  <1>
13645                                  <1> HDC1_BASEPORT      equ    1F0h
13646                                  <1> HDC2_BASEPORT      equ    170h
13647                                  <1>
13648                                  <1> align 2
13649                                  <1>
13650                                  <1> ;-----             STATUS REGISTER
13651                                  <1>
13652                                  <1> ST_ERROR     EQU    00000001B    ;
13653                                  <1> ST_INDEX     EQU    00000010B    ;
13654                                  <1> ST_CORRCTD   EQU    00000100B    ; ECC CORRECTION SUCCESSFUL
13655                                  <1> ST_DRQ              EQU    00001000B    ;
13656                                  <1> ST_SEEK_COMPL       EQU    00010000B    ; SEEK COMPLETE
13657                                  <1> ST_WRT_FLT   EQU    00100000B    ; WRITE FAULT
13658                                  <1> ST_READY     EQU    01000000B    ;
13659                                  <1> ST_BUSY      EQU    10000000B    ;
13660                                  <1>
13661                                  <1> ;-----             ERROR REGISTER
13662                                  <1>
13663                                  <1> ERR_DAM      EQU    00000001B    ; DATA ADDRESS MARK NOT FOUND
13664                                  <1> ERR_TRK_0    EQU    00000010B    ; TRACK 0 NOT FOUND ON RECAL
13665                                  <1> ERR_ABORT    EQU    00000100B    ; ABORTED COMMAND
13666                                  <1> ;            EQU    00001000B    ; NOT USED
13667                                  <1> ERR_ID               EQU    00010000B    ; ID NOT FOUND
13668                                  <1> ;            EQU    00100000B    ; NOT USED
13669                                  <1> ERR_DATA_ECC EQU    01000000B
13670                                  <1> ERR_BAD_BLOCK       EQU    10000000B
13671                                  <1>
13672                                  <1>
13673                                  <1> RECAL_CMD    EQU    00010000B    ; DRIVE RECAL(10H)
13674                                  <1> READ_CMD     EQU    00100000B    ;      READ   (20H)
13675                                  <1> WRITE_CMD    EQU    00110000B    ;      WRITE  (30H)
13676                                  <1> VERIFY_CMD   EQU    01000000B    ;      VERIFY (40H)
13677                                  <1> FMTTRK_CMD   EQU    01010000B    ; FORMAT TRACK      (50H)
13678                                  <1> INIT_CMD     EQU    01100000B    ;   INITIALIZE      (60H)
13679                                  <1> SEEK_CMD     EQU    01110000B    ;      SEEK   (70H)
13680                                  <1> DIAG_CMD     EQU    10010000B    ; DIAGNOSTIC (90H)
13681                                  <1> SET_PARM_CMD EQU    10010001B    ; DRIVE PARMS(91H)
13682                                  <1> NO_RETRIES   EQU    00000001B    ; CHD MODIFIER      (01H)
13683                                  <1> ECC_MODE     EQU    00000010B    ; CMD MODIFIER      (02H)
13684                                  <1> BUFFER_MODE  EQU    00001000B    ; CMD MODIFIER      (08H)
13685                                  <1>
13686                                  <1> ;MAX_FILE    EQU    2
13687                                  <1> ;S_MAX_FILE  EQU    2
13688                                  <1> MAX_FILE     equ    4            ; 22/12/2014
13689                                  <1> S_MAX_FILE   equ    4            ; 22/12/2014
13690                                  <1>
13691                                  <1> DELAY_1      EQU    25H          ; DELAY FOR OPERATION COMPLETE
13692                                  <1> DELAY_2      EQU    0600H        ; DELAY FOR READY
13693                                  <1> DELAY_3      EQU    0100H        ; DELAY FOR DATA REQUEST
13694                                  <1>
13695                                  <1> HF_FAIL      EQU    08H          ; CMOS FLAG IN BYTE 0EH
13696                                  <1>
```

```
13697                                  <1> ;-----             COMMAND BLOCK REFERENCE
13698                                  <1>
13699                                  <1> ;CMD_BLOCK      EQU     BP-8            ; @CMD_BLOCK REFERENCES BLOCK HEAD IN SS
13700                                  <1>                                 ;  (BP) POINTS TO COMMAND BLOCK TAIL
13701                                  <1>                                 ;      AS DEFINED BY THE "ENTER" PARMS
13702                                  <1> ; 19/12/2014
13703                                  <1> ORG_VECTOR   equ    4*13h       ; INT 13h vector
13704                                  <1> DISK_VECTOR  equ    4*40h       ; INT 40h vector (for floppy disks)
13705                                  <1> ;HDISK_INT   equ    4*76h       ; Primary HDC - Hardware interrupt (IRQ14)
13706                                  <1> ;HDISK_INT1  equ    4*76h       ; Primary HDC - Hardware interrupt (IRQ14)
13707                                  <1> ;HDISK_INT2  equ    4*77h       ; Secondary HDC - Hardware interrupt (IRQ15)
13708                                  <1> ;HF_TBL_VEC  equ    4*41h       ; Pointer to 1st fixed disk parameter table
13709                                  <1> ;HF1_TBL_VEC equ    4*46h       ; Pointer to 2nd fixed disk parameter table
13710                                  <1>
13711                                  <1> align 2
13712                                  <1>
13713                                  <1> ;--------------------------------------------------------------
13714                                  <1> ; FIXED DISK I/O SETUP                                       :
13715                                  <1> ;                                                 :
13716                                  <1> ;  -  ESTABLISH TRANSFER VECTORS FOR THE FIXED DISK          :
13717                                  <1> ;  -  PERFORM POWER ON DIAGNOSTICS                           :
13718                                  <1> ;     SHOULD AN ERROR OCCUR A "1701" MESSAGE IS DISPLAYED       :
13719                                  <1> ;                                                 :
13720                                  <1> ;--------------------------------------------------------------
13721                                  <1>
13722                                  <1> ; 29/05/2016 - TRDOS 386 (TRDOS v2.0)
13723                                  <1>
13724                                  <1> DISK_SETUP:
13725                                  <1>       ;CLI
13726                                  <1>       ;;MOV  AX,ABS0                  ; GET ABSOLUTE SEGMENT
13727                                  <1>       ;xor   ax,ax
13728                                  <1>       ;MOV   DS,AX                    ; SET SEGMENT REGISTER
13729                                  <1>       ;MOV   AX, [ORG_VECTOR]         ; GET DISKETTE VECTOR
13730                                  <1>       ;MOV   [DISK_VECTOR],AX         ;  INTO INT 40H
13731                                  <1>       ;MOV   AX, [ORG_VECTOR+2]
13732                                  <1>       ;MOV   [DISK_VECTOR+2],AX
13733                                  <1>       ;MOV   word [ORG_VECTOR],DISK_IO ; FIXED DISK HANDLER
13734                                  <1>       ;MOV   [ORG_VECTOR+2],CS
13735                                  <1>       ; 1st controller (primary master, slave)   - IRQ 14
13736                                  <1>       ;;MOV  word [HDISK_INT],HD_INT          ; FIXED DISK INTERRUPT
13737                                  <1>       ;mov   word [HDISK_INT1],HD_INT  ;
13738                                  <1>       ;;MOV  [HDISK_INT+2],CS
13739                                  <1>       ;mov   [HDISK_INT1+2],CS
13740                                  <1>       ; 2nd controller (secondary master, slave) - IRQ 15
13741                                  <1>       ;mov   word [HDISK_INT2],HD1_INT ;
13742                                  <1>       ;mov   [HDISK_INT2+2],CS
13743                                  <1>       ;
13744                                  <1>       ;;MOV  word [HF_TBL_VEC],HD0_DPT ; PARM TABLE DRIVE 80
13745                                  <1>       ;;MOV  word [HF_TBL_VEC+2],DPT_SEGM
13746                                  <1>       ;;MOV  word [HF1_TBL_VEC],HD1_DPT; PARM TABLE DRIVE 81
13747                                  <1>       ;;MOV  word [HF1_TBL_VEC+2],DPT_SEGM
13748                                  <1>       ;push  cs
13749                                  <1>       ;pop   ds
13750                                  <1>       ;mov   word [HDPM_TBL_VEC],HD0_DPT      ; PARM TABLE DRIVE 80h
13751                                  <1>       ;mov   word [HDPM_TBL_VEC+2],DPT_SEGM
13752 0000420E C705[C0520100]0000-     <1>       mov    dword [HDPM_TBL_VEC], (DPT_SEGM*16)+HD0_DPT
13753 00004216 0900                   <1>
13754                                  <1>       ;mov   word [HDPS_TBL_VEC],HD1_DPT      ; PARM TABLE DRIVE 81h
13755                                  <1>       ;mov   word [HDPS_TBL_VEC+2],DPT_SEGM
13756 00004218 C705[C4520100]2000-     <1>       mov    dword [HDPS_TBL_VEC], (DPT_SEGM*16)+HD1_DPT
13757 00004220 0900                   <1>
13758                                  <1>       ;mov   word [HDSM_TBL_VEC],HD2_DPT      ; PARM TABLE DRIVE 82h
13759                                  <1>       ;mov   word [HDSM_TBL_VEC+2],DPT_SEGM
13760 00004222 C705[C8520100]4000-     <1>       mov    dword [HDSM_TBL_VEC], (DPT_SEGM*16)+HD2_DPT
13761 0000422A 0900                   <1>
13762                                  <1>       ;mov   word [HDSS_TBL_VEC],HD3_DPT      ; PARM TABLE DRIVE 83h
13763                                  <1>       ;mov   word [HDSS_TBL_VEC+2],DPT_SEGM
13764 0000422C C705[CC520100]6000-     <1>       mov    dword [HDSS_TBL_VEC], (DPT_SEGM*16)+HD3_DPT
13765 00004234 0900                   <1>
13766                                  <1>       ;
13767                                  <1>       ;;IN   AL,INTB01          ; TURN ON SECOND INTERRUPT CHIP
13768                                  <1>       ;;;AND AL,0BFH
13769                                  <1>       ;;and  al, 3Fh                  ; enable IRQ 14 and IRQ 15
13770                                  <1>       ;;;JMP $+2
13771                                  <1>       ;;IODELAY
13772                                  <1>       ;;OUT  INTB01,AL
13773                                  <1>       ;;IODELAY
13774                                  <1>       ;;IN   AL,INTA01          ; LET INTERRUPTS PASS THRU TO
13775                                  <1>       ;;AND  AL,0FBH            ;  SECOND CHIP
13776                                  <1>       ;;;JMP $+2
13777                                  <1>       ;;IODELAY
13778                                  <1>       ;;OUT  INTA01,AL
13779                                  <1>       ;
13780                                  <1>       ;STI
13781                                  <1>       ;;PUSH DS                 ; MOVE ABS0 POINTER TO
13782                                  <1>       ;;POP  ES                 ; EXTRA SEGMENT POINTER
13783                                  <1>       ;;;CALL      DDS                ; ESTABLISH DATA SEGMENT
13784                                  <1>       ;;MOV  byte [DISK_STATUS1],0     ; RESET THE STATUS INDICATOR
13785                                  <1>       ;;MOV  byte [HF_NUM],0           ; ZERO NUMBER OF FIXED DISKS
13786                                  <1>       ;;MOV  byte [CONTROL_BYTE],0
13787                                  <1>       ;;MOV  byte [PORT_OFF],0   ; ZERO CARD OFFSET
13788                                  <1>       ; 20/12/2014 - private code by Erdogan Tan
13789                                  <1>                    ; (out of original PC-AT, PC-XT BIOS code)
13790                                  <1>       ;mov   si, hd0_type
13791 00004236 BE[F85C0000]           <1>       mov    esi, hd0_type
13792                                  <1>       ;mov   cx, 4
13793 0000423B B904000000             <1>       mov    ecx, 4
13794                                  <1> hde_l:
13795 00004240 AC                     <1>       lodsb
13796 00004241 3C80                   <1>       cmp    al, 80h                  ; 8?h = existing
13797 00004243 7206                   <1>       jb     short _L4
13798 00004245 FE05[BC520100]         <1>       inc    byte [HF_NUM]       ; + 1 hard (fixed) disk drives
13799                                  <1> _L4: ; 26/02/2015
```

```
13800 0000424B E2F3               <1>       loop   hde_l
13801                             <1> ;_L4:                           ; 0 <= [HF_NUM] =< 4
13802                             <1> ;L4:
13803                             <1>       ;
13804                             <1>       ;; 31/12/2014 - cancel controller diagnostics here
13805                             <1>       ;;;mov       cx, 3  ; 26/12/2014 (Award BIOS 1999)
13806                             <1>       ;;mov  cl, 3
13807                             <1>       ;;
13808                             <1>       ;;MOV  DL,80H             ; CHECK THE CONTROLLER
13809                             <1> ;;hdc_dl:
13810                             <1>       ;;MOV  AH,14H             ; USE CONTROLLER DIAGNOSTIC COMMAND
13811                             <1>       ;;INT  13H                ; CALL BIOS WITH DIAGNOSTIC COMMAND
13812                             <1>       ;;;JC  short CTL_ERRX           ; DISPLAY ERROR MESSAGE IF BAD RETURN
13813                             <1>       ;;;jc  short POD_DONE ;22/12/2014
13814                             <1>       ;;jnc  short hdc_reset0
13815                             <1>       ;;loop hdc_dl
13816                             <1>       ;;; 27/12/2014
13817                             <1>       ;;stc
13818                             <1>       ;;retn
13819                             <1>       ;
13820                             <1> ;;hdc_reset0:
13821                             <1>       ; 18/01/2015
13822 0000424D 8A0D[BC520100]     <1>       mov    cl, [HF_NUM]
13823 00004253 20C9               <1>       and    cl, cl
13824 00004255 740E               <1>       jz     short POD_DONE
13825                             <1>       ;
13826 00004257 B27F               <1>       mov    dl, 7Fh
13827                             <1> hdc_reset1:
13828 00004259 FEC2               <1>       inc    dl
13829                             <1>       ;; 31/12/2015
13830                             <1>       ;;push dx
13831                             <1>       ;;push cx
13832                             <1>       ;;push ds
13833                             <1>       ;;sub  ax, ax
13834                             <1>       ;;mov  ds, ax
13835                             <1>       ;;MOV  AX, [TIMER_LOW]          ; GET START TIMER COUNTS
13836                             <1>       ;;pop  ds
13837                             <1>       ;;MOV  BX,AX
13838                             <1>       ;;ADD  AX,6*182           ; 60 SECONDS* 18.2
13839                             <1>       ;;MOV  CX,AX
13840                             <1>       ;;mov  word [wait_count], 0      ; 22/12/2014 (reset wait counter)
13841                             <1>       ;;
13842                             <1>       ;; 31/12/2014 - cancel HD_RESET_1
13843                             <1>       ;;CALL HD_RESET_1         ; SET UP DRIVE 0, (1,2,3)
13844                             <1>       ;;pop  cx
13845                             <1>       ;;pop  dx
13846                             <1>       ;;
13847                             <1>       ; 18/01/2015
13848 0000425B B40D               <1>       mov    ah, 0Dh ; ALTERNATE RESET
13849                             <1>       ;int   13h
13850 0000425D E8A4FFFFFF         <1>       call   int13h
13851 00004262 E2F5               <1>       loop   hdc_reset1
13852 00004264 F8                 <1>       clc    ; 29/05/2016
13853                             <1> POD_DONE:
13854 00004265 C3                 <1>       RETn
13855                             <1>
13856                             <1> ;;-----     POD_ERROR
13857                             <1>
13858                             <1> ;;CTL_ERRX:
13859                             <1> ;     ;MOV   SI,OFFSET F1782     ; CONTROLLER ERROR
13860                             <1> ;     ;CALL  SET_FAIL            ; DO NOT IPL FROM DISK
13861                             <1> ;     ;CALL  E_MSG               ; DISPLAY ERROR AND SET (BP) ERROR FLAG
13862                             <1> ;     ;JMP   short POD_DONE
13863                             <1>
13864                             <1> ;;HD_RESET_1:
13865                             <1> ;;    ;PUSH  BX                  ; SAVE TIMER LIMITS
13866                             <1> ;;    ;PUSH  CX
13867                             <1> ;;RES_1: MOV AH,09H              ; SET DRIVE PARAMETERS
13868                             <1> ;;    INT    13H
13869                             <1> ;;    JC     short RES_2
13870                             <1> ;;    MOV    AH,11H             ; RECALIBRATE DRIVE
13871                             <1> ;;    INT    13H
13872                             <1> ;;    JNC    short RES_CK        ; DRIVE OK
13873                             <1> ;;RES_2: ;CALL     POD_TCHK           ; CHECK TIME OUT
13874                             <1> ;;    cmp    word [wait_count], 6*182 ; waiting time (in timer ticks)
13875                             <1> ;;                                ; (30 seconds)
13876                             <1> ;;    ;cmc
13877                             <1> ;;    ;JNC   short RES_1
13878                             <1> ;;    jb     short RES_1
13879                             <1> ;;;RES_FL: ;MOV    SI,OFFSET F1781     ; INDICATE DISK 1 FAILURE;
13880                             <1> ;;    ;TEST  DL,1
13881                             <1> ;;    ;JNZ   RES_E1
13882                             <1> ;;    ;MOV   SI,OFFSET F1780     ; INDICATE DISK 0 FAILURE
13883                             <1> ;;    ;CALL  SET_FAIL            ; DO NOT TRY TO IPL DISK 0
13884                             <1> ;;    ;JMP   SHORT RES_E1
13885                             <1> ;;RES_ER: ; 22/12/2014
13886                             <1> ;;RES_OK:
13887                             <1> ;;    ;POP   CX                  ; RESTORE TIMER LIMITS
13888                             <1> ;;    ;POP   BX
13889                             <1> ;;    RETn
13890                             <1> ;;
13891                             <1> ;;RES_RS: MOV      AH,00H              ; RESET THE DRIVE
13892                             <1> ;;    INT    13H
13893                             <1> ;;RES_CK: MOV      AH,08H              ; GET MAX CYLINDER,HEAD,SECTOR
13894                             <1> ;;    MOV    BL,DL               ; SAVE DRIVE CODE
13895                             <1> ;;    INT    13H
13896                             <1> ;;    JC     short RES_ER
13897                             <1> ;;    MOV    [NEC_STATUS],CX     ; SAVE MAX CYLINDER, SECTOR
13898                             <1> ;;    MOV    DL,BL               ; RESTORE DRIVE CODE
13899                             <1> ;;RES_3: MOV AX,0401H            ; VERIFY THE LAST SECTOR
13900                             <1> ;;    INT    13H
13901                             <1> ;;    JNC    short RES_OK        ; VERIFY OK
13902                             <1> ;;    CMP    AH,BAD_SECTOR       ; OK ALSO IF JUST ID READ
```

```
13903                                  <1> ;;    JE     short RES_OK
13904                                  <1> ;;    CMP    AH,DATA_CORRECTED
13905                                  <1> ;;    JE     short RES_OK
13906                                  <1> ;;    CMP    AH,BAD_ECC
13907                                  <1> ;;    JE     short RES_OK
13908                                  <1> ;;    ;CALL  POD_TCHK            ; CHECK FOR TIME OUT
13909                                  <1> ;;    cmp    word [wait_count], 6*182 ; waiting time (in timer ticks)
13910                                  <1> ;;                               ; (60 seconds)
13911                                  <1> ;;    cmc
13912                                  <1> ;;    JC     short RES_ER        ; FAILED
13913                                  <1> ;;    MOV    CX,[NEC_STATUS]     ; GET SECTOR ADDRESS, AND CYLINDER
13914                                  <1> ;;    MOV    AL,CL               ; SEPARATE OUT SECTOR NUMBER
13915                                  <1> ;;    AND    AL,3FH
13916                                  <1> ;;    DEC    AL                  ; TRY PREVIOUS ONE
13917                                  <1> ;;    JZ     short RES_RS        ; WE'VE TRIED ALL SECTORS ON TRACK
13918                                  <1> ;;    AND    CL,0C0H             ; KEEP CYLINDER BITS
13919                                  <1> ;;    OR     CL,AL               ; MERGE SECTOR WITH CYLINDER BITS
13920                                  <1> ;;    MOV    [NEC_STATUS],CX     ; SAVE CYLINDER, NEW SECTOR NUMBER
13921                                  <1> ;;    JMP    short RES_3         ; TRY AGAIN
13922                                  <1> ;;;RES_ER: MOV     SI,OFFSET F1791     ; INDICATE DISK 1 ERROR
13923                                  <1> ;;    ;TEST  DL,1
13924                                  <1> ;;    ;JNZ   short RES_E1
13925                                  <1> ;;    ;MOV   SI,OFFSET F1790     ; INDICATE DISK 0 ERROR
13926                                  <1> ;;;RES_E1:
13927                                  <1> ;;    ;CALL  E_MSG               ; DISPLAY ERROR AND SET (BP) ERROR FLAG
13928                                  <1> ;;;RES_OK:
13929                                  <1> ;;    ;POP   CX                  ; RESTORE TIMER LIMITS
13930                                  <1> ;;    ;POP   BX
13931                                  <1> ;;    ;RETn
13932                                  <1> ;
13933                                  <1> ;;SET_FAIL:
13934                                  <1> ;     ;MOV   AX,X*(CMOS_DIAG+NMI)      ; GET CMOS ERROR BYTE
13935                                  <1> ;     ;CALL  CMOS_READ
13936                                  <1> ;     ;OR    AL,HF_FAIL          ; SET DO NOT IPL FROM DISK FLAG
13937                                  <1> ;     ;XCHG  AH,AL               ; SAVE IT
13938                                  <1> ;     ;CALL  CMOS_WRITE          ; PUT IT OUT
13939                                  <1> ;     ;RETn
13940                                  <1> ;
13941                                  <1> ;;POD_TCHK:                      ; CHECK FOR 30 SECOND TIME OUT
13942                                  <1> ;     ;POP   AX                  ; SAVE RETURN
13943                                  <1> ;     ;POP   CX                  ; GET TIME OUT LIMITS
13944                                  <1> ;     ;POP   BX
13945                                  <1> ;     ;PUSH  BX                  ; AND SAVE THEM AGAIN
13946                                  <1> ;     ;PUSH  CX
13947                                  <1> ;     ;PUSH  AX
13948                                  <1> ;     ;push  ds
13949                                  <1> ;     ;xor   ax, ax
13950                                  <1> ;     ;mov   ds, ax              ; RESTORE RETURN
13951                                  <1> ;     ;MOV   AX, [TIMER_LOW]           ; AX = CURRENT TIME
13952                                  <1> ;     ;                          ; BX = START TIME
13953                                  <1> ;     ;                          ; CX = END TIME
13954                                  <1> ;     ;pop   ds
13955                                  <1> ;     ;CMP   BX,CX
13956                                  <1> ;     ;JB    short TCHK1         ; START < END
13957                                  <1> ;     ;CMP   BX,AX
13958                                  <1> ;     ;JB    short TCHKG         ; END < START < CURRENT
13959                                  <1> ;     ;JMP   SHORT TCHK2         ; END, CURRENT < START
13960                                  <1> ;;TCHK1: CMP AX,BX
13961                                  <1> ;;    JB     short TCHKNG        ; CURRENT < START < END
13962                                  <1> ;;TCHK2: CMP AX,CX
13963                                  <1> ;;    JB     short TCHKG         ; START < CURRENT < END
13964                                  <1> ;;                               ; OR CURRENT < END < START
13965                                  <1> ;;TCHKNG: STC                          ; CARRY SET INDICATES TIME OUT
13966                                  <1> ;;    RETn
13967                                  <1> ;;TCHKG: CLC                     ; INDICATE STILL TIME
13968                                  <1> ;;    RETn
13969                                  <1> ;;
13970                                  <1> ;;int_13h:
13971                                  <1>
13972                                  <1> ;----------------------------------------
13973                                  <1> ;     FIXED DISK BIOS ENTRY POINT      :
13974                                  <1> ;----------------------------------------
13975                                  <1>
13976                                  <1> ; 15/01/2017
13977                                  <1> ; 14/01/2017
13978                                  <1> ; 07/01/2017
13979                                  <1> ; 02/01/2017
13980                                  <1> ; 01/06/2016
13981                                  <1> ; 16/05/2016, 27/05/2016, 28/05/2016, 29/05/2016
13982                                  <1> ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
13983                                  <1> int33h:  ; DISK I/O
13984                                  <1>       ; 29/05/2016
13985 00004266 80642408FE             <1>       and    byte [esp+8], 11111110b  ; clear carry bit of eflags register
13986                                  <1>       ; 16/05/2016
13987 0000426B 1E                     <1>       push   ds
13988 0000426C 53                     <1>       push   ebx ; user's buffer address (virtual)
13989 0000426D 66BB1000               <1>       mov    bx, KDATA ; System (Kernel's) data segment
13990 00004271 8EDB                   <1>       mov    ds, bx
13991                                  <1>
13992                                  <1>       ;;15/01/2017
13993                                  <1>       ; 14/01/2017
13994                                  <1>       ; 02/01/2017
13995                                  <1>       ;;mov  byte [intflg], 33h  ; disk io interrupt
13996                                  <1>       ;pop   ebx
13997                                  <1>       ;mov   [user_buffer], ebx
13998                                  <1>
13999 00004273 8F05[B05F0100]         <1>       pop    dword [user_buffer] ; 01/06/2016
14000                                  <1>
14001 00004279 C605[E6580100]00       <1>       mov    byte [scount], 0 ; sector count for transfer
14002 00004280 80FC03                 <1>       cmp    ah, 03h ; chs write
14003 00004283 7744                   <1>       ja     short int33h_2
14004 00004285 7407                   <1>       je     short int33h_0
14005 00004287 80FC02                 <1>       cmp    ah, 02h ; chs read
```

```
14006 0000428A 726A                <1>       jb     short int33h_5
14007 0000428C EB63                <1>       jmp    short int33h_4
14008                              <1> int33h_0:
14009                              <1>       ; transfer user's buffer content to sector buffer
14010 0000428E 51                  <1>       push   ecx
14011 0000428F 0FB6C8              <1>       movzx  ecx, al
14012                              <1> int33h_1:
14013 00004292 56                  <1>       push   esi
14014 00004293 8B35[B05F0100]      <1>       mov    esi, [user_buffer]
14015                              <1>       ; esi = user's buffer address (virtual, ebx)
14016 00004299 57                  <1>       push   edi
14017 0000429A 06                  <1>       push   es
14018 0000429B 50                  <1>       push   eax
14019 0000429C 66B81000            <1>       mov    ax, KDATA
14020 000042A0 8EC0                <1>       mov    es, ax
14021 000042A2 BF00000700          <1>       mov    edi, Cluster_Buffer
14022 000042A7 C1E109              <1>       shl    ecx, 9 ; * 512
14023 000042AA E848A60000          <1>       call   transfer_from_user_buffer
14024 000042AF 58                  <1>       pop    eax
14025 000042B0 07                  <1>       pop    es
14026 000042B1 5F                  <1>       pop    edi
14027 000042B2 5E                  <1>       pop    esi
14028 000042B3 59                  <1>       pop    ecx
14029 000042B4 7340                <1>       jnc    short int33h_5
14030 000042B6 8B1D[B05F0100]      <1>       mov    ebx, [user_buffer] ; 01/06/2016
14031 000042BC 1F                  <1>       pop    ds
14032                              <1>
14033                              <1>       ;;15/01/2017
14034                              <1>       ; 02/01/2017
14035                              <1>       ;cli
14036                              <1>       ;;mov  byte [ss:intflg], 0 ; 07/01/2017
14037                              <1>       ;
14038                              <1>       ; (*) 29/05/2016
14039                              <1>       ; (*) retf 4 ; skip eflags on stack
14040                              <1>
14041                              <1>       ; 29/05/2016 -set carry flag on stack-
14042                              <1>       ; [esp] = EIP
14043                              <1>       ; [esp+4] = CS
14044                              <1>       ; [esp+8] = E-FLAGS
14045 000042BD 804C240801         <1>       or     byte [esp+8], 1  ; set carry bit of eflags register
14046                              <1>       ; [esp+12] = ESP (user)
14047                              <1>       ; [esp+16] = SS (User)
14048 000042C2 B8FF000000          <1>       mov    eax, 0FFh ; Unknown error !?
14049                              <1>       ;iretd
14050 000042C7 EB79                <1>       jmp    short int33h_7  ; 07/01/2017
14051                              <1>
14052                              <1>       ; (*) 29/05/2016 - 'ref 4' intruction causes to stack fault
14053                              <1>       ; (OUTER-PRIVILEGE-LEVEL)
14054                              <1>       ; INTEL 80386 PROGRAMMER'S REFERENCE MANUAL 1986
14055                              <1>       ; // RETF instruction:
14056                              <1>       ;
14057                              <1>       ; IF OperandMode=32 THEN
14058                              <1>       ;    Load CS:EIP from stack;
14059                              <1>       ;    Set CS RPL to CPL;
14060                              <1>       ;    Increment eSP by 8 plus the immediate offset if it exists;
14061                              <1>       ;    Load SS:eSP from stack;
14062                              <1>       ; ELSE (* OperandMode=16 *)
14063                              <1>       ;    Load CS:IP from stack;
14064                              <1>       ;    Set CS RPL to CPL;
14065                              <1>       ;    Increment eSP by 4 plus the immediate offset if it exists;
14066                              <1>       ;    Load SS:eSP from stack;
14067                              <1>       ; FI;
14068                              <1>       ;
14069                              <1>       ; //
14070                              <1>
14071                              <1> int33h_2:
14072 000042C9 80FC05              <1>       cmp    ah, 05h ; format track
14073 000042CC 770A                <1>       ja     short int33h_3
14074 000042CE 7226                <1>       jb     short int33h_5
14075 000042D0 51                  <1>       push   ecx
14076 000042D1 B901000000          <1>       mov    ecx, 1
14077 000042D6 EBBA                <1>       jmp    short int33h_1
14078                              <1> int33h_3:
14079 000042D8 80FC1C              <1>       cmp    ah, 1Ch ; LBA write
14080 000042DB 7719                <1>       ja     short int33h_5
14081 000042DD 74AF                <1>       je     short int33h_0
14082 000042DF 80FC1B              <1>       cmp    ah, 1Bh ; LBA read
14083 000042E2 740D                <1>       je     short int33h_4
14084 000042E4 80FC08              <1>       cmp    ah, 08h ; get disk parameters
14085 000042E7 750D                <1>       jne    short int33h_5
14086                              <1>       ; 01/06/2016
14087 000042E9 8B1D[B05F0100]      <1>       mov    ebx, [user_buffer] ; user's buffer address
14088 000042EF EB0A                <1>       jmp    short int33h_6
14089                              <1> int33h_4:
14090 000042F1 A2[E6580100]        <1>       mov    byte [scount], al ; <= 128 sectors
14091                              <1> int33h_5:
14092 000042F6 BB00000700          <1>       mov    ebx, Cluster_Buffer ; max. 65536 bytes
14093                              <1>                                  ; buf. addr: 70000h
14094                              <1>       ;mov   byte [ClusterBuffer_Valid], 0
14095                              <1> int33h_6:
14096 000042FB 1F                  <1>       pop    ds
14097 000042FC 9C                  <1>       pushfd
14098 000042FD 0E                  <1>       push   cs
14099 000042FE E84D000000          <1>       call   DISK_IO
14100 00004303 2E8B1D[B05F0100]    <1>       mov    ebx, [CS:user_buffer] ; 01/06/2016
14101 0000430A 723D                <1>       jc     short int33h_9
14102                              <1>       ;
14103 0000430C 2E803D[E6580100]00  <1>       cmp    byte [CS:scount], 0
14104 00004314 762C                <1>       jna    short int33h_7
14105                              <1>       ; transfer sector buffer content to user's buffer
14106 00004316 06                  <1>       push   es
14107 00004317 1E                  <1>       push   ds
14108 00004318 50                  <1>       push   eax
```

```
14109 00004319 66B81000            <1>       mov    ax, KDATA
14110 0000431D 8ED8                <1>       mov    ds, ax
14111 0000431F 8EC0                <1>       mov    es, ax
14112 00004321 51                  <1>       push   ecx
14113 00004322 56                  <1>       push   esi
14114 00004323 57                  <1>       push   edi
14115 00004324 0FB60D[E6580100]    <1>       movzx  ecx, byte [scount]
14116 0000432B C1E109              <1>       shl    ecx, 9 ; * 512 bytes
14117 0000432E 89DF                <1>       mov    edi, ebx ; user's buffer address
14118 00004330 BE00000700          <1>       mov    esi, Cluster_Buffer
14119 00004335 E873A50000          <1>       call   transfer_to_user_buffer
14120 0000433A 5F                  <1>       pop    edi
14121 0000433B 5E                  <1>       pop    esi
14122 0000433C 59                  <1>       pop    ecx
14123 0000433D 58                  <1>       pop    eax
14124 0000433E 1F                  <1>       pop    ds
14125 0000433F 07                  <1>       pop    es
14126 00004340 7202                <1>       jc     short int33h_8
14127                              <1> int33h_7:
14128 00004342 FA                  <1>       cli
14129                              <1>       ;;15/01/2017
14130                              <1>       ;;mov  byte [ss:intflg], 0 ; 07/01/2017
14131                              <1>       ; cf = 0  ; use eflags which is in stack
14132 00004343 CF                  <1>       iretd
14133                              <1> int33h_8:
14134 00004344 B8FF000000          <1>       mov    eax, 0FFh ; Unknown error !?
14135                              <1> int33h_9:
14136                              <1>       ; cf = 1
14137                              <1>
14138                              <1>       ; (*) 29/05/2016
14139                              <1>       ; (*) retf 4 ; skip eflags on stack
14140                              <1>       ; Note: This 'retf 4' was wrong, -it was causing
14141                              <1>       ;       to stack errors in ring 3-
14142                              <1>       ;     POP sequence of 'retf 4' is as
14143                              <1>       ;       "eip, cs, eflags, esp, ss, +4 bytes"
14144                              <1>         ;       it is not as "eip, cs, +4 bytes, esp, ss" !
14145                              <1>
14146                              <1>       ; 29/05/2016 -set carry flag on stack-
14147 00004349 804C240801          <1>       or     byte [esp+8], 1  ; set carry bit of eflags register
14148                              <1>       ;iretd
14149 0000434E EBF2                <1>       jmp    short int33h_7 ; 07/01/2017
14150                              <1>
14151                              <1> ; 09/12/2017
14152                              <1> ; 29/05/2016
14153                              <1> ; 27/05/2016 - TRDOS 386 (TRDOS v2.0)
14154                              <1>
14155                              <1> DISK_IO:
14156 00004350 80FA80              <1>       CMP    DL,80H              ; TEST FOR FIXED DISK DRIVE
14157                              <1>       ;JAE   short A1            ; YES, HANDLE HERE
14158                              <1>       ;;;INT 40H                 ; DISKETTE HANDLER
14159                              <1>       ;;call int40h
14160 00004353 0F8222F0FFFF        <1>       jb     DISKETTE_IO_1
14161                              <1> ;RET_2:
14162                              <1>       ;RETf  2                   ; BACK TO CALLER
14163                              <1> ;     retf   4
14164                              <1> A1:
14165 00004359 FB                  <1>       STI                        ; ENABLE INTERRUPTS
14166                              <1>       ;; 04/01/2015
14167                              <1>       ;;OR   AH,AH
14168                              <1>       ;;JNZ  short A2
14169                              <1>       ;;INT  40H                 ; RESET NEC WHEN AH=0
14170                              <1>       ;;SUB  AH,AH
14171 0000435A 80FA83              <1>       CMP    DL,(80H + S_MAX_FILE - 1)
14172                              <1>       ;JA    short RET_2
14173 0000435D 7616                <1>       jna    short _A0
14174                              <1>       ; 29/05/2016
14175 0000435F 1E                  <1>       push   ds
14176 00004360 6650                <1>       push   ax
14177 00004362 66B81000            <1>       mov    ax, KDATA
14178 00004366 8ED8                <1>       mov    ds, ax
14179 00004368 6658                <1>       pop    ax
14180 0000436A B4AA                <1>         mov     ah, 0AAh        ; Hard disk drive not ready !
14181                              <1>                                 ; (Programmer's guide to AMIBIOS, 1992)
14182 0000436C 8825[BB520100]      <1>       mov     byte [DISK_STATUS1], ah
14183 00004372 1F                  <1>       pop    ds
14184 00004373 EB38                <1>       jmp    short RET_2
14185                              <1> _A0:
14186                              <1>       ; 18/01/2015
14187 00004375 08E4                <1>       or     ah,ah
14188 00004377 743A                <1>       jz     short A4
14189 00004379 80FC0D              <1>       cmp    ah, 0Dh      ; Alternate reset
14190 0000437C 7504                <1>       jne    short A2
14191 0000437E 28E4                <1>       sub    ah,ah  ; Reset
14192 00004380 EB31                <1>       jmp    short A4
14193                              <1> A2:
14194 00004382 80FC08              <1>       CMP    AH,08H              ; GET PARAMETERS IS A SPECIAL CASE
14195                              <1>       ;JNZ   short A3
14196                              <1>         ;JMP    GET_PARM_N
14197 00004385 0F8432030000        <1>       je     GET_PARM_N
14198 0000438B 80FC15              <1> A3:   CMP    AH,15H              ; READ DASD TYPE IS ALSO
14199                              <1>       ;JNZ   short A4
14200                              <1>         ;JMP    READ_DASD_TYPE
14201 0000438E 0F84DB020000        <1>         je      READ_DASD_TYPE
14202                              <1>       ; 02/02/2015
14203 00004394 80FC1D              <1>       cmp    ah, 1Dh                  ;(Temporary for Retro UNIX 386 v1)
14204                              <1>       ; 12/01/2015
14205 00004397 F5                  <1>       cmc
14206 00004398 7319                <1>       jnc    short A4
14207                              <1> int33h_bad_cmd:
14208                              <1>       ; 16/05/2016
14209                              <1>       ; 30/01/2015
14210                              <1>       ; 29/05/2016
14211 0000439A 1E                  <1>       push   ds
```

```
14212 0000439B 6650                <1>       push   ax
14213 0000439D 66B81000            <1>       mov    ax, KDATA
14214 000043A1 8ED8                <1>       mov    ds, ax
14215 000043A3 6658                <1>       pop    ax
14216 000043A5 B401                <1>       mov    ah, BAD_CMD
14217 000043A7 8825[BB520100]      <1>       mov     [DISK_STATUS1], ah ; BAD_CMD  ; COMMAND ERROR
14218                              <1>         ;jmp short RET_2
14219                              <1> RET_2:
14220                              <1>       ; (*) 29/05/2016
14221                              <1>       ; (*) retf 4
14222 000043AD 804C240801          <1>       or     byte [esp+8], 1 ; set carry bit of eflags register
14223 000043B2 CF                  <1>       iretd
14224                              <1> A4:                             ; SAVE REGISTERS DURING OPERATION
14225 000043B3 C8080000            <1>       ENTER  8,0                ; SAVE (BP) AND MAKE ROOM FOR @CMD_BLOCK
14226 000043B7 53                  <1>       PUSH   eBX                ;  IN THE STACK, THE COMMAND BLOCK IS:
14227 000043B8 51                  <1>       PUSH   eCX                ;   @CMD_BLOCK == BYTE PTR [BP]-8
14228 000043B9 52                  <1>       PUSH   eDX
14229 000043BA 1E                  <1>       PUSH   DS
14230 000043BB 06                  <1>       PUSH   ES
14231 000043BC 56                  <1>       PUSH   eSI
14232 000043BD 57                  <1>       PUSH   eDI
14233                              <1>       ;;04/01/2015
14234                              <1>       ;;OR   AH,AH              ; CHECK FOR RESET
14235                              <1>       ;;JNZ  short A5
14236                              <1>       ;;MOV  DL,80H             ; FORCE DRIVE 80 FOR RESET
14237                              <1> ;;A5:
14238                              <1>       ;push  cs
14239                              <1>       ;pop   ds
14240                              <1>       ; 21/02/2015
14241 000043BE 6650                <1>       push   ax
14242 000043C0 66B81000            <1>       mov    ax, KDATA
14243 000043C4 8ED8                <1>       mov    ds, ax
14244 000043C6 8EC0                <1>       mov    es, ax
14245 000043C8 6658                <1>       pop    ax
14246 000043CA E88D000000          <1>       CALL   DISK_IO_CONT       ; PERFORM THE OPERATION
14247                              <1>       ;;CALL DDS                ; ESTABLISH SEGMENT
14248 000043CF 8A25[BB520100]      <1>       MOV    AH,[DISK_STATUS1]  ; GET STATUS FROM OPERATION
14249                              <1>       ;(*) CMP AH,1             ; SET THE CARRY FLAG TO INDICATE
14250                              <1>       ;(*) CMC                  ; SUCCESS OR FAILURE
14251 000043D5 5F                  <1>       POP    eDI                ; RESTORE REGISTERS
14252 000043D6 5E                  <1>       POP    eSI
14253 000043D7 07                  <1>         POP     ES
14254 000043D8 1F                  <1>         POP     DS
14255 000043D9 5A                  <1>       POP    eDX
14256 000043DA 59                  <1>       POP    eCX
14257 000043DB 5B                  <1>       POP    eBX
14258 000043DC C9                  <1>       LEAVE                     ; ADJUST (SP) AND RESTORE (BP)
14259                              <1>       ;RETf  2                  ; THROW AWAY SAVED FLAGS
14260                              <1>       ; (*) 29/05/2016
14261                              <1>       ; (*) retf 4
14262 000043DD 80FC01              <1>       cmp    ah, 1
14263 000043E0 7205                <1>       jc     short _A5
14264 000043E2 804C240801          <1>       or     byte [esp+8], 1 ; set carry bit of eflags register
14265                              <1> _A5:
14266 000043E7 CF                  <1>       iretd
14267                              <1>
14268                              <1> ; 21/02/2015
14269                              <1> ;       dw --> dd
14270                              <1> D1:                             ; FUNCTION TRANSFER TABLE
14271 000043E8 [AB450000]          <1>       dd     DISK_RESET         ; 000H
14272 000043EC [22460000]          <1>       dd     RETURN_STATUS      ; 001H
14273 000043F0 [2F460000]          <1>       dd     DISK_READ          ; 002H
14274 000043F4 [38460000]          <1>       dd     DISK_WRITE         ; 003H
14275 000043F8 [41460000]          <1>       dd     DISK_VERF          ; 004H
14276 000043FC [59460000]          <1>       dd     FMT_TRK            ; 005H
14277 00004400 [A1450000]          <1>       dd     BAD_COMMAND        ; 006H FORMAT BAD SECTORS
14278 00004404 [A1450000]          <1>       dd     BAD_COMMAND        ; 007H FORMAT DRIVE
14279 00004408 [A1450000]          <1>       dd     BAD_COMMAND        ; 008H RETURN PARAMETERS
14280 0000440C [44470000]          <1>       dd     INIT_DRV           ; 009H
14281 00004410 [A3470000]          <1>       dd     RD_LONG            ; 00AH
14282 00004414 [AC470000]          <1>       dd     WR_LONG            ; 00BH
14283 00004418 [B5470000]          <1>       dd     DISK_SEEK          ; 00CH
14284 0000441C [AB450000]          <1>       dd     DISK_RESET         ; 00DH
14285 00004420 [A1450000]          <1>       dd     BAD_COMMAND        ; 00EH READ BUFFER
14286 00004424 [A1450000]          <1>       dd     BAD_COMMAND        ; 00FH WRITE BUFFER
14287 00004428 [DD470000]          <1>       dd     TST_RDY            ; 010H
14288 0000442C [01480000]          <1>       dd     HDISK_RECAL        ; 011H
14289 00004430 [A1450000]          <1>       dd     BAD_COMMAND        ; 012H MEMORY DIAGNOSTIC
14290 00004434 [A1450000]          <1>       dd     BAD_COMMAND        ; 013H DRIVE DIAGNOSTIC
14291 00004438 [37480000]          <1>       dd     CTLR_DIAGNOSTIC    ; 014H CONTROLLER DIAGNOSTIC
14292                              <1>       ; 02/02/2015 (Temporary - Retro UNIX 386 v1 - DISK I/O test)
14293 0000443C [A1450000]          <1>       dd     BAD_COMMAND        ; 015h
14294 00004440 [A1450000]          <1>       dd     BAD_COMMAND        ; 016h
14295 00004444 [A1450000]          <1>       dd     BAD_COMMAND        ; 017h
14296 00004448 [A1450000]          <1>       dd     BAD_COMMAND        ; 018h
14297 0000444C [A1450000]          <1>       dd     BAD_COMMAND        ; 019h
14298 00004450 [A1450000]          <1>       dd     BAD_COMMAND        ; 01Ah
14299 00004454 [2F460000]          <1>       dd     DISK_READ          ; 01Bh ; LBA read
14300 00004458 [38460000]          <1>       dd     DISK_WRITE         ; 01Ch ; LBA write
14301                              <1> D1L     EQU    $ - D1
14302                              <1>
14303                              <1> DISK_IO_CONT:
14304                              <1>       ;;CALL DDS                ; ESTABLISH SEGMENT
14305 0000445C 80FC01              <1>       CMP    AH,01H             ; RETURN STATUS
14306                              <1>       ;;JNZ  short SU0
14307                              <1>         ;;JMP    RETURN_STATUS
14308 0000445F 0F84BD010000        <1>       je     RETURN_STATUS
14309                              <1> SU0:
14310 00004465 C605[BB520100]00    <1>       MOV    byte [DISK_STATUS1],0     ; RESET THE STATUS INDICATOR
14311                              <1>       ;;PUSH BX                 ; SAVE DATA ADDRESS
14312                              <1>       ;mov   si, bx ;; 14/02/2015
14313 0000446C 89DE                <1>       mov    esi, ebx ; 21/02/2015
14314 0000446E 8A1D[BC520100]      <1>       MOV    BL,[HF_NUM]        ; GET NUMBER OF DRIVES
```

```
14315                                  <1>       ;; 04/01/2015
14316                                  <1>       ;;PUSH AX
14317 00004474 80E27F                  <1>       AND    DL,7FH              ; GET DRIVE AS 0 OR 1
14318                                  <1>                                  ; (get drive number as 0 to 3)
14319 00004477 38D3                    <1>       CMP    BL,DL
14320                                  <1>         ;;JBE   BAD_COMMAND_POP         ; INVALID DRIVE
14321 00004479 0F8622010000            <1>         jbe     BAD_COMMAND ;; 14/02/2015
14322                                  <1>         ;
14323                                  <1>       ;;03/01/2015
14324 0000447F 29DB                    <1>       sub    ebx, ebx
14325 00004481 88D3                    <1>       mov    bl, dl
14326                                  <1>       ;sub   bh, bh
14327 00004483 883D[D0520100]          <1>       mov    [LBAMode], bh       ; 0
14328                                  <1>       ;;test byte [bx+hd0_type], 1     ; LBA ready ?
14329                                  <1>       ;test  byte [ebx+hd0_type], 1
14330                                  <1>       ;jz    short su1            ; no
14331                                  <1>       ;inc   byte [LBAMode]
14332                                  <1> ;su1:
14333                                  <1>       ; 21/02/2015 (32 bit modification)
14334                                  <1>       ;04/01/2015
14335 00004489 6650                    <1>       push   ax ; ***
14336                                  <1>       ;PUSH  ES ; **
14337 0000448B 6652                    <1>       PUSH   DX ; *
14338 0000448D 6650                    <1>       push   ax
14339 0000448F E889060000              <1>       CALL   GET_VEC              ; GET DISK PARAMETERS
14340                                  <1>       ; 02/02/2015
14341                                  <1>       ;mov   ax, [ES:BX+16] ; I/O port base address (1F0h, 170h)
14342 00004494 668B4310                <1>       mov    ax, [ebx+16]
14343 00004498 66A3[E85C0000]          <1>       mov    [HF_PORT], ax
14344                                  <1>       ;mov   dx, [ES:BX+18] ; control port address (3F6h, 376h)
14345 0000449E 668B5312                <1>       mov    dx, [ebx+18]
14346 000044A2 668915[EA5C0000]        <1>       mov    [HF_REG_PORT], dx
14347                                  <1>       ;mov   al, [ES:BX+20] ; head register upper nibble (A0h,B0h,E0h,F0h)
14348 000044A9 8A4314                  <1>       mov    al, [ebx+20]
14349                                  <1>       ; 23/02/2015
14350 000044AC A840                    <1>       test   al, 40h       ; LBA bit (bit 6)
14351 000044AE 7406                    <1>       jz     short su1
14352 000044B0 FE05[D0520100]          <1>       inc    byte [LBAMode] ; 1
14353                                  <1> su1:
14354 000044B6 C0E804                  <1>       shr    al, 4
14355 000044B9 2401                    <1>       and    al, 1
14356 000044BB A2[EC5C0000]            <1>       mov    [hf_m_s], al
14357                                  <1>       ;
14358                                  <1>       ; 03/01/2015
14359                                  <1>       ;MOV   AL,byte [ES:BX+8]   ; GET CONTROL BYTE MODIFIER
14360 000044C0 8A4308                  <1>       mov    al, [ebx+8]
14361                                  <1>       ;MOV   DX,[HF_REG_PORT]    ; Device Control register
14362 000044C3 EE                      <1>       OUT    DX,AL               ; SET EXTRA HEAD OPTION
14363                                  <1>                                  ; Control Byte:  (= 08h, here)
14364                                  <1>                                  ; bit 0 - 0
14365                                  <1>                                  ; bit 1 - nIEN (1 = disable irq)
14366                                  <1>                                  ; bit 2 - SRST (software RESET)
14367                                  <1>                                  ; bit 3 - use extra heads (8 to 15)
14368                                  <1>                                  ;         -always set to 1-
14369                                  <1>                                  ; (bits 3 to 7 are reserved
14370                                  <1>                                  ;          for ATA devices)
14371 000044C4 8A25[BD520100]          <1>       MOV    AH,[CONTROL_BYTE]   ; SET EXTRA HEAD OPTION IN
14372 000044CA 80E4C0                  <1>       AND    AH,0C0H             ; CONTROL BYTE
14373 000044CD 08C4                    <1>       OR     AH,AL
14374 000044CF 8825[BD520100]          <1>       MOV    [CONTROL_BYTE],AH
14375                                  <1>       ; 04/01/2015
14376 000044D5 6658                    <1>       pop    ax
14377 000044D7 665A                    <1>       pop    dx ; * ;; 14/02/2015
14378 000044D9 20E4                    <1>       and    ah, ah ; Reset function ?
14379 000044DB 7507                    <1>       jnz    short su2
14380                                  <1>       ;;pop  dx ; * ;; 14/02/2015
14381                                  <1>       ;pop   es ; **
14382 000044DD 6658                    <1>       pop    ax ; ***
14383                                  <1>       ;;pop  bx
14384 000044DF E9C7000000              <1>         jmp     DISK_RESET
14385                                  <1> su2:
14386 000044E4 803D[D0520100]00        <1>       cmp    byte [LBAMode], 0
14387 000044EB 7662                    <1>       jna    short su3
14388                                  <1>       ;
14389                                  <1>       ; 02/02/2015 (LBA read/write function calls)
14390 000044ED 80FC1B                  <1>       cmp    ah, 1Bh
14391 000044F0 720B                    <1>       jb     short lbarw1
14392 000044F2 80FC1C                  <1>       cmp    ah, 1Ch
14393 000044F5 775D                    <1>       ja     short invldfnc
14394                                  <1>       ;;pop  dx ; * ; 14/02/2015
14395                                  <1>       ;mov   ax, cx ; Lower word of LBA address (bits 0-15)
14396 000044F7 89C8                    <1>       mov    eax, ecx ; LBA address (21/02/2015)
14397                                  <1>       ;; 14/02/2015
14398 000044F9 88D1                    <1>       mov    cl, dl ; 14/02/2015
14399                                  <1>       ;;mov  dx, bx
14400                                  <1>       ;mov   dx, si ; higher word of LBA address (bits 16-23)
14401                                  <1>       ;;mov  bx, di
14402                                  <1>       ;mov   si, di ; Buffer offset
14403 000044FB EB32                    <1>       jmp    short lbarw2
14404                                  <1> lbarw1:
14405                                  <1>       ; convert CHS to LBA
14406                                  <1>       ;
14407                                  <1>       ; LBA calculation - AWARD BIOS - 1999 - AHDSK.ASM
14408                                  <1>       ; LBA = "# of Heads" * Sectors/Track * Cylinder + Head * Sectors/Track
14409                                  <1>       ;      + Sector - 1
14410 000044FD 6652                    <1>       push   dx ; * ;; 14/02/2015
14411                                  <1>       ;xor   dh, dh
14412 000044FF 31D2                    <1>       xor    edx, edx
14413                                  <1>       ;mov   dl, [ES:BX+14]      ; sectors per track (logical)
14414 00004501 8A530E                  <1>       mov    dl, [ebx+14]
14415                                  <1>       ;xor   ah, ah
14416 00004504 31C0                    <1>       xor    eax, eax
14417                                  <1>       ;mov   al, [ES:BX+2]; heads (logical)
```

```
14418 00004506 8A4302              <1>       mov    al, [ebx+2]
14419 00004509 FEC8                <1>       dec    al
14420 0000450B 6640                <1>       inc    ax           ; 0 =  256
14421 0000450D 66F7E2              <1>       mul    dx
14422                              <1>              ; AX = # of Heads" * Sectors/Track
14423 00004510 6689CA              <1>       mov    dx, cx
14424                              <1>       ;and   cx, 3Fh       ; sector  (1 to 63)
14425 00004513 83E13F              <1>       and    ecx, 3fh
14426 00004516 86D6                <1>       xchg   dl, dh
14427 00004518 C0EE06              <1>       shr    dh, 6
14428                              <1>              ; DX = cylinder (0 to 1023)
14429                              <1>       ;mul   dx
14430                              <1>              ; DX:AX = # of Heads" * Sectors/Track * Cylinder
14431 0000451B F7E2                <1>       mul    edx
14432 0000451D FEC9                <1>       dec    cl  ; sector - 1
14433                              <1>       ;add   ax, cx
14434                              <1>       ;adc   dx, 0
14435                              <1>              ; DX:AX = # of Heads" * Sectors/Track * Cylinder + Sector -1
14436 0000451F 01C8                <1>       add    eax, ecx
14437 00004521 6659                <1>       pop    cx ; * ; ch = head, cl = drive number (zero based)
14438                              <1>       ;push  dx
14439                              <1>       ;push  ax
14440 00004523 50                  <1>       push   eax
14441                              <1>       ;mov   al, [ES:BX+14]      ; sectors per track (logical)
14442 00004524 8A430E              <1>       mov    al, [ebx+14]
14443 00004527 F6E5                <1>       mul    ch
14444                              <1>              ;  AX = Head * Sectors/Track
14445 00004529 0FB7C0              <1>         movzx      eax, ax ; 09/12/2017
14446                              <1>       ;pop   dx
14447 0000452C 5A                  <1>       pop    edx
14448                              <1>       ;add   ax, dx
14449                              <1>       ;pop   dx
14450                              <1>       ;adc   dx, 0 ; add carry bit
14451 0000452D 01D0                <1>       add    eax, edx
14452                              <1> lbarw2:
14453 0000452F 29D2                <1>       sub    edx, edx ; 21/02/2015
14454 00004531 88CA                <1>       mov    dl, cl ; 21/02/2015
14455 00004533 C645F800            <1>         mov     byte [CMD_BLOCK], 0 ; Features Register
14456                              <1>                             ; NOTE: Features register (1F1h, 171h)
14457                              <1>                             ; is not used for ATA device R/W functions.
14458                              <1>                             ; It is old/obsolete 'write precompensation'
14459                              <1>                             ; register and error register
14460                              <1>                             ; for old ATA/IDE devices.
14461                              <1>       ; 18/01/2014
14462                              <1>       ;mov   ch, [hf_m_s] ; Drive 0 (master) or 1 (slave)
14463 00004537 8A0D[EC5C0000]      <1>       mov    cl, [hf_m_s]
14464                              <1>       ;shl   ch, 4        ; bit 4 (drive bit)
14465                              <1>       ;or    ch, 0E0h     ; bit 5 = 1
14466                              <1>                             ; bit 6 = 1 = LBA mode
14467                              <1>                             ; bit 7 = 1
14468 0000453D 80C90E              <1>       or     cl, 0Eh ; 1110b
14469                              <1>       ;and   dh, 0Fh             ; LBA byte 4 (bits 24 to 27)
14470 00004540 25FFFFFF0F          <1>       and    eax, 0FFFFFFFh
14471 00004545 C1E11C              <1>       shl    ecx, 28 ; 21/02/2015
14472                              <1>       ;or    dh, ch
14473 00004548 09C8                <1>       or     eax, ecx
14474                              <1>       ;;mov  [CMD_BLOCK+2], al ; LBA byte 1 (bits 0 to 7)
14475                              <1>                                ; (Sector Number Register)
14476                              <1>       ;;mov  [CMD_BLOCK+3], ah ; LBA byte 2 (bits 8 to 15)
14477                              <1>                                ; (Cylinder Low Register)
14478                              <1>       ;mov   [CMD_BLOCK+2], ax ; LBA byte 1, 2
14479                              <1>       ;mov   [CMD_BLOCK+4], dl ; LBA byte 3 (bits 16 to 23)
14480                              <1>                                ; (Cylinder High Register)
14481                              <1>       ;;mov  [CMD_BLOCK+5], dh ; LBA byte 4 (bits 24 to 27)
14482                              <1>                                ; (Drive/Head Register)
14483                              <1>
14484                              <1>       ;mov   [CMD_BLOCK+4], dx ; LBA byte 4, LBA & DEV select bits
14485 0000454A 8945FA              <1>       mov    [CMD_BLOCK+2], eax ; 21/02/2015
14486                              <1>       ;14/02/2015
14487                              <1>       ;mov   dl, cl ; Drive number (INIT_DRV)
14488 0000454D EB38                <1>       jmp    short su4
14489                              <1> su3:
14490                              <1>       ; 02/02/2015
14491                              <1>       ; (Temporary functions 1Bh & 1Ch are not valid for CHS mode)
14492 0000454F 80FC14              <1>       cmp    ah, 14h
14493 00004552 7604                <1>       jna    short chsfnc
14494                              <1> invldfnc:
14495                              <1>         ; 14/02/2015
14496                              <1>       ;pop   es ; **
14497 00004554 6658                <1>         pop     ax ; ***
14498                              <1>         ;jmp     short BAD_COMMAND_POP
14499 00004556 EB49                <1>         jmp     short BAD_COMMAND
14500                              <1> chsfnc:
14501                              <1>       ;MOV   AX,[ES:BX+5]        ; GET WRITE PRE-COMPENSATION CYLINDER
14502 00004558 668B4305            <1>       mov    ax, [ebx+5]
14503 0000455C 66C1E802            <1>       SHR    AX,2
14504 00004560 8845F8              <1>       MOV    [CMD_BLOCK],AL
14505                              <1>       ;;MOV  AL,[ES:BX+8]        ; GET CONTROL BYTE MODIFIER
14506                              <1>       ;;PUSH DX
14507                              <1>       ;;MOV  DX,[HF_REG_PORT]
14508                              <1>       ;;OUT  DX,AL               ; SET EXTRA HEAD OPTION
14509                              <1>       ;;POP  DX ; *
14510                              <1>       ;;POP  ES ; **
14511                              <1>       ;;MOV  AH,[CONTROL_BYTE]   ; SET EXTRA HEAD OPTION IN
14512                              <1>       ;;AND  AH,0C0H             ; CONTROL BYTE
14513                              <1>       ;;OR   AH,AL
14514                              <1>       ;;MOV  [CONTROL_BYTE],AH
14515                              <1>       ;
14516 00004563 88C8                <1>       MOV    AL,CL               ; GET SECTOR NUMBER
14517 00004565 243F                <1>       AND    AL,3FH
14518 00004567 8845FA              <1>       MOV    [CMD_BLOCK+2],AL
14519 0000456A 886DFB              <1>       MOV    [CMD_BLOCK+3],CH    ; GET CYLINDER NUMBER
14520 0000456D 88C8                <1>       MOV    AL,CL
```

```
14521 0000456F C0E806              <1>       SHR    AL,6
14522 00004572 8845FC              <1>       MOV    [CMD_BLOCK+4],AL    ; CYLINDER HIGH ORDER 2 BITS
14523                              <1>       ;;05/01/2015
14524                              <1>       ;;MOV  AL,DL               ; DRIVE NUMBER
14525 00004575 A0[EC5C0000]        <1>       mov    al, [hf_m_s]
14526 0000457A C0E004              <1>       SHL    AL,4
14527 0000457D 80E60F              <1>       AND    DH,0FH              ; HEAD NUMBER
14528 00004580 08F0                <1>       OR     AL,DH
14529                              <1>       ;OR    AL,80H or 20H
14530 00004582 0CA0                <1>       OR     AL,80h+20h          ; ECC AND 512 BYTE SECTORS
14531 00004584 8845FD              <1>       MOV    [CMD_BLOCK+5],AL    ; ECC/SIZE/DRIVE/HEAD
14532                              <1> su4:
14533                              <1>       ;POP   ES ; **
14534                              <1>         ;; 14/02/2015
14535                              <1>         ;;POP   AX
14536                              <1>         ;;MOV   [CMD_BLOCK+1],AL        ; SECTOR COUNT
14537                              <1>         ;;PUSH  AX
14538                              <1>         ;;MOV   AL,AH                   ; GET INTO LOW BYTE
14539                              <1>         ;;XOR   AH,AH                   ; ZERO HIGH BYTE
14540                              <1>         ;;SAL   AX,1                    ; *2 FOR TABLE LOOKUP
14541 00004587 6658                <1>         pop     ax ; ***
14542 00004589 8845F9              <1>         mov     [CMD_BLOCK+1], al
14543 0000458C 29DB                <1>         sub  ebx, ebx
14544 0000458E 88E3                <1>       mov     bl, ah
14545                              <1>         ;xor     bh, bh
14546                              <1>         ;sal     bx, 1
14547 00004590 66C1E302            <1>         sal  bx, 2  ; 32 bit offset (21/02/2015)
14548                              <1>       ;;MOV   SI,AX                   ; PUT INTO SI FOR BRANCH
14549                              <1>         ;;CMP   AX,D1L                  ; TEST WITHIN RANGE
14550                              <1>         ;;JNB   short BAD_COMMAND_POP
14551                              <1>         ;cmp     bx, D1L
14552 00004594 83FB74              <1>       cmp    ebx, D1L
14553 00004597 7308                <1>       jnb    short BAD_COMMAND
14554                              <1>         ;xchg    bx, si
14555 00004599 87DE                <1>         xchg ebx, esi
14556                              <1>       ;;;POP AX                 ; RESTORE AX
14557                              <1>       ;;;POP BX                 ; AND DATA ADDRESS
14558                              <1>
14559                              <1>       ;;PUSH CX
14560                              <1>       ;;PUSH AX                 ; ADJUST ES:BX
14561                              <1>       ;MOV   CX,BX              ; GET 3 HIGH ORDER NIBBLES OF BX
14562                              <1>       ;SHR   CX,4
14563                              <1>       ;MOV   AX,ES
14564                              <1>       ;ADD   AX,CX
14565                              <1>       ;MOV   ES,AX
14566                              <1>       ;AND   BX,000FH           ; ES:BX CHANGED TO ES:000X
14567                              <1>       ;;POP  AX
14568                              <1>       ;;POP  CX
14569                              <1>       ;;JMP  word [CS:SI+D1]
14570                              <1>       ;jmp   word [SI+D1]
14571 0000459B FFA6[E8430000]      <1>       jmp    dword [esi+D1]
14572                              <1> ;;BAD_COMMAND_POP:
14573                              <1> ;;    POP    AX
14574                              <1> ;;    POP    BX
14575                              <1> BAD_COMMAND:
14576 000045A1 C605[BB520100]01    <1>         MOV     byte [DISK_STATUS1],BAD_CMD  ; COMMAND ERROR
14577 000045A8 B000                <1>       MOV    AL,0
14578 000045AA C3                  <1>       RETn
14579                              <1>
14580                              <1> ;---------------------------------------
14581                              <1> ;     RESET THE DISK SYSTEM  (AH=00H) :
14582                              <1> ;---------------------------------------
14583                              <1>
14584                              <1> ; 18-1-2015 : one controller reset (not other one)
14585                              <1>
14586                              <1> DISK_RESET:
14587 000045AB FA                  <1>       CLI
14588 000045AC E4A1                <1>       IN     AL,INTB01          ; GET THE MASK REGISTER
14589                              <1>       ;JMP   $+2
14590                              <1>       IODELAY
14591 000045AE EB00                <2>  jmp short $+2
14592 000045B0 EB00                <2>  jmp short $+2
14593                              <1>       ;AND   AL,0BFH            ; ENABLE FIXED DISK INTERRUPT
14594 000045B2 243F                <1>       and    al,3Fh             ; 22/12/2014 (IRQ 14 & IRQ 15)
14595 000045B4 E6A1                <1>       OUT    INTB01,AL
14596 000045B6 FB                  <1>       STI                       ; START INTERRUPTS
14597                              <1>       ; 14/02/2015
14598 000045B7 6689D7              <1>       mov    di, dx
14599                              <1>       ; 04/01/2015
14600                              <1>       ;xor   di,di
14601                              <1> drst0:
14602 000045BA B004                <1>       MOV    AL,04H  ; bit 2 - SRST
14603                              <1>       ;MOV   DX,HF_REG_PORT
14604 000045BC 668B15[EA5C0000]    <1>       MOV    DX,[HF_REG_PORT]
14605 000045C3 EE                  <1>       OUT    DX,AL              ; RESET
14606                              <1> ;     MOV    CX,10              ; DELAY COUNT
14607                              <1> ;DRD: DEC    CX
14608                              <1> ;     JNZ    short DRD          ; WAIT 4.8 MICRO-SEC
14609                              <1>       ;mov   cx,2               ; wait for 30 micro seconds
14610 000045C4 B902000000          <1>         mov  ecx, 2 ; 21/02/2015
14611 000045C9 E81FD8FFFF          <1>       call    WAITF                   ; (Award Bios 1999 - WAIT_REFRESH,
14612                              <1>                                         ; 40 micro seconds)
14613 000045CE A0[BD520100]        <1>       mov    al,[CONTROL_BYTE]
14614 000045D3 240F                <1>       AND    AL,0FH             ; SET HEAD OPTION
14615 000045D5 EE                  <1>       OUT    DX,AL              ; TURN RESET OFF
14616 000045D6 E838040000          <1>       CALL   NOT_BUSY
14617 000045DB 7515                <1>       JNZ    short DRERR        ; TIME OUT ON RESET
14618 000045DD 668B15[E85C0000]    <1>       MOV    DX,[HF_PORT]
14619 000045E4 FEC2                <1>       inc    dl  ; HF_PORT+1
14620                              <1>       ; 02/01/2015 - Award BIOS 1999 - AHDSK.ASM
14621                              <1>         ;mov     cl, 10
14622 000045E6 B90A000000          <1>         mov     ecx, 10 ; 21/02/2015
14623                              <1> drst1:
```

```
14624 000045EB EC                  <1>       IN     AL,DX              ; GET RESET STATUS
14625 000045EC 3C01                <1>       CMP    AL,1
14626                              <1>       ; 04/01/2015
14627 000045EE 740A                <1>       jz     short drst2
14628                              <1>       ;JNZ   short DRERR        ; BAD RESET STATUS
14629                              <1>              ; Drive/Head Register - bit 4
14630 000045F0 E2F9                <1>       loop   drst1
14631                              <1> DRERR:
14632 000045F2 C605[BB520100]05    <1>       MOV    byte [DISK_STATUS1],BAD_RESET ; CARD FAILED
14633 000045F9 C3                  <1>       RETn
14634                              <1> drst2:
14635                              <1>       ; 14/02/2015
14636 000045FA 6689FA              <1>       mov    dx,di
14637                              <1> ;drst3:
14638                              <1> ;     ; 05/01/2015
14639                              <1> ;     shl    di,1
14640                              <1> ;     ; 04/01/2015
14641                              <1> ;     mov    ax,[di+hd_cports]
14642                              <1> ;     cmp    ax,[HF_REG_PORT]
14643                              <1> ;     je     short drst4
14644                              <1> ;     mov    [HF_REG_PORT], ax
14645                              <1> ;     ; 03/01/2015
14646                              <1> ;     mov    ax,[di+hd_ports]
14647                              <1> ;       mov     [HF_PORT], ax
14648                              <1> ;     ; 05/01/2014
14649                              <1> ;     shr    di,1
14650                              <1> ;     ; 04/01/2015
14651                              <1> ;     jmp    short drst0  ; reset other controller
14652                              <1> ;drst4:
14653                              <1> ;     ; 05/01/2015
14654                              <1> ;     shr    di,1
14655                              <1> ;     mov    al,[di+hd_dregs]
14656                              <1> ;     and    al,10h ; bit 4 only
14657                              <1> ;     shr    al,4 ; bit 4  -> bit 0
14658                              <1> ;     mov    [hf_m_s], al ; (0 = master, 1 = slave)
14659                              <1>       ;
14660 000045FD A0[EC5C0000]        <1>       mov    al, [hf_m_s] ; 18/01/2015
14661 00004602 A801                <1>       test   al,1
14662                              <1> ;     jnz    short drst6
14663 00004604 7516                <1>         jnz     short drst4
14664 00004606 8065FDEF            <1>       AND     byte [CMD_BLOCK+5],0EFH ; SET TO DRIVE 0
14665                              <1> ;drst5:
14666                              <1> drst3:
14667 0000460A E835010000          <1>       CALL   INIT_DRV           ; SET MAX HEADS
14668                              <1>       ;mov   dx,di
14669 0000460F E8ED010000          <1>       CALL   HDISK_RECAL        ; RECAL TO RESET SEEK SPEED
14670                              <1>       ; 04/01/2014
14671                              <1> ;     inc    di
14672                              <1> ;     mov    dx,di
14673                              <1> ;     cmp    dl,[HF_NUM]
14674                              <1> ;     jb     short drst3
14675                              <1> ;DRE:
14676 00004614 C605[BB520100]00    <1>       MOV    byte [DISK_STATUS1],0     ; IGNORE ANY SET UP ERRORS
14677 0000461B C3                  <1>       RETn
14678                              <1> ;drst6:
14679                              <1> drst4:             ; Drive/Head Register - bit 4
14680 0000461C 804DFD10            <1>       OR      byte [CMD_BLOCK+5],010H ; SET TO DRIVE 1
14681                              <1>         ;jmp    short drst5
14682 00004620 EBE8                <1>         jmp     short drst3
14683                              <1>
14684                              <1> ;----------------------------------------
14685                              <1> ;     DISK STATUS ROUTINE  (AH = 01H) :
14686                              <1> ;----------------------------------------
14687                              <1>
14688                              <1> RETURN_STATUS:
14689 00004622 A0[BB520100]        <1>       MOV    AL,[DISK_STATUS1]   ; OBTAIN PREVIOUS STATUS
14690 00004627 C605[BB520100]00    <1>         MOV     byte [DISK_STATUS1],0   ; RESET STATUS
14691 0000462E C3                  <1>       RETn
14692                              <1>
14693                              <1> ;----------------------------------------
14694                              <1> ;     DISK READ ROUTINE    (AH = 02H) :
14695                              <1> ;----------------------------------------
14696                              <1>
14697                              <1> DISK_READ:
14698 0000462F C645FE20            <1>       MOV    byte [CMD_BLOCK+6],READ_CMD
14699 00004633 E954020000          <1>         JMP     COMMANDI
14700                              <1>
14701                              <1> ;----------------------------------------
14702                              <1> ;     DISK WRITE ROUTINE   (AH = 03H) :
14703                              <1> ;----------------------------------------
14704                              <1>
14705                              <1> DISK_WRITE:
14706 00004638 C645FE30            <1>       MOV    byte [CMD_BLOCK+6],WRITE_CMD
14707 0000463C E9A6020000          <1>         JMP     COMMANDO
14708                              <1>
14709                              <1> ;----------------------------------------
14710                              <1> ;     DISK VERIFY       (AH = 04H) :
14711                              <1> ;----------------------------------------
14712                              <1>
14713                              <1> DISK_VERF:
14714 00004641 C645FE40            <1>       MOV    byte [CMD_BLOCK+6],VERIFY_CMD
14715 00004645 E814030000          <1>       CALL   COMMAND
14716 0000464A 750C                <1>       JNZ    short VERF_EXIT           ; CONTROLLER STILL BUSY
14717 0000464C E886030000          <1>       CALL   _WAIT               ; (Original: CALL WAIT)
14718 00004651 7505                <1>       JNZ    short VERF_EXIT           ; TIME OUT
14719 00004653 E813040000          <1>       CALL   CHECK_STATUS
14720                              <1> VERF_EXIT:
14721 00004658 C3                  <1>       RETn
14722                              <1>
14723                              <1> ;----------------------------------------
14724                              <1> ;     FORMATTING        (AH = 05H) :
14725                              <1> ;----------------------------------------
14726                              <1>
```

```
14727                                <1> FMT_TRK:                        ; FORMAT TRACK      (AH = 005H)
14728 00004659 C645FE50              <1>       MOV    byte [CMD_BLOCK+6],FMTTRK_CMD
14729                                <1>       ;PUSH  ES
14730                                <1>       ;PUSH  BX
14731 0000465D 53                    <1>       push   ebx
14732 0000465E E8BA040000            <1>       CALL   GET_VEC              ; GET DISK PARAMETERS ADDRESS
14733                                <1>       ;MOV   AL,[ES:BX+14]        ; GET SECTORS/TRACK
14734 00004663 8A430E                <1>       mov    al, [ebx+14]
14735 00004666 8845F9                <1>       MOV    [CMD_BLOCK+1],AL     ; SET SECTOR COUNT IN COMMAND
14736 00004669 5B                    <1>       pop    ebx
14737                                <1>       ;POP   BX
14738                                <1>       ;POP   ES
14739 0000466A E97F020000            <1>         JMP     CMD_OF                  ; GO EXECUTE THE COMMAND
14740                                <1>
14741                                <1> ;---------------------------------------
14742                                <1> ;     READ DASD TYPE           (AH = 15H) :
14743                                <1> ;---------------------------------------
14744                                <1>
14745                                <1> READ_DASD_TYPE:
14746                                <1> READ_D_T:                       ; GET DRIVE PARAMETERS
14747 0000466F 1E                    <1>       PUSH   DS                   ; SAVE REGISTERS
14748                                <1>       ;PUSH  ES
14749 00004670 53                    <1>       PUSH   eBX
14750                                <1>       ;CALL  DDS                  ; ESTABLISH ADDRESSING
14751                                <1>       ;push  cs
14752                                <1>       ;pop   ds
14753 00004671 66BB1000              <1>         mov  bx, KDATA
14754 00004675 8EDB                  <1>       mov    ds, bx
14755                                <1>       ;mov   es, bx
14756 00004677 C605[BB520100]00      <1>       MOV     byte [DISK_STATUS1],0
14757 0000467E 8A1D[BC520100]        <1>       MOV    BL,[HF_NUM]          ; GET NUMBER OF DRIVES
14758 00004684 80E27F                <1>       AND    DL,7FH               ; GET DRIVE NUMBER
14759 00004687 38D3                  <1>       CMP    BL,DL
14760 00004689 7627                  <1>       JBE    short RDT_NOT_PRESENT     ; RETURN DRIVE NOT PRESENT
14761 0000468B E88D040000            <1>       CALL   GET_VEC             ; GET DISK PARAMETER ADDRESS
14762                                <1>       ;MOV   AL,[ES:BX+2]         ; HEADS
14763 00004690 8A4302                <1>       mov    al, [ebx+2]
14764                                <1>       ;MOV   CL,[ES:BX+14]
14765 00004693 8A4B0E                <1>       mov    cl, [ebx+14]
14766 00004696 F6E9                  <1>       IMUL   CL                  ; * NUMBER OF SECTORS
14767                                <1>       ;MOV   CX,[ES:BX]           ; MAX NUMBER OF CYLINDERS
14768 00004698 668B0B                <1>       mov    cx ,[ebx]
14769                                <1>       ;
14770                                <1>       ; 02/01/2015
14771                                <1>       ; ** leave the last cylinder as reserved for diagnostics **
14772                                <1>       ; (Also in Award BIOS - 1999, AHDSK.ASM, FUN15 -> sub ax, 1)
14773 0000469B 6649                  <1>       DEC    CX                  ; LEAVE ONE FOR DIAGNOSTICS
14774                                <1>       ;
14775 0000469D 66F7E9                <1>       IMUL   CX                  ; NUMBER OF SECTORS
14776 000046A0 6689D1                <1>       MOV    CX,DX               ; HIGH ORDER HALF
14777 000046A3 6689C2                <1>       MOV    DX,AX               ; LOW ORDER HALF
14778                                <1>       ;SUB   AX,AX
14779 000046A6 28C0                  <1>       sub    al, al
14780 000046A8 B403                  <1>       MOV    AH,03H              ; INDICATE FIXED DISK
14781 000046AA 5B                    <1> RDT2: POP    eBX                 ; RESTORE REGISTERS
14782                                <1>       ;POP   ES
14783 000046AB 1F                    <1>       POP    DS
14784                                <1>       ; (*) CLC                  ; CLEAR CARRY
14785                                <1>       ;RETf  2
14786                                <1>       ; (*) 29/05/2016
14787                                <1>       ; (*) retf 4
14788 000046AC 80642408FE            <1>       and    byte [esp+8], 0FEh ; clear carry bit of eflags register
14789 000046B1 CF                    <1>       iretd
14790                                <1>
14791                                <1> RDT_NOT_PRESENT:
14792 000046B2 6629C0                <1>       SUB    AX,AX               ; DRIVE NOT PRESENT RETURN
14793 000046B5 6689C1                <1>       MOV    CX,AX               ; ZERO BLOCK COUNT
14794 000046B8 6689C2                <1>       MOV    DX,AX
14795 000046BB EBED                  <1>       JMP    short RDT2
14796                                <1>
14797                                <1> ; 28/05/2016
14798                                <1> ; 27/05/2016 - TRDOS 386 (TRDOS v2.0)
14799                                <1>
14800                                <1> ;---------------------------------------
14801                                <1> ;     GET PARAMETERS           (AH = 08H) :
14802                                <1> ;---------------------------------------
14803                                <1>
14804                                <1> GET_PARM_N:
14805                                <1>       ; ebx = user's buffer address for parameters table
14806                                <1> ;GET_PARM:                      ; GET DRIVE PARAMETERS
14807 000046BD 1E                    <1>       PUSH   DS                   ; SAVE REGISTERS
14808 000046BE 06                    <1>       PUSH   ES
14809 000046BF 53                    <1>       PUSH   eBX
14810                                <1>       ;MOV   AX,ABS0              ; ESTABLISH ADDRESSING
14811                                <1>       ;MOV   DS,AX
14812                                <1>       ;TEST  DL,1                 ; CHECK FOR DRIVE 1
14813                                <1>       ;JZ    short G0
14814                                <1>       ;LES   BX,@HF1_TBL_VEC
14815                                <1>       ;JMP   SHORT G1
14816                                <1> ;G0:  LES    BX,@HF_TBL_VEC
14817                                <1> ;G1:
14818                                <1>       ;CALL  DDS                  ; ESTABLISH SEGMENT
14819                                <1>       ; 22/12/2014
14820                                <1>       ;push  cs
14821                                <1>       ;pop   ds
14822 000046C0 66BB1000             <1>       mov    bx, KDATA
14823 000046C4 8EDB                  <1>       mov    ds, bx
14824 000046C6 8EC3                  <1>       mov    es, bx ; 27/05/2016
14825                                <1>       ;
14826 000046C8 80EA80                <1>       SUB    DL,80H
14827 000046CB 80FA04                <1>       CMP    DL,MAX_FILE         ; TEST WITHIN RANGE
14828 000046CE 7361                  <1>       JAE    short G4
14829                                <1>       ;
```

```
14830 000046D0 31DB                <1>       xor    ebx, ebx ; 21/02/2015
14831                              <1>       ; 22/12/2014
14832 000046D2 88D3                <1>       mov    bl, dl
14833                              <1>       ;xor   bh, bh
14834 000046D4 C0E302              <1>       shl    bl, 2               ; convert index to offset
14835                              <1>       ;add   bx, HF_TBL_VEC
14836 000046D7 81C3[C0520100]      <1>       add    ebx, HF_TBL_VEC
14837                              <1>       ;mov   ax, [bx+2]
14838                              <1>       ;mov   es, ax              ; dpt segment
14839                              <1>       ;mov   bx, [bx]            ; dpt offset
14840 000046DD 8B1B                <1>       mov    ebx, [ebx] ; 32 bit offset
14841                              <1>
14842 000046DF C605[BB520100]00    <1>       MOV    byte [DISK_STATUS1],0
14843                              <1>         ;MOV     AX,[ES:BX]             ; MAX NUMBER OF CYLINDERS
14844 000046E6 668B03              <1>       mov    ax, [ebx]
14845                              <1>       ;;SUB  AX,2                ; ADJUST FOR 0-N
14846 000046E9 6648                <1>       dec    ax                  ; max. cylinder number
14847 000046EB 88C5                <1>       MOV    CH,AL
14848 000046ED 66250003            <1>       AND    AX,0300H            ; HIGH TWO BITS OF CYLINDER
14849 000046F1 66D1E8              <1>       SHR    AX,1
14850 000046F4 66D1E8              <1>       SHR    AX,1
14851                              <1>       ;OR    AL,[ES:BX+14]       ; SECTORS
14852 000046F7 0A430E              <1>       or     al, [ebx+14]
14853 000046FA 88C1                <1>       MOV    CL,AL
14854                              <1>       ;MOV   DH,[ES:BX+2]        ; HEADS
14855 000046FC 8A7302              <1>       mov    dh, [ebx+2]
14856 000046FF FECE                <1>       DEC    DH                  ; 0-N RANGE
14857 00004701 8A15[BC520100]      <1>       MOV    DL,[HF_NUM]         ; DRIVE COUNT
14858 00004707 6629C0              <1>       SUB    AX,AX
14859                              <1>         ;27/12/2014
14860                              <1>       ;mov   di, bx              ; HDPT offset
14861                              <1>
14862                              <1>       ; 27/05/2016
14863                              <1>       ; return fixed disk parameters table to user
14864                              <1>       ; in user's buffer, which is pointed by EBX
14865                              <1>       ;
14866 0000470A 873C24              <1>       xchg   edi, [esp]          ; ebx (input)-> edi, edi -> [esp]
14867 0000470D 56                  <1>       push   esi
14868 0000470E 89DE                <1>       mov    esi, ebx            ; hard disk parameter table (32 bytes)
14869 00004710 89FB                <1>       mov    ebx, edi            ; ebx = user's buffer address
14870 00004712 51                  <1>       push   ecx
14871 00004713 50                  <1>       push   eax
14872 00004714 B920000000          <1>       mov    ecx, 32 ; 32 bytes
14873 00004719 E88FA10000          <1>       call   transfer_to_user_buffer ; trdosk6.s (16/05/2016)
14874 0000471E 58                  <1>       pop    eax
14875 0000471F 59                  <1>       pop    ecx
14876 00004720 5E                  <1>       pop    esi
14877 00004721 5F                  <1>       pop    edi
14878 00004722 730A                <1>       jnc    short G5
14879                              <1>       ; 29/05/2016 (*)
14880 00004724 B8FF000000          <1>       mov    eax, 0FFh ; unknown error !
14881                              <1> _G6:
14882 00004729 804C241001          <1>       or     byte [esp+16], 1 ; set carry bit of eflags register
14883                              <1> G5:
14884                              <1>       ; 27/05/2016
14885                              <1>       ;POP   eBX                ; RESTORE REGISTERS
14886 0000472E 07                  <1>       POP    ES
14887 0000472F 1F                  <1>       POP    DS
14888                              <1>       ;RETf  2
14889                              <1>       ; (*) 29/05/2016
14890                              <1>       ; (*) retf 4
14891                              <1>       ; (*) or byte [esp+8], 1 ; set carry bit of eflags register
14892 00004730 CF                  <1>       iretd
14893                              <1> G4:
14894 00004731 C605[BB520100]07    <1>       MOV     byte [DISK_STATUS1],INIT_FAIL ; OPERATION FAILED
14895 00004738 B407                <1>       MOV    AH,INIT_FAIL
14896 0000473A 28C0                <1>       SUB    AL,AL
14897 0000473C 6629D2              <1>       SUB    DX,DX
14898 0000473F 6629C9              <1>       SUB    CX,CX
14899                              <1>       ; 29/05/2016 (*)
14900                              <1>       ;STC                       ; SET ERROR FLAG
14901                              <1>       ;JMP   short G5
14902 00004742 EBE5                <1>       jmp    short _G6
14903                              <1>
14904                              <1> ;---------------------------------------
14905                              <1> ;     INITIALIZE DRIVE     (AH = 09H) :
14906                              <1> ;---------------------------------------
14907                              <1>       ; 03/01/2015
14908                              <1>       ; According to ATA-ATAPI specification v2.0 to v5.0
14909                              <1>       ; logical sector per logical track
14910                              <1>       ; and logical heads - 1 would be set but
14911                              <1>       ; it is seen as it will be good
14912                              <1>       ; if physical parameters will be set here
14913                              <1>       ; because, number of heads <= 16.
14914                              <1>       ; (logical heads usually more than 16)
14915                              <1>       ; NOTE: ATA logical parameters (software C, H, S)
14916                              <1>       ;     == INT 13h physical parameters
14917                              <1>
14918                              <1> ;INIT_DRV:
14919                              <1> ;     MOV    byte [CMD_BLOCK+6],SET_PARM_CMD
14920                              <1> ;     CALL   GET_VEC             ; ES:BX -> PARAMETER BLOCK
14921                              <1> ;     MOV    AL,[ES:BX+2]        ; GET NUMBER OF HEADS
14922                              <1> ;     DEC    AL                  ; CONVERT TO 0-INDEX
14923                              <1> ;     MOV    AH,[CMD_BLOCK+5]    ; GET SDH REGISTER
14924                              <1> ;     AND    AH,0F0H             ; CHANGE HEAD NUMBER
14925                              <1> ;     OR     AH,AL               ; TO MAX HEAD
14926                              <1> ;     MOV    [CMD_BLOCK+5],AH
14927                              <1> ;     MOV    AL,[ES:BX+14]       ; MAX SECTOR NUMBER
14928                              <1> ;     MOV    [CMD_BLOCK+1],AL
14929                              <1> ;     SUB    AX,AX
14930                              <1> ;     MOV    [CMD_BLOCK+3],AL    ; ZERO FLAGS
14931                              <1> ;     CALL   COMMAND             ; TELL CONTROLLER
14932                              <1> ;     JNZ    short INIT_EXIT           ; CONTROLLER BUSY ERROR
```

```
14933                                  <1> ;     CALL   NOT_BUSY           ; WAIT FOR IT TO BE DONE
14934                                  <1> ;     JNZ    short INIT_EXIT           ; TIME OUT
14935                                  <1> ;     CALL   CHECK_STATUS
14936                                  <1> ;INIT_EXIT:
14937                                  <1> ;     RETn
14938                                  <1>
14939                                  <1> ; 04/01/2015
14940                                  <1> ; 02/01/2015 - Derived from from AWARD BIOS 1999
14941                                  <1> ;                          AHDSK.ASM - INIT_DRIVE
14942                                  <1> INIT_DRV:
14943                                  <1>       ;xor   ah,ah
14944 00004744 31C0                   <1>       xor    eax, eax ; 21/02/2015
14945 00004746 B00B                   <1>       mov    al,11 ; Physical heads from translated HDPT
14946 00004748 3825[D0520100]         <1>         cmp     [LBAMode], ah   ; 0
14947 0000474E 7702                   <1>       ja     short idrv0
14948 00004750 B002                   <1>       mov    al,2  ; Physical heads from standard HDPT
14949                                  <1> idrv0:
14950                                  <1>       ; DL = drive number (0 based)
14951 00004752 E8C6030000             <1>       call   GET_VEC
14952                                  <1>       ;push  bx
14953 00004757 53                     <1>       push   ebx ; 21/02/2015
14954                                  <1>       ;add   bx,ax
14955 00004758 01C3                   <1>       add    ebx, eax
14956                                  <1>       ;; 05/01/2015
14957 0000475A 8A25[EC5C0000]         <1>       mov    ah, [hf_m_s] ; drive number (0= master, 1= slave)
14958                                  <1>       ;;and  ah,1
14959 00004760 C0E404                 <1>       shl    ah,4
14960 00004763 80CCA0                 <1>       or     ah,0A0h  ; Drive/Head register - 10100000b (A0h)
14961                                  <1>       ;mov   al,[es:bx]
14962 00004766 8A03                   <1>       mov    al, [ebx] ; 21/02/2015
14963 00004768 FEC8                   <1>       dec    al      ; last head number
14964                                  <1>       ;and   al,0Fh
14965 0000476A 08E0                   <1>       or     al,ah   ; lower 4 bits for head number
14966                                  <1>       ;
14967 0000476C C645FE91               <1>       mov    byte [CMD_BLOCK+6],SET_PARM_CMD
14968 00004770 8845FD                 <1>       mov    [CMD_BLOCK+5],al
14969                                  <1>       ;pop   bx
14970 00004773 5B                     <1>       pop    ebx
14971 00004774 29C0                   <1>       sub    eax, eax ; 21/02/2015
14972 00004776 B004                   <1>       mov    al,4 ; Physical sec per track from translated HDPT
14973 00004778 803D[D0520100]00       <1>       cmp    byte [LBAMode], 0
14974 0000477F 7702                   <1>       ja     short idrv1
14975 00004781 B00E                   <1>       mov    al,14 ; Physical sec per track from standard HDPT
14976                                  <1> idrv1:
14977                                  <1>       ;xor   ah,ah
14978                                  <1>       ;add   bx,ax
14979 00004783 01C3                   <1>       add    ebx, eax ; 21/02/2015
14980                                  <1>       ;mov   al,[es:bx]
14981                                  <1>                      ; sector number
14982 00004785 8A03                   <1>       mov    al, [ebx]
14983 00004787 8845F9                 <1>       mov    [CMD_BLOCK+1],al
14984 0000478A 28C0                   <1>       sub    al,al
14985 0000478C 8845FB                 <1>       mov    [CMD_BLOCK+3],al  ; ZERO FLAGS
14986 0000478F E8CA010000             <1>       call   COMMAND        ; TELL CONTROLLER
14987 00004794 750C                   <1>       jnz    short INIT_EXIT       ; CONTROLLER BUSY ERROR
14988 00004796 E878020000             <1>       call   NOT_BUSY       ; WAIT FOR IT TO BE DONE
14989 0000479B 7505                   <1>       jnz    short INIT_EXIT       ; TIME OUT
14990 0000479D E8C9020000             <1>       call   CHECK_STATUS
14991                                  <1> INIT_EXIT:
14992 000047A2 C3                     <1>       RETn
14993                                  <1>
14994                                  <1> ;---------------------------------------
14995                                  <1> ;     READ LONG         (AH = 0AH) :
14996                                  <1> ;---------------------------------------
14997                                  <1>
14998                                  <1> RD_LONG:
14999                                  <1>       ;MOV   @CMD_BLOCK+6,READ_CMD OR ECC_MODE
15000 000047A3 C645FE22               <1>         mov     byte [CMD_BLOCK+6],READ_CMD + ECC_MODE
15001 000047A7 E9E0000000             <1>         JMP     COMMANDI
15002                                  <1>
15003                                  <1> ;---------------------------------------
15004                                  <1> ;     WRITE LONG        (AH = 0BH) :
15005                                  <1> ;---------------------------------------
15006                                  <1>
15007                                  <1> WR_LONG:
15008                                  <1>       ;MOV   @CMD_BLOCK+6,WRITE_CMD OR ECC_MODE
15009 000047AC C645FE32               <1>         MOV     byte [CMD_BLOCK+6],WRITE_CMD + ECC_MODE
15010 000047B0 E932010000             <1>         JMP     COMMANDO
15011                                  <1>
15012                                  <1> ;---------------------------------------
15013                                  <1> ;     SEEK              (AH = 0CH) :
15014                                  <1> ;---------------------------------------
15015                                  <1>
15016                                  <1> DISK_SEEK:
15017 000047B5 C645FE70               <1>         MOV     byte [CMD_BLOCK+6],SEEK_CMD
15018 000047B9 E8A0010000             <1>       CALL   COMMAND
15019 000047BE 751C                   <1>       JNZ    short DS_EXIT             ; CONTROLLER BUSY ERROR
15020 000047C0 E812020000             <1>       CALL   _WAIT
15021 000047C5 7515                   <1>         JNZ     DS_EXIT                 ; TIME OUT ON SEEK
15022 000047C7 E89F020000             <1>       CALL   CHECK_STATUS
15023 000047CC 803D[BB520100]40       <1>         CMP     byte [DISK_STATUS1],BAD_SEEK
15024 000047D3 7507                   <1>       JNE    short DS_EXIT
15025 000047D5 C605[BB520100]00       <1>         MOV     byte [DISK_STATUS1],0
15026                                  <1> DS_EXIT:
15027 000047DC C3                     <1>       RETn
15028                                  <1>
15029                                  <1> ;---------------------------------------
15030                                  <1> ;     TEST DISK READY      (AH = 10H) :
15031                                  <1> ;---------------------------------------
15032                                  <1>
15033                                  <1> TST_RDY:                        ; WAIT FOR CONTROLLER
15034 000047DD E831020000             <1>       CALL   NOT_BUSY
15035 000047E2 751C                   <1>       JNZ    short TR_EX
```

```
15036 000047E4 8A45FD              <1>       MOV    AL,[CMD_BLOCK+5]    ; SELECT DRIVE
15037 000047E7 668B15[E85C0000]    <1>       MOV    DX,[HF_PORT]
15038 000047EE 80C206              <1>       add    dl,6
15039 000047F1 EE                  <1>       OUT    DX,AL
15040 000047F2 E88C020000          <1>       CALL   CHECK_ST            ; CHECK STATUS ONLY
15041 000047F7 7507                <1>       JNZ    short TR_EX
15042 000047F9 C605[BB520100]00    <1>       MOV    byte [DISK_STATUS1],0     ; WIPE OUT DATA CORRECTED ERROR
15043                              <1> TR_EX:
15044 00004800 C3                  <1>       RETn
15045                              <1>
15046                              <1> ;----------------------------------------
15047                              <1> ;     RECALIBRATE       (AH = 11H) :
15048                              <1> ;----------------------------------------
15049                              <1>
15050                              <1> HDISK_RECAL:
15051 00004801 C645FE10            <1>         MOV     byte [CMD_BLOCK+6],RECAL_CMD ; 10h, 16
15052 00004805 E854010000          <1>       CALL   COMMAND             ; START THE OPERATION
15053 0000480A 7523                <1>       JNZ    short RECAL_EXIT    ; ERROR
15054 0000480C E8C6010000          <1>       CALL   _WAIT               ; WAIT FOR COMPLETION
15055 00004811 7407                <1>       JZ     short RECAL_X             ; TIME OUT ONE OK ?
15056 00004813 E8BF010000          <1>       CALL   _WAIT               ; WAIT FOR COMPLETION LONGER
15057 00004818 7515                <1>       JNZ    short RECAL_EXIT    ; TIME OUT TWO TIMES IS ERROR
15058                              <1> RECAL_X:
15059 0000481A E84C020000          <1>       CALL   CHECK_STATUS
15060 0000481F 803D[BB520100]40    <1>       CMP    byte [DISK_STATUS1],BAD_SEEK ; SEEK NOT COMPLETE
15061 00004826 7507                <1>       JNE    short RECAL_EXIT    ; IS OK
15062 00004828 C605[BB520100]00    <1>       MOV    byte [DISK_STATUS1],0
15063                              <1> RECAL_EXIT:
15064 0000482F 803D[BB520100]00    <1>         CMP     byte [DISK_STATUS1],0
15065 00004836 C3                  <1>       RETn
15066                              <1>
15067                              <1> ;----------------------------------------
15068                              <1> ;      CONTROLLER DIAGNOSTIC (AH = 14H) :
15069                              <1> ;----------------------------------------
15070                              <1>
15071                              <1> CTLR_DIAGNOSTIC:
15072 00004837 FA                  <1>         CLI                             ; DISABLE INTERRUPTS WHILE CHANGING MASK
15073 00004838 E4A1                <1>       IN     AL,INTB01          ; TURN ON SECOND INTERRUPT CHIP
15074                              <1>       ;AND   AL,0BFH
15075 0000483A 243F                <1>       and    al, 3Fh                  ; enable IRQ 14 & IRQ 15
15076                              <1>       ;JMP   $+2
15077                              <1>       IODELAY
15078 0000483C EB00                <2>  jmp short $+2
15079 0000483E EB00                <2>  jmp short $+2
15080 00004840 E6A1                <1>       OUT    INTB01,AL
15081                              <1>       IODELAY
15082 00004842 EB00                <2>  jmp short $+2
15083 00004844 EB00                <2>  jmp short $+2
15084 00004846 E421                <1>       IN     AL,INTA01          ; LET INTERRUPTS PASS THRU TO
15085 00004848 24FB                <1>       AND    AL,0FBH            ;  SECOND CHIP
15086                              <1>       ;JMP   $+2
15087                              <1>       IODELAY
15088 0000484A EB00                <2>  jmp short $+2
15089 0000484C EB00                <2>  jmp short $+2
15090 0000484E E621                <1>       OUT    INTA01,AL
15091 00004850 FB                  <1>       STI
15092 00004851 E8BD010000          <1>       CALL   NOT_BUSY           ; WAIT FOR CARD
15093 00004856 752B                <1>       JNZ    short CD_ERR       ; BAD CARD
15094                              <1>       ;MOV   DX, HF_PORT+7
15095 00004858 668B15[E85C0000]    <1>       mov    dx, [HF_PORT]
15096 0000485F 80C207              <1>       add    dl, 7
15097 00004862 B090                <1>       MOV    AL,DIAG_CMD        ; START DIAGNOSE
15098 00004864 EE                  <1>       OUT    DX,AL
15099 00004865 E8A9010000          <1>       CALL   NOT_BUSY           ; WAIT FOR IT TO COMPLETE
15100 0000486A B480                <1>       MOV    AH,TIME_OUT
15101 0000486C 7517                <1>       JNZ    short CD_EXIT             ; TIME OUT ON DIAGNOSTIC
15102                              <1>       ;MOV   DX,HF_PORT+1       ; GET ERROR REGISTER
15103 0000486E 668B15[E85C0000]    <1>       mov    dx, [HF_PORT]
15104 00004875 FEC2                <1>       inc    dl
15105 00004877 EC                  <1>       IN     AL,DX
15106 00004878 A2[B2520100]        <1>       MOV    [HF_ERROR],AL      ; SAVE IT
15107 0000487D B400                <1>       MOV    AH,0
15108 0000487F 3C01                <1>       CMP    AL,1               ; CHECK FOR ALL OK
15109 00004881 7402                <1>       JE     SHORT CD_EXIT
15110 00004883 B420                <1> CD_ERR: MOV  AH,BAD_CNTLR
15111                              <1> CD_EXIT:
15112 00004885 8825[BB520100]      <1>       MOV    [DISK_STATUS1],AH
15113 0000488B C3                  <1>       RETn
15114                              <1>
15115                              <1> ;----------------------------------------
15116                              <1> ; COMMANDI                         :
15117                              <1> ;     REPEATEDLY INPUTS DATA TILL      :
15118                              <1> ;     NSECTOR RETURNS ZERO             :
15119                              <1> ;----------------------------------------
15120                              <1> COMMANDI:
15121 0000488C E862020000          <1>       CALL   CHECK_DMA          ; CHECK 64K BOUNDARY ERROR
15122 00004891 7253                <1>       JC     short CMD_ABORT
15123                              <1>       ;MOV   DI,BX
15124 00004893 89DF                <1>       mov    edi, ebx ; 21/02/2015
15125 00004895 E8C4000000          <1>       CALL   COMMAND            ; OUTPUT COMMAND
15126 0000489A 754A                <1>       JNZ    short CMD_ABORT
15127                              <1> CMD_I1:
15128 0000489C E836010000          <1>       CALL   _WAIT              ; WAIT FOR DATA REQUEST INTERRUPT
15129 000048A1 7543                <1>       JNZ    short TM_OUT       ; TIME OUT
15130                              <1> cmd_i1x: ; 18/02/2016
15131                              <1>       ;MOV   CX,256             ; SECTOR SIZE IN WORDS
15132 000048A3 B900010000          <1>       mov    ecx, 256 ; 21/02/2015
15133                              <1>       ;MOV   DX,HF_PORT
15134 000048A8 668B15[E85C0000]    <1>       mov    dx,[HF_PORT]
15135 000048AF FA                  <1>       CLI
15136 000048B0 FC                  <1>       CLD
15137 000048B1 F3666D              <1>       REP    INSW               ; GET THE SECTOR
15138 000048B4 FB                  <1>       STI
```

```
15139 000048B5 F645FE02            <1>       TEST   byte [CMD_BLOCK+6],ECC_MODE ; CHECK FOR NORMAL INPUT
15140 000048B9 7419                <1>       JZ     short CMD_I3
15141 000048BB E880010000          <1>       CALL   WAIT_DRQ            ; WAIT FOR DATA REQUEST
15142 000048C0 7224                <1>       JC     short TM_OUT
15143                              <1>       ;MOV   DX,HF_PORT
15144 000048C2 668B15[E85C0000]    <1>       mov    dx,[HF_PORT]
15145                              <1>       ;MOV   CX,4                ; GET ECC BYTES
15146 000048C9 B904000000          <1>       mov    ecx, 4 ; mov cx, 4
15147 000048CE EC                  <1> CMD_I2: IN   AL,DX
15148                              <1>       ;MOV   [ES:DI],AL          ; GO SLOW FOR BOARD
15149 000048CF 8807                <1>       mov    [edi], al ; 21/02/2015
15150 000048D1 47                  <1>       INC    eDI
15151 000048D2 E2FA                <1>       LOOP   CMD_I2
15152                              <1> CMD_I3:
15153                              <1>       ; wait for 400 ns
15154 000048D4 80C207              <1>       add    dl, 7
15155 000048D7 EC                  <1>       in     al, dx
15156 000048D8 EC                  <1>       in     al, dx
15157 000048D9 EC                  <1>       in     al, dx
15158                              <1>       ;
15159 000048DA E88C010000          <1>       CALL   CHECK_STATUS
15160 000048DF 7505                <1>       JNZ    short CMD_ABORT           ; ERROR RETURNED
15161 000048E1 FE4DF9              <1>       DEC    byte [CMD_BLOCK+1]  ; CHECK FOR MORE
15162                              <1>       ;JNZ   SHORT CMD_I1
15163 000048E4 75BD                <1>       jnz    short cmd_i1x ; 18/02/2016
15164                              <1> CMD_ABORT:
15165 000048E6 C3                  <1> TM_OUT: RETn
15166                              <1>
15167                              <1> ;---------------------------------------
15168                              <1> ; COMMANDO                         :
15169                              <1> ;     REPEATEDLY OUTPUTS DATA TILL     :
15170                              <1> ;     NSECTOR RETURNS ZERO             :
15171                              <1> ;---------------------------------------
15172                              <1> COMMANDO:
15173 000048E7 E807020000          <1>       CALL   CHECK_DMA           ; CHECK 64K BOUNDARY ERROR
15174 000048EC 72F8                <1>       JC     short CMD_ABORT
15175 000048EE 89DE                <1> CMD_OF: MOV  eSI,eBX ; 21/02/2015
15176 000048F0 E869000000          <1>       CALL   COMMAND             ; OUTPUT COMMAND
15177 000048F5 75EF                <1>       JNZ    short CMD_ABORT
15178 000048F7 E844010000          <1>       CALL   WAIT_DRQ            ; WAIT FOR DATA REQUEST
15179 000048FC 72E8                <1>       JC     short TM_OUT              ; TOO LONG
15180                              <1> CMD_O1: ;PUSH      DS
15181                              <1>       ;PUSH  ES                  ; MOVE ES TO DS
15182                              <1>       ;POP   DS
15183                              <1>       ;MOV   CX,256              ; PUT THE DATA OUT TO THE CARD
15184                              <1>       ;MOV   DX,HF_PORT
15185                              <1>       ; 01/02/2015
15186 000048FE 668B15[E85C0000]    <1>       mov    dx, [HF_PORT]
15187                              <1>       ;push  es
15188                              <1>       ;pop   ds
15189                              <1>       ;mov   cx, 256
15190 00004905 B900010000          <1>       mov    ecx, 256 ; 21/02/2015
15191 0000490A FA                  <1>       CLI
15192 0000490B FC                  <1>       CLD
15193 0000490C F3666F              <1>       REP    OUTSW
15194 0000490F FB                  <1>       STI
15195                              <1>       ;POP   DS                  ; RESTORE DS
15196 00004910 F645FE02            <1>       TEST   byte [CMD_BLOCK+6],ECC_MODE ; CHECK FOR NORMAL OUTPUT
15197 00004914 7419                <1>       JZ     short CMD_O3
15198 00004916 E825010000          <1>       CALL   WAIT_DRQ            ; WAIT FOR DATA REQUEST
15199 0000491B 72C9                <1>       JC     short TM_OUT
15200                              <1>       ;MOV   DX,HF_PORT
15201 0000491D 668B15[E85C0000]    <1>       mov    dx, [HF_PORT]
15202                              <1>       ;MOV   CX,4                ; OUTPUT THE ECC BYTES
15203 00004924 B904000000          <1>       mov    ecx, 4  ; mov cx, 4
15204                              <1> CMD_O2: ;MOV AL,[ES:SI]
15205 00004929 8A06                <1>       mov    al, [esi]
15206 0000492B EE                  <1>       OUT    DX,AL
15207 0000492C 46                  <1>       INC    eSI
15208 0000492D E2FA                <1>       LOOP   CMD_O2
15209                              <1> CMD_O3:
15210 0000492F E8A3000000          <1>       CALL   _WAIT               ; WAIT FOR SECTOR COMPLETE INTERRUPT
15211 00004934 75B0                <1>       JNZ    short TM_OUT        ; ERROR RETURNED
15212 00004936 E830010000          <1>       CALL   CHECK_STATUS
15213 0000493B 75A9                <1>       JNZ    short CMD_ABORT
15214 0000493D F605[B1520100]08    <1>       TEST   byte [HF_STATUS],ST_DRQ   ; CHECK FOR MORE
15215 00004944 75B8                <1>       JNZ    SHORT CMD_O1
15216                              <1>       ;MOV   DX,HF_PORT+2        ; CHECK RESIDUAL SECTOR COUNT
15217 00004946 668B15[E85C0000]    <1>       mov    dx, [HF_PORT]
15218                              <1>       ;add   dl, 2
15219 0000494D FEC2                <1>       inc    dl
15220 0000494F FEC2                <1>       inc    dl
15221 00004951 EC                  <1>       IN     AL,DX               ;
15222 00004952 A8FF                <1>       TEST   AL,0FFH             ;
15223 00004954 7407                <1>       JZ     short CMD_O4              ; COUNT = 0  OK
15224 00004956 C605[BB520100]BB    <1>       MOV    byte [DISK_STATUS1],UNDEF_ERR
15225                              <1>                                  ; OPERATION ABORTED - PARTIAL TRANSFER
15226                              <1> CMD_O4:
15227 0000495D C3                  <1>       RETn
15228                              <1>
15229                              <1> ;-----------------------------------------------------
15230                              <1> ; COMMAND                                       :
15231                              <1> ;     THIS ROUTINE OUTPUTS THE COMMAND BLOCK        :
15232                              <1> ; OUTPUT                                        :
15233                              <1> ;     BL = STATUS                               :
15234                              <1> ;     BH = ERROR REGISTER                       :
15235                              <1> ;-----------------------------------------------------
15236                              <1>
15237                              <1> COMMAND:
15238 0000495E 53                  <1>       PUSH   eBX                 ; WAIT FOR SEEK COMPLETE AND READY
15239                              <1>       ;;MOV  CX,DELAY_2          ; SET INITIAL DELAY BEFORE TEST
15240                              <1> COMMAND1:
15241                              <1>       ;;PUSH CX                  ; SAVE LOOP COUNT
```

```
15242 0000495F E879FEFFFF          <1>       CALL   TST_RDY            ; CHECK DRIVE READY
15243                              <1>       ;;POP  CX
15244 00004964 7419                <1>       JZ     short COMMAND2            ; DRIVE IS READY
15245 00004966 803D[BB520100]80    <1>         CMP     byte [DISK_STATUS1],TIME_OUT ; TST_RDY TIMED OUT--GIVE UP
15246                              <1>       ;JZ    short CMD_TIMEOUT
15247                              <1>       ;;LOOP COMMAND1           ; KEEP TRYING FOR A WHILE
15248                              <1>       ;JMP   SHORT COMMAND4            ; ITS NOT GOING TO GET READY
15249 0000496D 7507                <1>       jne    short COMMAND4
15250                              <1> CMD_TIMEOUT:
15251 0000496F C605[BB520100]20    <1>       MOV    byte [DISK_STATUS1],BAD_CNTLR
15252                              <1> COMMAND4:
15253 00004976 5B                  <1>       POP    eBX
15254 00004977 803D[BB520100]00    <1>         CMP     byte [DISK_STATUS1],0   ; SET CONDITION CODE FOR CALLER
15255 0000497E C3                  <1>       RETn
15256                              <1> COMMAND2:
15257 0000497F 5B                  <1>       POP    eBX
15258 00004980 57                  <1>       PUSH   eDI
15259 00004981 C605[B3520100]00    <1>       MOV    byte [HF_INT_FLAG],0      ; RESET INTERRUPT FLAG
15260 00004988 FA                  <1>       CLI                       ; INHIBIT INTERRUPTS WHILE CHANGING MASK
15261 00004989 E4A1                <1>       IN     AL,INTB01          ; TURN ON SECOND INTERRUPT CHIP
15262                              <1>       ;AND   AL,0BFH
15263 0000498B 243F                <1>       and    al, 3Fh                  ; Enable IRQ 14 & 15
15264                              <1>       ;JMP   $+2
15265                              <1>       IODELAY
15266 0000498D EB00                <2>  jmp short $+2
15267 0000498F EB00                <2>  jmp short $+2
15268 00004991 E6A1                <1>       OUT    INTB01,AL
15269 00004993 E421                <1>       IN     AL,INTA01          ; LET INTERRUPTS PASS THRU TO
15270 00004995 24FB                <1>       AND    AL,0FBH            ;  SECOND CHIP
15271                              <1>       ;JMP   $+2
15272                              <1>       IODELAY
15273 00004997 EB00                <2>  jmp short $+2
15274 00004999 EB00                <2>  jmp short $+2
15275 0000499B E621                <1>       OUT    INTA01,AL
15276 0000499D FB                  <1>       STI
15277 0000499E 31FF                <1>       XOR    eDI,eDI                  ; INDEX THE COMMAND TABLE
15278                              <1>       ;MOV   DX,HF_PORT+1       ; DISK ADDRESS
15279 000049A0 668B15[E85C0000]    <1>       mov    dx, [HF_PORT]
15280 000049A7 FEC2                <1>       inc    dl
15281 000049A9 F605[BD520100]C0    <1>       TEST   byte [CONTROL_BYTE],0C0H ; CHECK FOR RETRY SUPPRESSION
15282 000049B0 7411                <1>       JZ     short COMMAND3
15283 000049B2 8A45FE              <1>       MOV    AL, [CMD_BLOCK+6]   ; YES-GET OPERATION CODE
15284 000049B5 24F0                <1>       AND    AL,0F0H             ; GET RID OF MODIFIERS
15285 000049B7 3C20                <1>       CMP    AL,20H              ; 20H-40H IS READ, WRITE, VERIFY
15286 000049B9 7208                <1>       JB     short COMMAND3
15287 000049BB 3C40                <1>       CMP    AL,40H
15288 000049BD 7704                <1>       JA     short COMMAND3
15289 000049BF 804DFE01            <1>       OR     byte [CMD_BLOCK+6],NO_RETRIES
15290                              <1>                                  ; VALID OPERATION FOR RETRY SUPPRESS
15291                              <1> COMMAND3:
15292 000049C3 8A443DF8            <1>       MOV    AL,[CMD_BLOCK+eDI]  ; GET THE COMMAND STRING BYTE
15293 000049C7 EE                  <1>       OUT    DX,AL               ; GIVE IT TO CONTROLLER
15294                              <1>       IODELAY
15295 000049C8 EB00                <2>  jmp short $+2
15296 000049CA EB00                <2>  jmp short $+2
15297 000049CC 47                  <1>       INC    eDI                 ; NEXT BYTE IN COMMAND BLOCK
15298 000049CD 6642                <1>       INC    DX                  ; NEXT DISK ADAPTER REGISTER
15299 000049CF 6683FF07            <1>       cmp    di, 7  ; 1/1/2015   ; ALL DONE?
15300 000049D3 75EE                <1>       JNZ    short COMMAND3            ; NO--GO DO NEXT ONE
15301 000049D5 5F                  <1>       POP    eDI
15302 000049D6 C3                  <1>       RETn                       ; ZERO FLAG IS SET
15303                              <1>
15304                              <1> ;CMD_TIMEOUT:
15305                              <1> ;     MOV    byte [DISK_STATUS1],BAD_CNTLR
15306                              <1> ;COMMAND4:
15307                              <1> ;     POP    BX
15308                              <1> ;     CMP    [DISK_STATUS1],0    ; SET CONDITION CODE FOR CALLER
15309                              <1> ;     RETn
15310                              <1>
15311                              <1> ;---------------------------------------
15312                              <1> ;     WAIT FOR INTERRUPT        :
15313                              <1> ;---------------------------------------
15314                              <1> ;WAIT:
15315                              <1> _WAIT:
15316 000049D7 FB                  <1>       STI                        ; MAKE SURE INTERRUPTS ARE ON
15317                              <1>       ;SUB   CX,CX               ; SET INITIAL DELAY BEFORE TEST
15318                              <1>       ;CLC
15319                              <1>       ;MOV   AX,9000H            ; DEVICE WAIT INTERRUPT
15320                              <1>       ;INT   15H
15321                              <1>       ;JC    WT2                 ; DEVICE TIMED OUT
15322                              <1>       ;MOV   BL,DELAY_1          ; SET DELAY COUNT
15323                              <1>
15324                              <1>       ;mov   bl, WAIT_HDU_INT_HI
15325                              <1>       ;; 21/02/2015
15326                              <1>       ;;mov  bl, WAIT_HDU_INT_HI + 1
15327                              <1>       ;;mov  cx, WAIT_HDU_INT_LO
15328 000049D8 B915160500          <1>       mov    ecx, WAIT_HDU_INT_LH
15329                              <1>                                  ; (AWARD BIOS -> WAIT_FOR_MEM)
15330                              <1> ;-----      WAIT LOOP
15331                              <1>
15332                              <1> WT1:
15333                              <1>       ;TEST  byte [HF_INT_FLAG],80H    ; TEST FOR INTERRUPT
15334 000049DD F605[B3520100]C0    <1>       test   byte [HF_INT_FLAG],0C0h
15335                              <1>       ;LOOPZ WT1
15336 000049E4 7517                <1>       JNZ    short WT3           ; INTERRUPT--LETS GO
15337                              <1>       ;DEC   BL
15338                              <1>       ;JNZ   short WT1           ; KEEP TRYING FOR A WHILE
15339                              <1>
15340                              <1> WT1_hi:
15341 000049E6 E461                <1>       in     al, SYS1 ; 61h (PORT_B)   ; wait for lo to hi
15342 000049E8 A810                <1>       test   al, 10h                   ; transition on memory
15343 000049EA 75FA                <1>       jnz    short WT1_hi        ; refresh.
15344                              <1> WT1_lo:
```

```
15345 000049EC E461               <1>       in     al, SYS1           ; 061h (PORT_B)
15346 000049EE A810               <1>       test   al, 10h
15347 000049F0 74FA               <1>       jz     short WT1_lo
15348 000049F2 E2E9               <1>       loop   WT1
15349                             <1>       ;;or   bl, bl
15350                             <1>       ;;jz   short WT2
15351                             <1>       ;;dec  bl
15352                             <1>       ;;jmp  short WT1
15353                             <1>       ;dec   bl
15354                             <1>       ;jnz   short WT1
15355                             <1>
15356 000049F4 C605[BB520100]80   <1> WT2:  MOV    byte [DISK_STATUS1],TIME_OUT ; REPORT TIME OUT ERROR
15357 000049FB EB0E               <1>       JMP    SHORT WT4
15358 000049FD C605[BB520100]00   <1> WT3:  MOV    byte [DISK_STATUS1],0
15359 00004A04 C605[B3520100]00   <1>       MOV    byte [HF_INT_FLAG],0
15360 00004A0B 803D[BB520100]00   <1> WT4:  CMP    byte [DISK_STATUS1],0     ; SET CONDITION CODE FOR CALLER
15361 00004A12 C3                 <1>       RETn
15362                             <1>
15363                             <1> ;---------------------------------------
15364                             <1> ;     WAIT FOR CONTROLLER NOT BUSY     :
15365                             <1> ;---------------------------------------
15366                             <1> NOT_BUSY:
15367 00004A13 FB                 <1>       STI                        ; MAKE SURE INTERRUPTS ARE ON
15368                             <1>       ;PUSH  eBX
15369                             <1>       ;SUB   CX,CX               ; SET INITIAL DELAY BEFORE TEST
15370 00004A14 668B15[E85C0000]   <1>       mov    DX, [HF_PORT]
15371 00004A1B 80C207             <1>       add    dl, 7               ; Status port (HF_PORT+7)
15372                             <1>       ;MOV   BL,DELAY_1
15373                             <1>                                  ; wait for 10 seconds
15374                             <1>       ;mov   cx, WAIT_HDU_INT_LO ; 1615h
15375                             <1>       ;;mov  bl, WAIT_HDU_INT_HI ;   05h
15376                             <1>       ;mov   bl, WAIT_HDU_INT_HI + 1
15377 00004A1E B915160500         <1>       mov    ecx, WAIT_HDU_INT_LH  ; 21/02/2015
15378                             <1>       ;
15379                             <1> ;;      mov     byte [wait_count], 0    ; Reset wait counter
15380                             <1> NB1:
15381 00004A23 EC                 <1>       IN     AL,DX               ; CHECK STATUS
15382                             <1>       ;TEST  AL,ST_BUSY
15383 00004A24 2480               <1>       and    al, ST_BUSY
15384                             <1>       ;LOOPNZ      NB1
15385 00004A26 7410               <1>       JZ     short NB2           ; NOT BUSY--LETS GO
15386                             <1>       ;DEC   BL
15387                             <1>       ;JNZ   short NB1           ; KEEP TRYING FOR A WHILE
15388                             <1>
15389 00004A28 E461               <1> NB1_hi: IN   AL,SYS1                   ; wait for hi to lo
15390 00004A2A A810               <1>       TEST   AL,010H                   ; transition on memory
15391 00004A2C 75FA               <1>       JNZ    SHORT NB1_hi        ; refresh.
15392 00004A2E E461               <1> NB1_lo: IN   AL,SYS1
15393 00004A30 A810               <1>       TEST   AL,010H
15394 00004A32 74FA               <1>       JZ     short NB1_lo
15395 00004A34 E2ED               <1>       LOOP   NB1
15396                             <1>       ;dec   bl
15397                             <1>       ;jnz   short NB1
15398                             <1>       ;
15399                             <1> ;;      cmp     byte [wait_count], 182  ; 10 seconds (182 timer ticks)
15400                             <1> ;;    jb     short NB1
15401                             <1>       ;
15402                             <1>       ;MOV   [DISK_STATUS1],TIME_OUT   ; REPORT TIME OUT ERROR
15403                             <1>       ;JMP   SHORT NB3
15404 00004A36 B080               <1>       mov    al, TIME_OUT
15405                             <1> NB2:
15406                             <1>       ;MOV   byte [DISK_STATUS1],0
15407                             <1> ;NB3:
15408                             <1>       ;POP   eBX
15409 00004A38 A2[BB520100]       <1>       mov    [DISK_STATUS1], al  ;;; will be set after return
15410                             <1>       ;CMP   byte [DISK_STATUS1],0     ; SET CONDITION CODE FOR CALLER
15411 00004A3D 08C0               <1>       or     al, al              ; (zf = 0 --> timeout)
15412 00004A3F C3                 <1>       RETn
15413                             <1>
15414                             <1> ;---------------------------------------
15415                             <1> ;     WAIT FOR DATA REQUEST           :
15416                             <1> ;---------------------------------------
15417                             <1> WAIT_DRQ:
15418                             <1>       ;MOV   CX,DELAY_3
15419                             <1>       ;MOV   DX,HF_PORT+7
15420 00004A40 668B15[E85C0000]   <1>       mov    dx, [HF_PORT]
15421 00004A47 80C207             <1>       add    dl, 7
15422                             <1>       ;;MOV  bl, WAIT_HDU_DRQ_HI ; 0
15423                             <1>       ;MOV   cx, WAIT_HDU_DRQ_LO ; 1000 (30 milli seconds)
15424                             <1>                                  ; (but it is written as 2000
15425                             <1>                                  ; micro seconds in ATORGS.ASM file
15426                             <1>                                  ; of Award Bios - 1999, D1A0622)
15427 00004A4A B9E8030000         <1>       mov    ecx, WAIT_HDU_DRQ_LH ; 21/02/2015
15428 00004A4F EC                 <1> WQ_1: IN     AL,DX               ; GET STATUS
15429 00004A50 A808               <1>       TEST   AL,ST_DRQ           ; WAIT FOR DRQ
15430 00004A52 7516               <1>       JNZ    short WQ_OK
15431                             <1>       ;LOOP  WQ_1                ; KEEP TRYING FOR A SHORT WHILE
15432                             <1> WQ_hi:
15433 00004A54 E461               <1>       IN     AL,SYS1                   ; wait for hi to lo
15434 00004A56 A810               <1>       TEST   AL,010H                   ; transition on memory
15435 00004A58 75FA               <1>       JNZ    SHORT WQ_hi         ; refresh.
15436 00004A5A E461               <1> WQ_lo:  IN      AL,SYS1
15437 00004A5C A810               <1>       TEST   AL,010H
15438 00004A5E 74FA               <1>       JZ     SHORT WQ_lo
15439 00004A60 E2ED               <1>       LOOP   WQ_1
15440                             <1>
15441 00004A62 C605[BB520100]80   <1>         MOV     byte [DISK_STATUS1],TIME_OUT ; ERROR
15442 00004A69 F9                 <1>       STC
15443                             <1> WQ_OK:
15444 00004A6A C3                 <1>       RETn
15445                             <1> ;WQ_OK:     ;CLC
15446                             <1> ;     RETn
15447                             <1>
```

```
15448                                  <1> ;----------------------------------------
15449                                  <1> ;     CHECK FIXED DISK STATUS   :
15450                                  <1> ;----------------------------------------
15451                                  <1> CHECK_STATUS:
15452 00004A6B E813000000               <1>       CALL   CHECK_ST           ; CHECK THE STATUS BYTE
15453 00004A70 7509                     <1>       JNZ    short CHECK_S1           ; AN ERROR WAS FOUND
15454 00004A72 A801                     <1>       TEST   AL,ST_ERROR        ; WERE THERE ANY OTHER ERRORS
15455 00004A74 7405                     <1>       JZ     short CHECK_S1           ; NO ERROR REPORTED
15456 00004A76 E849000000               <1>       CALL   CHECK_ER           ; ERROR REPORTED
15457                                  <1> CHECK_S1:
15458 00004A7B 803D[BB520100]00         <1>       CMP    byte [DISK_STATUS1],0     ; SET STATUS FOR CALLER
15459 00004A82 C3                       <1>       RETn
15460                                  <1>
15461                                  <1> ;----------------------------------------
15462                                  <1> ;     CHECK FIXED DISK STATUS BYTE     :
15463                                  <1> ;----------------------------------------
15464                                  <1> CHECK_ST:
15465                                  <1>       ;MOV   DX,HF_PORT+7       ; GET THE STATUS
15466 00004A83 668B15[E85C0000]         <1>       mov    dx, [HF_PORT]
15467 00004A8A 80C207                   <1>       add    dl, 7
15468                                  <1>
15469                                  <1>       ; 17/02/2016
15470                                  <1>       ;(http://wiki.osdev.org/ATA_PIO_Mode)
15471                                  <1>       ;"delay 400ns to allow drive to set new values of BSY and DRQ"
15472 00004A8D EC                       <1>       IN     AL,DX
15473                                  <1>       ;in    al, dx ; 100ns
15474                                  <1>       ;in    al, dx ; 100ns
15475                                  <1>       ;in    al, dx ; 100ns
15476                                  <1>       NEWIODELAY ; 18/02/2016 (AWARD BIOS - 1999, 'CKST' in AHSDK.ASM)
15477 00004A8E E6EB                     <2>  out 0ebh,al
15478                                  <1>       ;
15479 00004A90 A2[B1520100]             <1>       MOV    [HF_STATUS],AL
15480 00004A95 B400                     <1>       MOV    AH,0
15481 00004A97 A880                     <1>       TEST   AL,ST_BUSY         ; IF STILL BUSY
15482 00004A99 751A                     <1>       JNZ    short CKST_EXIT          ;  REPORT OK
15483 00004A9B B4CC                     <1>       MOV    AH,WRITE_FAULT
15484 00004A9D A820                     <1>       TEST   AL,ST_WRT_FLT      ; CHECK FOR WRITE FAULT
15485 00004A9F 7514                     <1>       JNZ    short CKST_EXIT
15486 00004AA1 B4AA                     <1>       MOV    AH,NOT_RDY
15487 00004AA3 A840                     <1>       TEST   AL,ST_READY        ; CHECK FOR NOT READY
15488 00004AA5 740E                     <1>       JZ     short CKST_EXIT
15489 00004AA7 B440                     <1>       MOV    AH,BAD_SEEK
15490 00004AA9 A810                     <1>       TEST   AL,ST_SEEK_COMPL   ; CHECK FOR SEEK NOT COMPLETE
15491 00004AAB 7408                     <1>       JZ     short CKST_EXIT
15492 00004AAD B411                     <1>       MOV    AH,DATA_CORRECTED
15493 00004AAF A804                     <1>       TEST   AL,ST_CORRCTD      ; CHECK FOR CORRECTED ECC
15494 00004AB1 7502                     <1>       JNZ    short CKST_EXIT
15495 00004AB3 B400                     <1>       MOV    AH,0
15496                                  <1> CKST_EXIT:
15497 00004AB5 8825[BB520100]           <1>       MOV    [DISK_STATUS1],AH  ; SET ERROR FLAG
15498 00004ABB 80FC11                   <1>       CMP    AH,DATA_CORRECTED  ; KEEP GOING WITH DATA CORRECTED
15499 00004ABE 7403                     <1>       JZ     short CKST_EX1
15500 00004AC0 80FC00                   <1>       CMP    AH,0
15501                                  <1> CKST_EX1:
15502 00004AC3 C3                       <1>       RETn
15503                                  <1>
15504                                  <1> ;----------------------------------------
15505                                  <1> ;     CHECK FIXED DISK ERROR REGISTER :
15506                                  <1> ;----------------------------------------
15507                                  <1> CHECK_ER:
15508                                  <1>       ;MOV   DX, HF_PORT+1      ; GET THE ERROR REGISTER
15509 00004AC4 668B15[E85C0000]         <1>       mov    dx, [HF_PORT]      ;
15510 00004ACB FEC2                     <1>       inc    dl
15511 00004ACD EC                       <1>       IN     AL,DX
15512 00004ACE A2[B2520100]             <1>       MOV    [HF_ERROR],AL
15513 00004AD3 53                       <1>       PUSH   eBX ; 21/02/2015
15514 00004AD4 B908000000               <1>       MOV    eCX,8              ; TEST ALL 8 BITS
15515 00004AD9 D0E0                     <1> CK1:  SHL    AL,1               ; MOVE NEXT ERROR BIT TO CARRY
15516 00004ADB 7202                     <1>       JC     short CK2          ; FOUND THE ERROR
15517 00004ADD E2FA                     <1>       LOOP   CK1                ; KEEP TRYING
15518 00004ADF BB[DC5C0000]             <1> CK2:  MOV    eBX, ERR_TBL       ; COMPUTE ADDRESS OF
15519 00004AE4 01CB                     <1>       ADD    eBX,eCX                  ; ERROR CODE
15520                                  <1>       ;;MOV  AH,BYTE [CS:BX]          ; GET ERROR CODE
15521                                  <1>       ;mov   ah, [bx]
15522 00004AE6 8A23                     <1>       mov    ah, [ebx] ; 21/02/2015
15523 00004AE8 8825[BB520100]           <1> CKEX: MOV    [DISK_STATUS1],AH  ; SAVE ERROR CODE
15524 00004AEE 5B                       <1>       POP    eBX
15525 00004AEF 80FC00                   <1>       CMP    AH,0
15526 00004AF2 C3                       <1>       RETn
15527                                  <1>
15528                                  <1> ;--------------------------------------------------------
15529                                  <1> ; CHECK_DMA                                :
15530                                  <1> ;  -CHECK ES:BX AND # SECTORS TO MAKE SURE THAT IT WILL :
15531                                  <1> ;   FIT WITHOUT SEGMENT OVERFLOW.                      :
15532                                  <1> ;  -ES:BX HAS BEEN REVISED TO THE FORMAT SSSS:000X     :
15533                                  <1> ;  -OK IF # SECTORS < 80H (7FH IF LONG READ OR WRITE)     :
15534                                  <1> ;  -OK IF # SECTORS = 80H (7FH) AND BX <= 00H (04H) :
15535                                  <1> ;  -ERROR OTHERWISE                                    :
15536                                  <1> ;--------------------------------------------------------
15537                                  <1> CHECK_DMA:
15538 00004AF3 6650                     <1>       PUSH   AX                 ; SAVE REGISTERS
15539 00004AF5 66B80080                 <1>       MOV    AX,8000H           ; AH = MAX # SECTORS AL = MAX OFFSET
15540 00004AF9 F645FE02                 <1>         TEST    byte [CMD_BLOCK+6],ECC_MODE
15541 00004AFD 7404                     <1>       JZ     short CKD1
15542 00004AFF 66B8047F                 <1>       MOV    AX,7F04H           ; ECC IS 4 MORE BYTES
15543 00004B03 3A65F9                   <1> CKD1: CMP    AH, [CMD_BLOCK+1]  ; NUMBER OF SECTORS
15544 00004B06 7706                     <1>       JA     short CKDOK        ; IT WILL FIT
15545 00004B08 7208                     <1>       JB     short CKDERR       ; TOO MANY
15546 00004B0A 38D8                     <1>       CMP    AL,BL              ; CHECK OFFSET ON MAX SECTORS
15547 00004B0C 7204                     <1>       JB     short CKDERR       ; ERROR
15548 00004B0E F8                       <1> CKDOK:       CLC                      ; CLEAR CARRY
15549 00004B0F 6658                     <1>       POP    AX
15550 00004B11 C3                       <1>       RETn                      ; NORMAL RETURN
```

```
15551 00004B12 F9                  <1> CKDERR: STC                     ; INDICATE ERROR
15552 00004B13 C605[BB520100]09    <1>         MOV     byte [DISK_STATUS1],DMA_BOUNDARY
15553 00004B1A 6658                <1>       POP    AX
15554 00004B1C C3                  <1>       RETn
15555                              <1>
15556                              <1> ;---------------------------------------
15557                              <1> ;     SET UP ES:BX-> DISK PARMS :
15558                              <1> ;---------------------------------------
15559                              <1>
15560                              <1> ; INPUT -> DL = 0 based drive number
15561                              <1> ; OUTPUT -> ES:BX = disk parameter table address
15562                              <1>
15563                              <1> GET_VEC:
15564                              <1>       ;SUB   AX,AX              ; GET DISK PARAMETER ADDRESS
15565                              <1>       ;MOV   ES,AX
15566                              <1>       ;TEST  DL,1
15567                              <1>       ;JZ    short GV_0
15568                              <1> ;     LES    BX,[HF1_TBL_VEC]    ; ES:BX -> DRIVE PARAMETERS
15569                              <1> ;     JMP    SHORT GV_EXIT
15570                              <1> ;GV_0:
15571                              <1> ;     LES    BX,[HF_TBL_VEC]           ; ES:BX -> DRIVE PARAMETERS
15572                              <1> ;
15573                              <1>       ;xor   bh, bh
15574 00004B1D 31DB                <1>       xor    ebx, ebx
15575 00004B1F 88D3                <1>       mov    bl, dl
15576                              <1>       ;;02/01/2015
15577                              <1>       ;;shl  bl, 1               ; port address offset
15578                              <1>       ;;mov  ax, [bx+hd_ports]   ; Base port address (1F0h, 170h)
15579                              <1>       ;;shl  bl, 1               ; dpt pointer offset
15580 00004B21 C0E302              <1>       shl    bl, 2  ;;
15581                              <1>       ;add   bx, HF_TBL_VEC            ; Disk parameter table pointer
15582 00004B24 81C3[C0520100]      <1>       add    ebx, HF_TBL_VEC ; 21/02/2015
15583                              <1>       ;push  word [bx+2]         ; dpt segment
15584                              <1>       ;pop   es
15585                              <1>       ;mov   bx, [bx]            ; dpt offset
15586 00004B2A 8B1B                <1>       mov    ebx, [ebx]
15587                              <1> ;GV_EXIT:
15588 00004B2C C3                  <1>       RETn
15589                              <1>
15590                              <1> hdc1_int: ; 21/02/2015
15591                              <1> ;--- HARDWARE INT 76H -- ( IRQ LEVEL  14 ) ----------------------
15592                              <1> ;                                                 :
15593                              <1> ;     FIXED DISK INTERRUPT ROUTINE                      :
15594                              <1> ;                                                 :
15595                              <1> ;-------------------------------------------------------------
15596                              <1>
15597                              <1> ; 22/12/2014
15598                              <1> ; IBM PC-XT Model 286 System BIOS Source Code - DISK.ASM (HD_INT)
15599                              <1> ;      '11/15/85'
15600                              <1> ; AWARD BIOS 1999 (D1A0622)
15601                              <1> ;     Source Code - ATORGS.ASM (INT_HDISK, INT_HDISK1)
15602                              <1>
15603                              <1> ;int_76h:
15604                              <1> HD_INT:
15605 00004B2D 6650                <1>       PUSH   AX
15606 00004B2F 1E                  <1>       PUSH   DS
15607                              <1>       ;CALL  DDS
15608                              <1>       ; 21/02/2015 (32 bit, 386 pm modification)
15609 00004B30 66B81000            <1>       mov    ax, KDATA
15610 00004B34 8ED8                <1>       mov    ds, ax
15611                              <1>       ;
15612                              <1>       ;;MOV  @HF_INT_FLAG,0FFH   ; ALL DONE
15613                              <1>         ;mov     byte [CS:HF_INT_FLAG], 0FFh
15614 00004B36 C605[B3520100]FF    <1>       mov    byte [HF_INT_FLAG], 0FFh
15615                              <1>       ;
15616 00004B3D 6652                <1>       push   dx
15617 00004B3F 66BAF701            <1>       mov    dx, HDC1_BASEPORT+7 ; Status Register (1F7h)
15618                              <1>                                  ; Clear Controller
15619                              <1> Clear_IRQ1415:                         ; (Award BIOS - 1999)
15620 00004B43 EC                  <1>       in     al, dx              ;
15621 00004B44 665A                <1>       pop    dx
15622                              <1>       NEWIODELAY
15623 00004B46 E6EB                <2>  out 0ebh,al
15624                              <1>       ;
15625 00004B48 B020                <1>       MOV    AL,EOI              ; NON-SPECIFIC END OF INTERRUPT
15626 00004B4A E6A0                <1>       OUT    INTB00,AL           ; FOR CONTROLLER #2
15627                              <1>       ;JMP   $+2                 ; WAIT
15628                              <1>       NEWIODELAY
15629 00004B4C E6EB                <2>  out 0ebh,al
15630 00004B4E E620                <1>       OUT    INTA00,AL           ; FOR CONTROLLER #1
15631 00004B50 1F                  <1>       POP    DS
15632                              <1>       ;STI                       ; RE-ENABLE INTERRUPTS
15633                              <1>       ;MOV   AX,9100H            ; DEVICE POST
15634                              <1>       ;INT   15H                 ;  INTERRUPT
15635                              <1> irq15_iret: ; 25/02/2015
15636 00004B51 6658                <1>       POP    AX
15637 00004B53 CF                  <1>       IRETd                      ; RETURN FROM INTERRUPT
15638                              <1>
15639                              <1> hdc2_int: ; 21/02/2015
15640                              <1> ;++++ HARDWARE INT 77H ++ ( IRQ LEVEL  15 ) ++++++++++++++++++++
15641                              <1> ;                                                 :
15642                              <1> ;     FIXED DISK INTERRUPT ROUTINE                      :
15643                              <1> ;                                                 :
15644                              <1> ;+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
15645                              <1>
15646                              <1> ;int_77h:
15647                              <1> HD1_INT:
15648 00004B54 6650                <1>       PUSH   AX
15649                              <1>       ; Check if that is a spurious IRQ (from slave PIC)
15650                              <1>       ; 25/02/2015 (source: http://wiki.osdev.org/8259_PIC)
15651 00004B56 B00B                <1>       mov    al, 0Bh  ; In-Service Register
15652 00004B58 E6A0                <1>       out    0A0h, al
15653 00004B5A EB00                <1>         jmp short $+2
```

```
15654 00004B5C EB00                <1>       jmp short $+2
15655 00004B5E E4A0                <1>       in     al, 0A0h
15656 00004B60 2480                <1>       and    al, 80h ; bit 7 (is it real IRQ 15 or fake?)
15657 00004B62 74ED                <1>       jz     short irq15_iret ; Fake (spurious)IRQ, do not send EOI)
15658                              <1>       ;
15659 00004B64 1E                  <1>       PUSH   DS
15660                              <1>       ;CALL  DDS
15661                              <1>       ; 21/02/2015 (32 bit, 386 pm modification)
15662 00004B65 66B81000            <1>       mov    ax, KDATA
15663 00004B69 8ED8                <1>       mov    ds, ax
15664                              <1>       ;
15665                              <1>       ;;MOV  @HF_INT_FLAG,0FFH   ; ALL DONE
15666                              <1>         ;or      byte [CS:HF_INT_FLAG],0C0h
15667 00004B6B 800D[B3520100]C0    <1>       or     byte [HF_INT_FLAG], 0C0h
15668                              <1>       ;
15669 00004B72 6652                <1>       push   dx
15670 00004B74 66BA7701            <1>       mov    dx, HDC2_BASEPORT+7 ; Status Register (177h)
15671                              <1>                                  ; Clear Controller (Award BIOS 1999)
15672 00004B78 EBC9                <1>       jmp    short Clear_IRQ1415
15673                              <1>
15674                              <1>
15675                              <1> ;%include 'diskdata.inc' ; 11/03/2015
15676                              <1> ;%include 'diskbss.inc' ; 11/03/2015
15677                              <1>
15678                              <1>
15679                              <1> ;////////////////////////////////////////////////////////////////
15680                              <1> ;; END OF DISK I/O SYTEM ///
15681                                  %include 'memory.s'  ; 09/03/2015
15682                              <1> ; ****************************************************************************
15683                              <1> ; TRDOS386.ASM (TRDOS 386 Kernel) - v2.0.0 - memory.s
15684                              <1> ; ----------------------------------------------------------------------------
15685                              <1> ; Last Update: 22/07/2017
15686                              <1> ; ----------------------------------------------------------------------------
15687                              <1> ; Beginning: 24/01/2016
15688                              <1> ; ----------------------------------------------------------------------------
15689                              <1> ; Assembler: NASM version 2.11 (trdos386.s)
15690                              <1> ; ----------------------------------------------------------------------------
15691                              <1> ; Turkish Rational DOS
15692                              <1> ; Operating System Project v2.0 by ERDOGAN TAN (Beginning: 04/01/2016)
15693                              <1> ;
15694                              <1> ; Derived from 'Retro UNIX 386 Kernel - v0.2.1.0' source code by Erdogan Tan
15695                              <1> ; memory.inc (18/10/2015)
15696                              <1> ; ****************************************************************************
15697                              <1>
15698                              <1> ; MEMORY.ASM - Retro UNIX 386 v1 MEMORY MANAGEMENT FUNCTIONS (PROCEDURES)
15699                              <1> ; Retro UNIX 386 v1 Kernel (unix386.s, v0.2.0.14) - MEMORY.INC
15700                              <1> ; Last Modification: 18/10/2015
15701                              <1>
15702                              <1> ; ///////// MEMORY MANAGEMENT FUNCTIONS (PROCEDURES) ///////////////
15703                              <1>
15704                              <1> ;;04/11/2014 (unix386.s)
15705                              <1> ;PDE_A_PRESENT     equ   1            ; Present flag for PDE
15706                              <1> ;PDE_A_WRITE equ   2            ; Writable (write permission) flag
15707                              <1> ;PDE_A_USER  equ   4            ; User (non-system/kernel) page flag
15708                              <1> ;;
15709                              <1> ;PTE_A_PRESENT     equ   1            ; Present flag for PTE (bit 0)
15710                              <1> ;PTE_A_WRITE equ   2            ; Writable (write permission) flag (bit 1)
15711                              <1> ;PTE_A_USER  equ   4            ; User (non-system/kernel) page flag (bit 2)
15712                              <1> ;PTE_A_ACCESS   equ      32           ; Accessed flag (bit 5) ; 09/03/2015
15713                              <1>
15714                              <1> ; 27/04/2015
15715                              <1> ; 09/03/2015
15716                              <1> PAGE_SIZE    equ 4096           ; page size in bytes
15717                              <1> PAGE_SHIFT   equ 12             ; page table shift count
15718                              <1> PAGE_D_SHIFT       equ 22 ; 12 + 10    ; page directory shift count
15719                              <1> PAGE_OFF     equ 0FFFh          ; 12 bit byte offset in page frame
15720                              <1> PTE_MASK     equ 03FFh          ; page table entry mask
15721                              <1> PTE_DUPLICATED  equ 200h        ; duplicated page sign (AVL bit 0)
15722                              <1> PDE_A_CLEAR  equ 0F000h         ; to clear PDE attribute bits
15723                              <1> PTE_A_CLEAR  equ 0F000h         ; to clear PTE attribute bits
15724                              <1> LOGIC_SECT_SIZE equ 512               ; logical sector size
15725                              <1> ERR_MAJOR_PF equ 0E0h           ; major error: page fault
15726                              <1> ERR_MINOR_IM equ 4 ;15/10/2016 (1->4); insufficient (out of) memory
15727                              <1> ERR_MINOR_PV equ 6 ;15/10/2016 (1->4); protection violation
15728                              <1> SWP_DISK_READ_ERR     equ 40
15729                              <1> SWP_DISK_NOT_PRESENT_ERR   equ 41
15730                              <1> SWP_SECTOR_NOT_PRESENT_ERR equ 42
15731                              <1> SWP_NO_FREE_SPACE_ERR      equ 43
15732                              <1> SWP_DISK_WRITE_ERR         equ 44
15733                              <1> SWP_NO_PAGE_TO_SWAP_ERR    equ 45
15734                              <1> PTE_A_ACCESS_BIT equ 5  ; Bit 5 (accessed flag)
15735                              <1> SECTOR_SHIFT     equ 3  ; sector shift (to convert page block number)
15736                              <1> ; 12/07/2016
15737                              <1> PTE_SHARED   equ 400h           ; AVL bit 1, direct memory access bit
15738                              <1>                                 ; (Indicates that the page is not allocated
15739                              <1>                                 ; for the process, it is a shared or system
15740                              <1>                                        ; page, it must not be deallocated!)
15741                              <1> ;
15742                              <1> ;; Retro Unix 386 v1 - paging method/principles
15743                              <1> ;;
15744                              <1> ;; 10/10/2014
15745                              <1> ;; RETRO UNIX 386 v1 - PAGING METHOD/PRINCIPLES
15746                              <1> ;;
15747                              <1> ;; KERNEL PAGE MAP: 1 to 1 physical memory page map
15748                              <1> ;;    (virtual address = physical address)
15749                              <1> ;; KERNEL PAGE TABLES:
15750                              <1> ;;    Kernel page directory and all page tables are
15751                              <1> ;;    on memory as initialized, as equal to physical memory
15752                              <1> ;;    layout. Kernel pages can/must not be swapped out/in.
15753                              <1> ;;
15754                              <1> ;;    what for: User pages may be swapped out, when accessing
15755                              <1> ;;    a page in kernel/system mode, if it would be swapped out,
15756                              <1> ;;    kernel would have to swap it in! But it is also may be
```

```
15757                           <1> ;;    in use by a user process. (In system/kernel mode
15758                           <1> ;;    kernel can access all memory pages even if they are
15759                           <1> ;;    reserved/allocated for user processes. Swap out/in would
15760                           <1> ;;    cause conflicts.)
15761                           <1> ;;
15762                           <1> ;;    As result of these conditions,
15763                           <1> ;;    all kernel pages must be initialized as equal to
15764                           <1> ;;    physical layout for preventing page faults.
15765                           <1> ;;    Also, calling "allocate page" procedure after
15766                           <1> ;;    a page fault can cause another page fault (double fault)
15767                           <1> ;;    if all kernel page tables would not be initialized.
15768                           <1> ;;
15769                           <1> ;;    [first_page] = Beginning of users space, as offset to
15770                           <1> ;;    memory allocation table. (double word aligned)
15771                           <1> ;;
15772                           <1> ;;    [next_page] = first/next free space to be searched
15773                           <1> ;;    as offset to memory allocation table. (dw aligned)
15774                           <1> ;;
15775                           <1> ;;    [last_page] = End of memory (users space), as offset
15776                           <1> ;;    to memory allocation table. (double word aligned)
15777                           <1> ;;
15778                           <1> ;; USER PAGE TABLES:
15779                           <1> ;;    Demand paging (& 'copy on write' allocation method) ...
15780                           <1> ;;           'ready only' marked copies of the
15781                           <1> ;;          parent process's page table entries (for
15782                           <1> ;;           same physical memory).
15783                           <1> ;;           (A page will be copied to a new page after
15784                           <1> ;;            if it causes R/W page fault.)
15785                           <1> ;;
15786                           <1> ;;    Every user process has own (different)
15787                           <1> ;;    page directory and page tables.
15788                           <1> ;;
15789                           <1> ;;    Code starts at virtual address 0, always.
15790                           <1> ;;    (Initial value of EIP is 0 in user mode.)
15791                           <1> ;;    (Programs can be written/developed as simple
15792                           <1> ;;     flat memory programs.)
15793                           <1> ;;
15794                           <1> ;; MEMORY ALLOCATION STRATEGY:
15795                           <1> ;;    Memory page will be allocated by kernel only
15796                           <1> ;;           (in kernel/system mode only).
15797                           <1> ;;    * After a
15798                           <1> ;;      - 'not present' page fault
15799                           <1> ;;      - 'writing attempt on read only page' page fault
15800                           <1> ;;    * For loading (opening, reading) a file or disk/drive
15801                           <1> ;;    * As responce to 'allocate additional memory blocks'
15802                           <1> ;;      request by running process.
15803                           <1> ;;    * While creating a process, allocating a new buffer,
15804                           <1> ;;      new page tables etc.
15805                           <1> ;;
15806                           <1> ;;    At first,
15807                           <1> ;;    - 'allocate page' procedure will be called;
15808                           <1> ;,       if it will return with a valid (>0) physical address
15809                           <1> ;;       (that means the relevant M.A.T. bit has been RESET)
15810                           <1> ;;       relevant memory page/block will be cleared (zeroed).
15811                           <1> ;;    - 'allocate page' will be called for allocating page
15812                           <1> ;;       directory, page table and running space (data/code).
15813                           <1> ;;    - every successful 'allocate page' call will decrease
15814                           <1> ;;      'free_pages' count (pointer).
15815                           <1> ;;    - 'out of (insufficient) memory error' will be returned
15816                           <1> ;;      if 'free_pages' points to a ZERO.
15817                           <1> ;;    - swapping out and swapping in (if it is not a new page)
15818                           <1> ;;      procedures will be called as responce to 'out of memory'
15819                           <1> ;;      error except errors caused by attribute conflicts.
15820                           <1> ;;     (swapper functions)
15821                           <1> ;;
15822                           <1> ;;    At second,
15823                           <1> ;;    - page directory entry will be updated then page table
15824                           <1> ;;      entry will be updated.
15825                           <1> ;;
15826                           <1> ;; MEMORY ALLOCATION TABLE FORMAT:
15827                           <1> ;;    - M.A.T. has a size according to available memory as
15828                           <1> ;;      follows:
15829                           <1> ;;            - 1 (allocation) bit per 1 page (4096 bytes)
15830                           <1> ;;            - a bit with value of 0 means allocated page
15831                           <1> ;;            - a bit with value of 1 means a free page
15832                           <1> ;,    - 'free_pages' pointer holds count of free pages
15833                           <1> ;;      depending on M.A.T.
15834                           <1> ;;          (NOTE: Free page count will not be checked
15835                           <1> ;;          again -on M.A.T.- after initialization.
15836                           <1> ;;          Kernel will trust on initial count.)
15837                           <1> ;,    - 'free_pages' count will be decreased by allocation
15838                           <1> ;;      and it will be increased by deallocation procedures.
15839                           <1> ;;
15840                           <1> ;;    - Available memory will be calculated during
15841                           <1> ;;      the kernel's initialization stage (in real mode).
15842                           <1> ;;      Memory allocation table and kernel page tables
15843                           <1> ;;      will be formatted/sized as result of available
15844                           <1> ;;      memory calculation before paging is enabled.
15845                           <1> ;;
15846                           <1> ;; For 4GB Available/Present Memory: (max. possible memory size)
15847                           <1> ;;    - Memory Allocation Table size will be 128 KB.
15848                           <1> ;;    - Memory allocation for kernel page directory size
15849                           <1> ;;      is always 4 KB. (in addition to total allocation size
15850                           <1> ;;      for page tables)
15851                           <1> ;;    - Memory allocation for kernel page tables (1024 tables)
15852                           <1> ;;      is 4 MB (1024*4*1024 bytes).
15853                           <1> ;;    - User (available) space will be started
15854                           <1> ;;      at 6th MB of the memory (after 1MB+4MB).
15855                           <1> ;;    - The first 640 KB is for kernel's itself plus
15856                           <1> ;;      memory allocation table and kernel's page directory
15857                           <1> ;;      (D0000h-EFFFFh may be used as kernel space...)
15858                           <1> ;;    - B0000h to B7FFFh address space (32 KB) will be used
15859                           <1> ;;      for buffers.
```

```
15860                           <1> ;;    - ROMBIOS, VIDEO BUFFER and VIDEO ROM space are reserved.
15861                           <1> ;;      (A0000h-AFFFFh, C0000h-CFFFFh, F0000h-FFFFFh)
15862                           <1> ;;    - Kernel page tables start at 100000h (2nd MB)
15863                           <1> ;;
15864                           <1> ;; For 1GB Available Memory:
15865                           <1> ;;    - Memory Allocation Table size will be 32 KB.
15866                           <1> ;;    - Memory allocation for kernel page directory size
15867                           <1> ;;      is always 4 KB. (in addition to total allocation size
15868                           <1> ;;      for page tables)
15869                           <1> ;;    - Memory allocation for kernel page tables (256 tables)
15870                           <1> ;;      is 1 MB (256*4*1024 bytes).
15871                           <1> ;;    - User (available) space will be started
15872                           <1> ;;      at 3th MB of the memory (after 1MB+1MB).
15873                           <1> ;;    - The first 640 KB is for kernel's itself plus
15874                           <1> ;;      memory allocation table and kernel's page directory
15875                           <1> ;;      (D0000h-EFFFFh may be used as kernel space...)
15876                           <1> ;;    - B0000h to B7FFFh address space (32 KB) will be used
15877                           <1> ;;      for buffers.
15878                           <1> ;;    - ROMBIOS, VIDEO BUFFER and VIDEO ROM space are reserved.
15879                           <1> ;;      (A0000h-AFFFFh, C0000h-CFFFFh, F0000h-FFFFFh)
15880                           <1> ;;    - Kernel page tables start at 100000h (2nd MB).
15881                           <1> ;;
15882                           <1> ;;
15883                           <1>
15884                           <1>
15885                           <1> ;;*****************************************************************************
15886                           <1> ;;
15887                           <1> ;; RETRO UNIX 386 v1 - Paging (Method for Copy On Write paging principle)
15888                           <1> ;; DEMAND PAGING - PARENT&CHILD PAGE TABLE DUPLICATION PRINCIPLES (23/04/2015)
15889                           <1>
15890                           <1> ;; Main factor: "sys fork" system call
15891                           <1> ;;
15892                           <1> ;;          FORK
15893                           <1> ;;                      |----> parent - duplicated PTEs, read only pages
15894                           <1> ;;  writable pages ---->|
15895                           <1> ;;                      |----> child - duplicated PTEs, read only pages
15896                           <1> ;;
15897                           <1> ;; AVL bit (0) of Page Table Entry is used as duplication sign
15898                           <1> ;;
15899                           <1> ;; AVL Bit 0 [PTE Bit 9] = 'Duplicated PTE belongs to child' sign/flag (if it is set)
15900                           <1> ;; Note: Dirty bit (PTE bit 6) may be used instead of AVL bit 0 (PTE bit 9)
15901                           <1> ;;       -while R/W bit is 0-.
15902                           <1> ;;
15903                           <1> ;; Duplicate page tables with writable pages (the 1st sys fork in the process):
15904                           <1> ;; # Parent's Page Table Entries are updated to point same pages as read only,
15905                           <1> ;;   as duplicated PTE bit  -AVL bit 0, PTE bit 9- are reset/clear.
15906                           <1> ;; # Then Parent's Page Table is copied to Child's Page Table.
15907                           <1> ;; # Child's Page Table Entries are updated as duplicated child bit
15908                           <1> ;;   -AVL bit 0, PTE bit 9- is set.
15909                           <1> ;;
15910                           <1> ;; Duplicate page tables with read only pages (several sys fork system calls):
15911                           <1> ;; # Parent's read only pages are copied to new child pages.
15912                           <1> ;;   Parent's PTE attributes are not changed.
15913                           <1> ;;   (Because, there is another parent-child fork before this fork! We must not
15914                           <1> ;;    destroy/mix previous fork result).
15915                           <1> ;; # Child's Page Table Entries (which are corresponding to Parent's
15916                           <1> ;;   read only pages) are set as writable (while duplicated PTE bit is clear).
15917                           <1> ;; # Parent's PTEs with writable page attribute are updated to point same pages
15918                           <1> ;;   as read only, (while) duplicated PTE bit is reset (clear).
15919                           <1> ;; # Parent's Page Table Entries (with writable page attribute) are duplicated
15920                           <1> ;;   as Child's Page Table Entries without copying actual page.
15921                           <1> ;; # Child 's Page Table Entries (which are corresponding to Parent's writable
15922                           <1> ;;   pages) are updated as duplicated PTE bit (AVL bit 0, PTE bit 9- is set.
15923                           <1> ;;
15924                           <1> ;; !? WHAT FOR (duplication after duplication):
15925                           <1> ;; In UNIX method for sys fork (a typical 'fork' application in /etc/init)
15926                           <1> ;; program/executable code continues from specified location as child process,
15927                           <1> ;; returns back previous code location as parent process, every child after
15928                           <1> ;; every sys fork uses last image of code and data just prior the fork.
15929                           <1> ;; Even if the parent code changes data, the child will not see the changed data
15930                           <1> ;; after the fork. In Retro UNIX 8086 v1, parent's process segment (32KB)
15931                           <1> ;; was copied to child's process segment (all of code and data) according to
15932                           <1> ;; original UNIX v1 which copies all of parent process code and data -core-
15933                           <1> ;; to child space -core- but swaps that core image -of child- on to disk.
15934                           <1> ;; If I (Erdogan Tan) would use a method of to copy parent's core
15935                           <1> ;; (complete running image of parent process) to the child process;
15936                           <1> ;; for big sizes, i would force Retro UNIX 386 v1 to spend many memory pages
15937                           <1> ;; and times only for a sys fork. (It would excessive reservation for sys fork,
15938                           <1> ;; because sys fork usually is prior to sys exec; sys exec always establishes
15939                           <1> ;; a new/fresh core -running space-, by clearing all code/data content).
15940                           <1> ;; 'Read Only' page flag ensures page fault handler is needed only for a few write
15941                           <1> ;; attempts between sys fork and sys exec, not more... (I say so by thinking
15942                           <1> ;; of "/etc/init" content, specially.) sys exec will clear page tables and
15943                           <1> ;; new/fresh pages will be used to load and run new executable/program.
15944                           <1> ;; That is what for i have preferred "copy on write", "duplication" method
15945                           <1> ;; for sharing same read only pages between parent and child processes.
15946                           <1> ;; That is a pitty i have to use new private flag (AVL bit 0, "duplicated PTE
15947                           <1> ;; belongs to child" sign) for cooperation on duplicated pages between a parent
15948                           <1> ;; and it's child processes; otherwise parent process would destroy data belongs
15949                           <1> ;; to its child or vice versa; or some pages would remain unclaimed
15950                           <1> ;; -deallocation problem-.
15951                           <1> ;; Note: to prevent conflicts, read only pages must not be swapped out...
15952                           <1> ;;
15953                           <1> ;; WHEN PARENT 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 reset/clear, there is a child uses same page.
15957                           <1> ;;   - Parent's read only page -previous page- is copied to a new writable page.
15958                           <1> ;;   - Parent's PTE is updated as writable page, as unique page (AVL=0)
15959                           <1> ;;   - (Page fault handler whill check this PTE later, if child process causes to
15960                           <1> ;;     page fault due to write attempt on read only page. Of course, the previous
15961                           <1> ;;     read only page will be converted to writable and unique page which belongs
15962                           <1> ;;     to child process.)
```

```
                        <1> ;; WHEN CHILD TRIES TO WRITE IT'S READ ONLY (DUPLICATED) PAGE:
                        <1> ;; # Page fault handler will do those:
                        <1> ;;   - 'Duplicated PTE' flag (PTE bit 9) is checked (on the failed PTE).
                        <1> ;;   - If it is set, there is a parent uses -or was using- same page.
                        <1> ;;   - Same PTE address within parent's page table is checked if it has same page
                        <1> ;;     address or not.
                        <1> ;;   - If parent's PTE has same address, child will continue with a new writable page.
                        <1> ;;     Parent's PTE will point to same (previous) page as writable, unique (AVL=0).
                        <1> ;;   - If parent's PTE has different address, child will continue with it's
                        <1> ;;     own/same page but read only flag (0) will be changed to writable flag (1) and
                        <1> ;;     'duplicated PTE (belongs to child)' flag/sign will be cleared/reset.
                        <1> ;;
                        <1> ;; NOTE: When a child process is terminated, read only flags of parent's page tables
                        <1> ;;       will be set as writable (and unique) in case of child process was using
                        <1> ;;       same pages with duplicated child PTE sign... Depending on sys fork and
                        <1> ;;       duplication method details, it is not possible multiple child processes
                        <1> ;;       were using same page with duplicated PTEs.
                        <1> ;;
                        <1> ;;////////////////////////////////////////////////////////////////////////////////
                        <1>
                        <1> ;; 08/10/2014
                        <1> ;; 11/09/2014 - Retro UNIX 386 v1 PAGING (further) draft
                        <1> ;;          by Erdogan Tan (Based on KolibriOS 'memory.inc')
                        <1>
                        <1> ;; 'allocate_page' code is derived and modified from KolibriOS
                        <1> ;; 'alloc_page' procedure in 'memory.inc'
                        <1> ;; (25/08/2014, Revision: 5057) file
                        <1> ;; by KolibriOS Team (2004-2012)
                        <1>
                        <1> allocate_page:
                        <1>       ; 01/07/2015
                        <1>       ; 05/05/2015
                        <1>       ; 30/04/2015
                        <1>       ; 16/10/2014
                        <1>       ; 08/10/2014
                        <1>       ; 09/09/2014 (Retro UNIX 386 v1 - beginning)
                        <1>       ;
                        <1>       ; INPUT -> none
                        <1>       ;
                        <1>       ; OUTPUT ->
                        <1>       ;     EAX = PHYSICAL (real/flat) ADDRESS OF THE ALLOCATED PAGE
                        <1>       ;     (corresponding MEMORY ALLOCATION TABLE bit is RESET)
                        <1>       ;
                        <1>       ;     CF = 1 and EAX = 0
                        <1>       ;              if there is not a free page to be allocated
                        <1>       ;
                        <1>       ; Modified Registers -> none (except EAX)
                        <1>       ;
00004B7A A1[28520100]   <1>       mov   eax, [free_pages]
00004B7F 21C0           <1>       and   eax, eax
00004B81 7438           <1>       jz    short out_of_memory
                        <1>       ;
00004B83 53             <1>       push  ebx
00004B84 51             <1>       push  ecx
                        <1>       ;
00004B85 BB00001000     <1>       mov   ebx, MEM_ALLOC_TBL   ; Memory Allocation Table offset
00004B8A 89D9           <1>       mov   ecx, ebx
                        <1>                                  ; NOTE: 32 (first_page) is initial
                        <1>                                  ; value of [next_page].
                        <1>                                  ; It points to the first available
                        <1>                                  ; page block for users (ring 3) ...
                        <1>                                  ; (MAT offset 32 = 1024/32)
                        <1>                                  ; (at the of the first 4 MB)
00004B8C 031D[2C520100] <1>       add   ebx, [next_page] ; Free page searching starts from here
                        <1>                              ; next_free_page >> 5
00004B92 030D[30520100] <1>       add   ecx, [last_page] ; Free page searching ends here
                        <1>                              ; (total_pages - 1) >> 5
                        <1> al_p_scan:
00004B98 39CB           <1>       cmp   ebx, ecx
00004B9A 770A           <1>       ja    short al_p_notfound
                        <1>       ;
                        <1>       ; 01/07/2015
                        <1>       ; AMD64 Architecture Programmer's Manual
                        <1>       ; Volume 3:
                        <1>       ; General-Purpose and System Instructions
                        <1>       ;
                        <1>       ; BSF - Bit Scan Forward
                        <1>       ;
                        <1>       ;   Searches the value in a register or a memory location
                        <1>       ;   (second operand) for the least-significant set bit.
                        <1>       ;   If a set bit is found, the instruction clears the zero flag (ZF)
                        <1>       ;   and stores the index of the least-significant set bit in a destination
                        <1>       ;   register (first operand). If the second operand contains 0,
                        <1>       ;   the instruction sets ZF to 1 and does not change the contents of the
                        <1>       ;   destination register. The bit index is an unsigned offset from bit 0
                        <1>       ;   of the searched value
                        <1>       ;
00004B9C 0FBC03         <1>       bsf   eax, [ebx] ; Scans source operand for first bit set (1).
                        <1>                        ; Clear ZF if a bit is found set (1) and
                        <1>                        ; loads the destination with an index to
                        <1>                        ; first set bit. (0 -> 31)
                        <1>                        ; Sets ZF to 1 if no bits are found set.
00004B9F 7525           <1>       jnz   short al_p_found ; ZF = 0 -> a free page has been found
                        <1>                      ;
                        <1>                      ; NOTE:  a Memory Allocation Table bit
                        <1>                      ;       with value of 1 means
                        <1>                      ;       the corresponding page is free
                        <1>                      ;       (Retro UNIX 386 v1 feature only!)
00004BA1 83C304         <1>       add   ebx, 4
                        <1>                      ; We return back for searching next page block
                        <1>                      ; NOTE: [free_pages] is not ZERO; so,
                        <1>                      ;     we always will find at least 1 free page here.
00004BA4 EBF2           <1>         jmp     short al_p_scan
```

```
16066                                  <1>       ;
16067                                  <1> al_p_notfound:
16068 00004BA6 81E900001000            <1>       sub    ecx, MEM_ALLOC_TBL
16069 00004BAC 890D[2C520100]          <1>       mov    [next_page], ecx ; next/first free page = last page
16070                                  <1>                               ; (deallocate_page procedure will change it)
16071 00004BB2 31C0                    <1>       xor    eax, eax
16072 00004BB4 A3[28520100]            <1>       mov    [free_pages], eax ; 0
16073 00004BB9 59                      <1>       pop    ecx
16074 00004BBA 5B                      <1>       pop    ebx
16075                                  <1>       ;
16076                                  <1> out_of_memory:
16077 00004BBB E85B040000              <1>       call   swap_out
16078 00004BC0 7325                    <1>       jnc    short al_p_ok  ; [free_pages] = 0, re-allocation by swap_out
16079                                  <1>       ;
16080 00004BC2 29C0                    <1>       sub    eax, eax ; 0
16081 00004BC4 F9                      <1>       stc
16082 00004BC5 C3                      <1>       retn
16083                                  <1>
16084                                  <1> al_p_found:
16085 00004BC6 89D9                    <1>       mov    ecx, ebx
16086 00004BC8 81E900001000            <1>       sub    ecx, MEM_ALLOC_TBL
16087 00004BCE 890D[2C520100]          <1>       mov    [next_page], ecx ; Set first free page searching start
16088                                  <1>                               ; address/offset (to the next)
16089 00004BD4 FF0D[28520100]          <1>         dec     dword [free_pages] ; 1 page has been allocated (X = X-1)
16090                                  <1>       ;
16091 00004BDA 0FB303                  <1>       btr    [ebx], eax     ; The destination bit indexed by the source value
16092                                  <1>                               ; is copied into the Carry Flag and then cleared
16093                                  <1>                               ; in the destination.
16094                                  <1>                               ;
16095                                  <1>                               ; Reset the bit which is corresponding to the
16096                                  <1>                               ; (just) allocated page.
16097                                  <1>       ; 01/07/2015 (4*8 = 32, 1 allocation byte = 8 pages)
16098 00004BDD C1E103                  <1>       shl    ecx, 3         ; (page block offset * 32) + page index
16099 00004BE0 01C8                    <1>       add    eax, ecx       ; = page number
16100 00004BE2 C1E00C                  <1>       shl    eax, 12              ; physical address of the page (flat/real value)
16101                                  <1>       ; EAX = physical address of memory page
16102                                  <1>       ;
16103                                  <1>       ; NOTE: The relevant page directory and page table entry will be updated
16104                                  <1>       ;       according to this EAX value...
16105 00004BE5 59                      <1>       pop    ecx
16106 00004BE6 5B                      <1>       pop    ebx
16107                                  <1> al_p_ok:
16108 00004BE7 C3                      <1>       retn
16109                                  <1>
16110                                  <1>
16111                                  <1> make_page_dir:
16112                                  <1>       ; 18/04/2015
16113                                  <1>       ; 12/04/2015
16114                                  <1>       ; 23/10/2014
16115                                  <1>       ; 16/10/2014
16116                                  <1>       ; 09/10/2014 ; (Retro UNIX 386 v1 - beginning)
16117                                  <1>       ;
16118                                  <1>       ; INPUT ->
16119                                  <1>       ;      none
16120                                  <1>       ; OUTPUT ->
16121                                  <1>       ;      (EAX = 0)
16122                                  <1>       ;      cf = 1 -> insufficient (out of) memory error
16123                                  <1>       ;      cf = 0 ->
16124                                  <1>       ;      u.pgdir = page directory (physical) address of the current
16125                                  <1>       ;               process/user.
16126                                  <1>       ;
16127                                  <1>       ; Modified Registers -> EAX
16128                                  <1>       ;
16129 00004BE8 E88DFFFFFF              <1>       call   allocate_page
16130 00004BED 7216                    <1>       jc     short mkpd_error
16131                                  <1>       ;
16132 00004BEF A3[B8030300]            <1>       mov    [u.pgdir], eax    ; Page dir address for current user/process
16133                                  <1>                                ; (Physical address)
16134                                  <1> clear_page:
16135                                  <1>       ; 18/04/2015
16136                                  <1>       ; 09/10/2014 ; (Retro UNIX 386 v1 - beginning)
16137                                  <1>       ;
16138                                  <1>       ; INPUT ->
16139                                  <1>       ;      EAX = physical address of the page
16140                                  <1>       ; OUTPUT ->
16141                                  <1>       ;      all bytes of the page will be cleared
16142                                  <1>       ;
16143                                  <1>       ; Modified Registers -> none
16144                                  <1>       ;
16145 00004BF4 57                      <1>       push   edi
16146 00004BF5 51                      <1>       push   ecx
16147 00004BF6 50                      <1>       push   eax
16148 00004BF7 B900040000              <1>       mov    ecx, PAGE_SIZE / 4
16149 00004BFC 89C7                    <1>       mov    edi, eax
16150 00004BFE 31C0                    <1>       xor    eax, eax
16151 00004C00 F3AB                    <1>       rep    stosd
16152 00004C02 58                      <1>       pop    eax
16153 00004C03 59                      <1>       pop    ecx
16154 00004C04 5F                      <1>       pop    edi
16155                                  <1> mkpd_error:
16156                                  <1> mkpt_error:
16157 00004C05 C3                      <1>       retn
16158                                  <1>
16159                                  <1> make_page_table:
16160                                  <1>       ; 23/06/2015
16161                                  <1>       ; 18/04/2015
16162                                  <1>       ; 12/04/2015
16163                                  <1>       ; 16/10/2014
16164                                  <1>       ; 09/10/2014 ; (Retro UNIX 386 v1 - beginning)
16165                                  <1>       ;
16166                                  <1>       ; INPUT ->
16167                                  <1>       ;      EBX = virtual (linear) address
16168                                  <1>       ;      ECX = page table attributes (lower 12 bits)
```

```
16169                                  <1>       ;           (higher 20 bits must be ZERO)
16170                                  <1>       ;           (bit 0 must be 1)
16171                                  <1>       ;     u.pgdir = page directory (physical) address
16172                                  <1>       ; OUTPUT ->
16173                                  <1>       ;     EDX = Page directory entry address
16174                                  <1>       ;     EAX = Page table address
16175                                  <1>       ;     cf = 1 -> insufficient (out of) memory error
16176                                  <1>       ;     cf = 0 -> page table address in the PDE (EDX)
16177                                  <1>       ;
16178                                  <1>       ; Modified Registers -> EAX, EDX
16179                                  <1>       ;
16180 00004C06 E86FFFFFFF             <1>       call  allocate_page
16181 00004C0B 72F8                   <1>       jc    short mkpt_error
16182 00004C0D E811000000             <1>       call  set_pde
16183 00004C12 EBE0                   <1>       jmp   short clear_page
16184                                  <1>
16185                                  <1> make_page:
16186                                  <1>       ; 24/07/2015
16187                                  <1>       ; 23/06/2015 ; (Retro UNIX 386 v1 - beginning)
16188                                  <1>       ;
16189                                  <1>       ; INPUT ->
16190                                  <1>       ;     EBX = virtual (linear) address
16191                                  <1>       ;     ECX = page attributes (lower 12 bits)
16192                                  <1>       ;           (higher 20 bits must be ZERO)
16193                                  <1>       ;           (bit 0 must be 1)
16194                                  <1>       ;     u.pgdir = page directory (physical) address
16195                                  <1>       ; OUTPUT ->
16196                                  <1>       ;     EBX = Virtual address
16197                                  <1>       ;     (EDX = PTE value)
16198                                  <1>       ;     EAX = Physical address
16199                                  <1>       ;     cf = 1 -> insufficient (out of) memory error
16200                                  <1>       ;
16201                                  <1>       ; Modified Registers -> EAX, EDX
16202                                  <1>       ;
16203 00004C14 E861FFFFFF             <1>       call  allocate_page
16204 00004C19 7207                   <1>       jc    short mkp_err
16205 00004C1B E821000000             <1>       call  set_pte
16206 00004C20 73D2                   <1>       jnc   short clear_page ; 18/04/2015
16207                                  <1> mkp_err:
16208 00004C22 C3                     <1>       retn
16209                                  <1>
16210                                  <1>
16211                                  <1> set_pde:    ; Set page directory entry (PDE)
16212                                  <1>       ; 20/07/2015
16213                                  <1>       ; 18/04/2015
16214                                  <1>       ; 12/04/2015
16215                                  <1>       ; 23/10/2014
16216                                  <1>       ; 10/10/2014 ; (Retro UNIX 386 v1 - beginning)
16217                                  <1>       ;
16218                                  <1>       ; INPUT ->
16219                                  <1>       ;     EAX = physical address
16220                                  <1>       ;           (use present value if EAX = 0)
16221                                  <1>       ;     EBX = virtual (linear) address
16222                                  <1>       ;     ECX = page table attributes (lower 12 bits)
16223                                  <1>       ;           (higher 20 bits must be ZERO)
16224                                  <1>       ;           (bit 0 must be 1)
16225                                  <1>       ;     u.pgdir = page directory (physical) address
16226                                  <1>       ; OUTPUT ->
16227                                  <1>       ;     EDX = PDE address
16228                                  <1>       ;     EAX = page table address (physical)
16229                                  <1>       ;     ;(CF=1 -> Invalid page address)
16230                                  <1>       ;
16231                                  <1>       ; Modified Registers -> EDX
16232                                  <1>       ;
16233 00004C23 89DA                   <1>       mov   edx, ebx
16234 00004C25 C1EA16                 <1>       shr   edx, PAGE_D_SHIFT ; 22
16235 00004C28 C1E202                 <1>       shl   edx, 2 ; offset to page directory (1024*4)
16236 00004C2B 0315[B8030300]         <1>       add   edx, [u.pgdir]
16237                                  <1>       ;
16238 00004C31 21C0                   <1>       and   eax, eax
16239 00004C33 7506                   <1>       jnz   short spde_1
16240                                  <1>       ;
16241 00004C35 8B02                   <1>       mov   eax, [edx]  ; old PDE value
16242                                  <1>       ;test  al, 1
16243                                  <1>       ;jz   short spde_2
16244 00004C37 662500F0               <1>       and   ax, PDE_A_CLEAR ; 0F000h  ; clear lower 12 bits
16245                                  <1> spde_1:
16246                                  <1>       ;and  cx, 0FFFh
16247 00004C3B 8902                   <1>       mov   [edx], eax
16248 00004C3D 66090A                 <1>       or    [edx], cx
16249 00004C40 C3                     <1>       retn
16250                                  <1> ;spde_2: ; error
16251                                  <1> ;     stc
16252                                  <1> ;     retn
16253                                  <1>
16254                                  <1> set_pte:    ; Set page table entry (PTE)
16255                                  <1>       ; 24/07/2015
16256                                  <1>       ; 20/07/2015
16257                                  <1>       ; 23/06/2015
16258                                  <1>       ; 18/04/2015
16259                                  <1>       ; 12/04/2015
16260                                  <1>       ; 10/10/2014 ; (Retro UNIX 386 v1 - beginning)
16261                                  <1>       ;
16262                                  <1>       ; INPUT ->
16263                                  <1>       ;     EAX = physical page address
16264                                  <1>       ;           (use present value if EAX = 0)
16265                                  <1>       ;     EBX = virtual (linear) address
16266                                  <1>       ;     ECX = page attributes (lower 12 bits)
16267                                  <1>       ;           (higher 20 bits must be ZERO)
16268                                  <1>       ;           (bit 0 must be 1)
16269                                  <1>       ;     u.pgdir = page directory (physical) address
16270                                  <1>       ; OUTPUT ->
16271                                  <1>       ;     EAX = physical page address
```

```
16272                                  <1>       ;     (EDX = PTE value)
16273                                  <1>       ;     EBX = virtual address
16274                                  <1>       ;
16275                                  <1>       ;     CF = 1 -> error
16276                                  <1>       ;
16277                                  <1>       ; Modified Registers -> EAX, EDX
16278                                  <1>       ;
16279 00004C41 50                     <1>       push  eax
16280 00004C42 A1[B8030300]           <1>       mov   eax, [u.pgdir] ; 20/07/2015
16281 00004C47 E837000000             <1>       call  get_pde
16282                                  <1>             ; EDX = PDE address
16283                                  <1>             ; EAX = PDE value
16284 00004C4C 5A                     <1>       pop   edx ; physical page address
16285 00004C4D 722A                   <1>       jc    short spte_err ; PDE not present
16286                                  <1>       ;
16287 00004C4F 53                     <1>       push  ebx ; 24/07/2015
16288 00004C50 662500F0               <1>       and   ax, PDE_A_CLEAR ; 0F000h ; clear lower 12 bits
16289                                  <1>                             ; EDX = PT address (physical)
16290 00004C54 C1EB0C                 <1>       shr   ebx, PAGE_SHIFT ; 12
16291 00004C57 81E3FF030000           <1>       and   ebx, PTE_MASK; 03FFh
16292                                  <1>                            ; clear higher 10 bits (PD bits)
16293 00004C5D C1E302                 <1>       shl   ebx, 2   ; offset to page table (1024*4)
16294 00004C60 01C3                   <1>       add   ebx, eax
16295                                  <1>       ;
16296 00004C62 8B03                   <1>       mov   eax, [ebx] ; Old PTE value
16297 00004C64 A801                   <1>       test  al, 1
16298 00004C66 740C                   <1>       jz    short spte_0
16299 00004C68 09D2                   <1>       or    edx, edx
16300 00004C6A 750F                   <1>       jnz   short spte_1
16301 00004C6C 662500F0               <1>       and   ax, PTE_A_CLEAR ; 0F000h ; clear lower 12 bits
16302 00004C70 89C2                   <1>       mov   edx, eax
16303 00004C72 EB09                   <1>       jmp   short spte_2
16304                                  <1> spte_0:
16305                                  <1>       ; If this PTE contains a swap (disk) address,
16306                                  <1>       ; it can be updated by using 'swap_in' procedure
16307                                  <1>       ; only!
16308 00004C74 21C0                   <1>       and   eax, eax
16309 00004C76 7403                   <1>       jz    short spte_1
16310                                  <1>       ; 24/07/2015
16311                                  <1>       ; swapped page ! (on disk)
16312 00004C78 5B                     <1>       pop   ebx
16313                                  <1> spte_err:
16314 00004C79 F9                     <1>       stc
16315 00004C7A C3                     <1>       retn
16316                                  <1> spte_1:
16317 00004C7B 89D0                   <1>       mov   eax, edx
16318                                  <1> spte_2:
16319 00004C7D 09CA                   <1>       or    edx, ecx
16320                                  <1>       ; 23/06/2015
16321 00004C7F 8913                   <1>       mov   [ebx], edx ; PTE value in EDX
16322                                  <1>       ; 24/07/2015
16323 00004C81 5B                     <1>       pop   ebx
16324 00004C82 C3                     <1>       retn
16325                                  <1>
16326                                  <1> get_pde:    ; Get present value of the relevant PDE
16327                                  <1>       ; 20/07/2015
16328                                  <1>       ; 18/04/2015
16329                                  <1>       ; 12/04/2015
16330                                  <1>       ; 10/10/2014 ; (Retro UNIX 386 v1 - beginning)
16331                                  <1>       ;
16332                                  <1>       ; INPUT ->
16333                                  <1>       ;     EBX = virtual (linear) address
16334                                  <1>       ;     EAX = page directory (physical) address
16335                                  <1>       ; OUTPUT ->
16336                                  <1>       ;     EDX = Page directory entry address
16337                                  <1>       ;     EAX = Page directory entry value
16338                                  <1>       ;     CF = 1 -> PDE not present or invalid ?
16339                                  <1>       ; Modified Registers -> EDX, EAX
16340                                  <1>       ;
16341 00004C83 89DA                   <1>       mov   edx, ebx
16342 00004C85 C1EA16                 <1>       shr   edx, PAGE_D_SHIFT ; 22  (12+10)
16343 00004C88 C1E202                 <1>       shl   edx, 2 ; offset to page directory (1024*4)
16344 00004C8B 01C2                   <1>       add   edx, eax ; page directory address (physical)
16345 00004C8D 8B02                   <1>       mov   eax, [edx]
16346 00004C8F A801                   <1>       test  al, PDE_A_PRESENT ; page table is present or not !
16347 00004C91 751F                   <1>       jnz   short gpte_retn
16348 00004C93 F9                     <1>       stc
16349                                  <1> gpde_retn:
16350 00004C94 C3                     <1>       retn
16351                                  <1>
16352                                  <1> get_pte:
16353                                  <1>             ; Get present value of the relevant PTE
16354                                  <1>       ; 29/07/2015
16355                                  <1>       ; 20/07/2015
16356                                  <1>       ; 18/04/2015
16357                                  <1>       ; 12/04/2015
16358                                  <1>       ; 10/10/2014 ; (Retro UNIX 386 v1 - beginning)
16359                                  <1>       ;
16360                                  <1>       ; INPUT ->
16361                                  <1>       ;     EBX = virtual (linear) address
16362                                  <1>       ;     EAX = page directory (physical) address
16363                                  <1>       ; OUTPUT ->
16364                                  <1>       ;     EDX = Page table entry address (if CF=0)
16365                                  <1>       ;           Page directory entry address (if CF=1)
16366                                  <1>       ;            (Bit 0 value is 0 if PT is not present)
16367                                  <1>       ;     EAX = Page table entry value (page address)
16368                                  <1>       ;     CF = 1 -> PDE not present or invalid ?
16369                                  <1>       ; Modified Registers -> EAX, EDX
16370                                  <1>       ;
16371 00004C95 E8E9FFFFFF             <1>       call  get_pde
16372 00004C9A 72F8                   <1>       jc    short gpde_retn    ; page table is not present
16373                                  <1>       ;jnc  short gpte_1
16374                                  <1>       ;retn
```

```
16375                                  <1> ;gpte_1:
16376 00004C9C 662500F0                <1>         and    ax, PDE_A_CLEAR ; 0F000h ; clear lower 12 bits
16377 00004CA0 89DA                    <1>         mov    edx, ebx
16378 00004CA2 C1EA0C                  <1>         shr    edx, PAGE_SHIFT ; 12
16379 00004CA5 81E2FF030000            <1>         and    edx, PTE_MASK; 03FFh
16380                                  <1>                       ; clear higher 10 bits (PD bits)
16381 00004CAB C1E202                  <1>         shl    edx, 2 ; offset from start of page table (1024*4)
16382 00004CAE 01C2                    <1>         add    edx, eax
16383 00004CB0 8B02                    <1>         mov    eax, [edx]
16384                                  <1> gpte_retn:
16385 00004CB2 C3                      <1>         retn
16386                                  <1>
16387                                  <1> deallocate_page_dir:
16388                                  <1>         ; 15/09/2015
16389                                  <1>         ; 05/08/2015
16390                                  <1>         ; 30/04/2015
16391                                  <1>         ; 28/04/2015
16392                                  <1>         ; 17/10/2014
16393                                  <1>         ; 12/10/2014 (Retro UNIX 386 v1 - beginning)
16394                                  <1>         ;
16395                                  <1>         ; INPUT ->
16396                                  <1>         ;     EAX = PHYSICAL ADDRESS OF THE PAGE DIRECTORY (CHILD)
16397                                  <1>         ;     EBX = PHYSICAL ADDRESS OF THE PARENT'S PAGE DIRECTORY
16398                                  <1>         ; OUTPUT ->
16399                                  <1>         ;     All of page tables in the page directory
16400                                  <1>         ;     and page dir's itself will be deallocated
16401                                  <1>         ;     except 'read only' duplicated pages (will be converted
16402                                  <1>         ;     to writable pages).
16403                                  <1>         ;
16404                                  <1>         ; Modified Registers -> EAX
16405                                  <1>         ;
16406                                  <1>         ;
16407 00004CB3 56                      <1>         push   esi
16408 00004CB4 51                      <1>         push   ecx
16409 00004CB5 50                      <1>         push   eax
16410 00004CB6 89C6                    <1>         mov    esi, eax
16411 00004CB8 31C9                    <1>         xor    ecx, ecx
16412                                  <1>         ; The 1st PDE points to Kernel Page Table 0 (the 1st 4MB),
16413                                  <1>         ; it must not be deallocated
16414 00004CBA 890E                    <1>         mov    [esi], ecx ; 0 ; clear PDE 0
16415                                  <1> dapd_0:
16416 00004CBC AD                      <1>         lodsd
16417 00004CBD A801                    <1>         test   al, PDE_A_PRESENT ; bit 0, present flag (must be 1)
16418 00004CBF 7409                    <1>         jz     short dapd_1
16419 00004CC1 662500F0                <1>         and    ax, PDE_A_CLEAR ; 0F000h ; clear lower 12 (attribute) bits
16420 00004CC5 E812000000              <1>         call   deallocate_page_table
16421                                  <1> dapd_1:
16422 00004CCA 41                      <1>         inc    ecx ; page directory entry index
16423 00004CCB 81F900040000            <1>         cmp    ecx, PAGE_SIZE / 4 ; 1024
16424 00004CD1 72E9                    <1>         jb     short dapd_0
16425                                  <1> dapd_2:
16426 00004CD3 58                      <1>         pop    eax
16427 00004CD4 E87F000000              <1>         call   deallocate_page     ; deallocate the page dir's itself
16428 00004CD9 59                      <1>         pop    ecx
16429 00004CDA 5E                      <1>         pop    esi
16430 00004CDB C3                      <1>         retn
16431                                  <1>
16432                                  <1> deallocate_page_table:
16433                                  <1>         ; 12/07/2016
16434                                  <1>         ; 19/09/2015
16435                                  <1>         ; 15/09/2015
16436                                  <1>         ; 05/08/2015
16437                                  <1>         ; 30/04/2015
16438                                  <1>         ; 28/04/2015
16439                                  <1>         ; 24/10/2014
16440                                  <1>         ; 23/10/2014
16441                                  <1>         ; 12/10/2014 (Retro UNIX 386 v1 - beginning)
16442                                  <1>         ;
16443                                  <1>         ; INPUT ->
16444                                  <1>         ;     EAX = PHYSICAL (real/flat) ADDRESS OF THE PAGE TABLE
16445                                  <1>         ;     EBX = PHYSICAL ADDRESS OF THE PARENT'S PAGE DIRECTORY
16446                                  <1>         ;     (ECX = page directory entry index)
16447                                  <1>         ; OUTPUT ->
16448                                  <1>         ;     All of pages in the page table and page table's itself
16449                                  <1>         ;     will be deallocated except 'read only' duplicated pages
16450                                  <1>         ;     (will be converted to writable pages).
16451                                  <1>         ;
16452                                  <1>         ; Modified Registers -> EAX
16453                                  <1>         ;
16454 00004CDC 56                      <1>         push   esi
16455 00004CDD 57                      <1>         push   edi
16456 00004CDE 52                      <1>         push   edx
16457 00004CDF 50                      <1>         push   eax ; *
16458 00004CE0 89C6                    <1>         mov    esi, eax
16459 00004CE2 31FF                    <1>         xor    edi, edi ; 0
16460                                  <1> dapt_0:
16461 00004CE4 AD                      <1>         lodsd
16462 00004CE5 A801                    <1>         test   al, PTE_A_PRESENT ; bit 0, present flag (must be 1)
16463 00004CE7 7441                    <1>         jz     short dapt_1
16464                                  <1>         ;
16465 00004CE9 A802                    <1>         test   al, PTE_A_WRITE   ; bit 1, writable (r/w) flag
16466                                  <1>                                  ; (must be 1)
16467 00004CEB 754C                    <1>         jnz    short dapt_3
16468                                  <1>         ; Read only -duplicated- page (belongs to a parent or a child)
16469 00004CED 66A90002                <1>           test    ax, PTE_DUPLICATED ; Was this page duplicated
16470                                  <1>                                    ; as child's page ?
16471 00004CF1 7451                    <1>         jz     short dapt_4 ; Clear PTE but don't deallocate the page!
16472                                  <1>         ; check the parent's PTE value is read only & same page or not..
16473                                  <1>         ; ECX = page directory entry index (0-1023)
16474 00004CF3 53                      <1>         push   ebx
16475 00004CF4 51                      <1>         push   ecx
16476 00004CF5 66C1E102                <1>         shl    cx, 2 ; *4
16477 00004CF9 01CB                    <1>         add    ebx, ecx ; PDE offset (for the parent)
```

```
16478 00004CFB 8B0B              <1>       mov   ecx, [ebx]
16479 00004CFD F6C101            <1>       test  cl, PDE_A_PRESENT ; present (valid) or not ?
16480 00004D00 7435              <1>       jz    short dapt_2 ; parent process does not use this page
16481 00004D02 6681E100F0        <1>       and   cx, PDE_A_CLEAR ; 0F000h ; Clear attribute bits
16482                            <1>       ; EDI = page table entry index (0-1023)
16483 00004D07 89FA              <1>       mov   edx, edi
16484 00004D09 66C1E202          <1>       shl   dx, 2 ; *4
16485 00004D0D 01CA              <1>       add   edx, ecx ; PTE offset (for the parent)
16486 00004D0F 8B1A              <1>       mov   ebx, [edx]
16487 00004D11 F6C301            <1>       test  bl, PTE_A_PRESENT ; present or not ?
16488 00004D14 7421              <1>       jz    short dapt_2 ; parent process does not use this page
16489 00004D16 662500F0          <1>       and   ax, PTE_A_CLEAR ; 0F000h ; Clear attribute bits
16490 00004D1A 6681E300F0        <1>       and   bx, PTE_A_CLEAR ; 0F000h ; Clear attribute bits
16491 00004D1F 39D8              <1>       cmp   eax, ebx     ; parent's and child's pages are same ?
16492 00004D21 7514              <1>       jne   short dapt_2 ; not same page
16493                            <1>                          ; deallocate the child's page
16494 00004D23 800A02            <1>         or     byte [edx], PTE_A_WRITE ; convert to writable page (parent)
16495 00004D26 59                <1>       pop   ecx
16496 00004D27 5B                <1>       pop   ebx
16497 00004D28 EB1A              <1>       jmp   short dapt_4
16498                            <1> dapt_1:
16499 00004D2A 09C0              <1>       or    eax, eax     ; swapped page ?
16500 00004D2C 741D              <1>       jz    short dapt_5 ; no
16501                            <1>                          ; yes
16502 00004D2E D1E8              <1>       shr   eax, 1
16503 00004D30 E8CA040000        <1>       call  unlink_swap_block ; Deallocate swapped page block
16504                            <1>                               ; on the swap disk (or in file)
16505 00004D35 EB14              <1>       jmp   short dapt_5
16506                            <1> dapt_2:
16507 00004D37 59                <1>       pop   ecx
16508 00004D38 5B                <1>       pop   ebx
16509                            <1> dapt_3:
16510                            <1>       ; 12/07/2016
16511 00004D39 66A90004          <1>       test  ax, PTE_SHARED ; shared or direct memory access indicator
16512 00004D3D 7505              <1>       jnz   short dapt_4   ; AVL bit 1 = 1, do not deallocate this page!
16513                            <1>       ;
16514                            <1>       ;and  ax, PTE_A_CLEAR ; 0F000h ; clear lower 12 (attribute) bits
16515 00004D3F E814000000        <1>       call  deallocate_page ; set the mem allocation bit of this page
16516                            <1> dapt_4:
16517 00004D44 C746FC00000000    <1>       mov   dword [esi-4], 0 ; clear/reset PTE (child, dupl. as parent)
16518                            <1> dapt_5:
16519 00004D4B 47                <1>       inc   edi ; page table entry index
16520 00004D4C 81FF00040000      <1>       cmp   edi, PAGE_SIZE / 4 ; 1024
16521 00004D52 7290              <1>       jb    short dapt_0
16522                            <1>       ;
16523 00004D54 58                <1>       pop   eax ; *
16524 00004D55 5A                <1>       pop   edx
16525 00004D56 5F                <1>       pop   edi
16526 00004D57 5E                <1>       pop   esi
16527                            <1>       ;
16528                            <1>       ;call  deallocate_page    ; deallocate the page table's itself
16529                            <1>       ;retn
16530                            <1>
16531                            <1> deallocate_page:
16532                            <1>       ; 15/09/2015
16533                            <1>       ; 28/04/2015
16534                            <1>       ; 10/03/2015
16535                            <1>       ; 17/10/2014
16536                            <1>       ; 12/10/2014 (Retro UNIX 386 v1 - beginning)
16537                            <1>       ;
16538                            <1>       ; INPUT ->
16539                            <1>       ;     EAX = PHYSICAL (real/flat) ADDRESS OF THE ALLOCATED PAGE
16540                            <1>       ; OUTPUT ->
16541                            <1>       ;     [free_pages] is increased
16542                            <1>       ;     (corresponding MEMORY ALLOCATION TABLE bit is SET)
16543                            <1>       ;     CF = 1 if the page is already deallocated
16544                            <1>       ;            (or not allocated) before.
16545                            <1>       ;
16546                            <1>       ; Modified Registers -> EAX
16547                            <1>       ;
16548 00004D58 53                <1>       push  ebx
16549 00004D59 52                <1>       push  edx
16550                            <1>       ;
16551 00004D5A C1E80C            <1>       shr   eax, PAGE_SHIFT      ; shift physical address to
16552                            <1>                                  ; 12 bits right
16553                            <1>                                  ; to get page number
16554 00004D5D 89C2              <1>       mov   edx, eax
16555                            <1>       ; 15/09/2015
16556 00004D5F C1EA03            <1>       shr   edx, 3               ; to get offset to M.A.T.
16557                            <1>                                  ; (1 allocation bit = 1 page)
16558                            <1>                                  ; (1 allocation bytes = 8 pages)
16559 00004D62 80E2FC            <1>       and   dl, 0FCh             ; clear lower 2 bits
16560                            <1>                                  ; (to get 32 bit position)
16561                            <1>       ;
16562 00004D65 BB00001000        <1>       mov   ebx, MEM_ALLOC_TBL   ; Memory Allocation Table address
16563 00004D6A 01D3              <1>       add   ebx, edx
16564 00004D6C 83E01F            <1>       and   eax, 1Fh             ; lower 5 bits only
16565                            <1>                                  ; (allocation bit position)
16566 00004D6F 3B15[2C520100]    <1>       cmp   edx, [next_page]     ; is the new free page address lower
16567                            <1>                                  ; than the address in 'next_page' ?
16568                            <1>                                  ; (next/first free page value)
16569 00004D75 7306              <1>       jnb   short dap_1          ; no
16570 00004D77 8915[2C520100]    <1>       mov   [next_page], edx     ; yes
16571                            <1> dap_1:
16572 00004D7D 0FAB03            <1>       bts   [ebx], eax           ; unlink/release/deallocate page
16573                            <1>                                  ; set relevant bit to 1.
16574                            <1>                                  ; set CF to the previous bit value
16575                            <1>       ;cmc                       ; complement carry flag
16576                            <1>       ;jc   short dap_2          ; do not increase free_pages count
16577                            <1>                                  ; if the page is already deallocated
16578                            <1>                                  ; before.
16579 00004D80 FF05[28520100]    <1>         inc     dword [free_pages]
16580                            <1> dap_2:
```

```
00004D86 5A                   <1>       pop    edx
00004D87 5B                   <1>       pop    ebx
00004D88 C3                   <1>       retn
                              <1> 
                              <1> ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                              <1> ;;                                                              ;;
                              <1> ;; Copyright (C) KolibriOS team 2004-2012. All rights reserved. ;;
                              <1> ;; Distributed under terms of the GNU General Public License    ;;
                              <1> ;;                                                              ;;
                              <1> ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                              <1> 
                              <1> ;;$Revision: 5057 $
                              <1> 
                              <1> 
                              <1> ;;align 4
                              <1> ;;proc alloc_page
                              <1> 
                              <1> ;;        pushfd
                              <1> ;;        cli
                              <1> ;;        push    ebx
                              <1> ;;;//-
                              <1> ;;        cmp     [pg_data.pages_free], 1
                              <1> ;;        jle     .out_of_memory
                              <1> ;;;//-
                              <1> ;;
                              <1> ;;        mov     ebx, [page_start]
                              <1> ;;        mov     ecx, [page_end]
                              <1> ;;.l1:
                              <1> ;;        bsf     eax, [ebx];
                              <1> ;;        jnz     .found
                              <1> ;;        add     ebx, 4
                              <1> ;;        cmp     ebx, ecx
                              <1> ;;        jb      .l1
                              <1> ;;        pop     ebx
                              <1> ;;        popfd
                              <1> ;;        xor     eax, eax
                              <1> ;;        ret
                              <1> ;;.found:
                              <1> ;;;//-
                              <1> ;;        dec     [pg_data.pages_free]
                              <1> ;;        jz      .out_of_memory
                              <1> ;;;//-
                              <1> ;;        btr     [ebx], eax
                              <1> ;;        mov     [page_start], ebx
                              <1> ;;        sub     ebx, sys_pgmap
                              <1> ;;        lea     eax, [eax+ebx*8]
                              <1> ;;        shl     eax, 12
                              <1> ;;;//-       dec [pg_data.pages_free]
                              <1> ;;        pop     ebx
                              <1> ;;        popfd
                              <1> ;;        ret
                              <1> ;;;//-
                              <1> ;;.out_of_memory:
                              <1> ;;        mov     [pg_data.pages_free], 1
                              <1> ;;        xor     eax, eax
                              <1> ;;        pop     ebx
                              <1> ;;        popfd
                              <1> ;;        ret
                              <1> ;;;//-
                              <1> ;;endp
                              <1> 
                              <1> duplicate_page_dir:
                              <1>       ; 21/09/2015
                              <1>       ; 31/08/2015
                              <1>       ; 20/07/2015
                              <1>       ; 28/04/2015
                              <1>       ; 27/04/2015
                              <1>       ; 18/04/2015
                              <1>       ; 12/04/2015
                              <1>       ; 18/10/2014
                              <1>       ; 16/10/2014 (Retro UNIX 386 v1 - beginning)
                              <1>       ;
                              <1>       ; INPUT ->
                              <1>       ;      [u.pgdir] = PHYSICAL (real/flat) ADDRESS of the parent's
                              <1>       ;                 page directory.
                              <1>       ; OUTPUT ->
                              <1>       ;      EAX =  PHYSICAL (real/flat) ADDRESS of the child's
                              <1>       ;             page directory.
                              <1>       ;      (New page directory with new page table entries.)
                              <1>       ;      (New page tables with read only copies of the parent's
                              <1>       ;      pages.)
                              <1>       ;      EAX = 0 -> Error (CF = 1)
                              <1>       ;
                              <1>       ; Modified Registers -> none (except EAX)
                              <1>       ;
00004D89 E8ECFDFFFF           <1>       call   allocate_page
00004D8E 723E                 <1>       jc     short dpd_err
                              <1>       ;
00004D90 55                   <1>       push   ebp ; 20/07/2015
00004D91 56                   <1>       push   esi
00004D92 57                   <1>       push   edi
00004D93 53                   <1>       push   ebx
00004D94 51                   <1>       push   ecx
00004D95 8B35[B8030300]       <1>       mov    esi, [u.pgdir]
00004D9B 89C7                 <1>       mov    edi, eax
00004D9D 50                   <1>       push   eax ; save child's page directory address
                              <1>       ; 31/08/2015
                              <1>       ; copy PDE 0 from the parent's page dir to the child's page dir
                              <1>       ; (use same system space for all user page tables)
00004D9E A5                   <1>       movsd
00004D9F BD00004000           <1>       mov    ebp, 1024*4096 ; pass the 1st 4MB (system space)
00004DA4 B9FF030000           <1>       mov    ecx, (PAGE_SIZE / 4) - 1 ; 1023
                              <1> dpd_0:
```

```
16684 00004DA9 AD                <1>       lodsd
16685                            <1>       ;or    eax, eax
16686                            <1>         ;jnz     short dpd_1
16687 00004DAA A801              <1>       test   al, PDE_A_PRESENT ;  bit 0 =  1
16688 00004DAC 7508              <1>       jnz    short dpd_1
16689                            <1>       ; 20/07/2015 (virtual address at the end of the page table)
16690 00004DAE 81C500004000      <1>       add    ebp, 1024*4096 ; page size * PTE count
16691 00004DB4 EB0F              <1>       jmp    short dpd_2
16692                            <1> dpd_1:
16693 00004DB6 662500F0          <1>       and    ax, PDE_A_CLEAR ; 0F000h ; clear attribute bits
16694 00004DBA 89C3              <1>       mov    ebx, eax
16695                            <1>       ; EBX = Parent's page table address
16696 00004DBC E81F000000        <1>       call   duplicate_page_table
16697 00004DC1 720C              <1>       jc     short dpd_p_err
16698                            <1>       ; EAX = Child's page table address
16699 00004DC3 0C07              <1>       or     al, PDE_A_PRESENT + PDE_A_WRITE + PDE_A_USER
16700                            <1>                      ; set bit 0, bit 1 and bit 2 to 1
16701                            <1>                      ; (present, writable, user)
16702                            <1> dpd_2:
16703 00004DC5 AB                <1>       stosd
16704 00004DC6 E2E1              <1>       loop   dpd_0
16705                            <1>       ;
16706 00004DC8 58                <1>       pop    eax  ; restore child's page directory address
16707                            <1> dpd_3:
16708 00004DC9 59                <1>       pop    ecx
16709 00004DCA 5B                <1>       pop    ebx
16710 00004DCB 5F                <1>       pop    edi
16711 00004DCC 5E                <1>       pop    esi
16712 00004DCD 5D                <1>       pop    ebp ; 20/07/2015
16713                            <1> dpd_err:
16714 00004DCE C3                <1>       retn
16715                            <1> dpd_p_err:
16716                            <1>       ; release the allocated pages missing (recover free space)
16717 00004DCF 58                <1>       pop    eax  ; the new page directory address (physical)
16718 00004DD0 8B1D[B8030300]    <1>       mov    ebx, [u.pgdir] ; parent's page directory address
16719 00004DD6 E8D8FEFFFF        <1>       call   deallocate_page_dir
16720 00004DDB 29C0              <1>       sub    eax, eax ; 0
16721 00004DDD F9                <1>       stc
16722 00004DDE EBE9              <1>       jmp    short dpd_3
16723                            <1>
16724                            <1> duplicate_page_table:
16725                            <1>       ; 20/02/2017
16726                            <1>       ; 21/09/2015
16727                            <1>       ; 20/07/2015
16728                            <1>       ; 05/05/2015
16729                            <1>       ; 28/04/2015
16730                            <1>       ; 27/04/2015
16731                            <1>       ; 18/04/2015
16732                            <1>       ; 18/10/2014
16733                            <1>       ; 16/10/2014 (Retro UNIX 386 v1 - beginning)
16734                            <1>       ;
16735                            <1>       ; INPUT ->
16736                            <1>       ;     EBX = PHYSICAL (real/flat) ADDRESS of the parent's page table.
16737                            <1>       ;       20/02/2017
16738                            <1>       ;     EBP = Linear address of the page (from 'duplicate_page_dir')
16739                            <1>       ;           (Linear address = CORE + user's virtual address)
16740                            <1>       ; OUTPUT ->
16741                            <1>       ;     EAX = PHYSICAL (real/flat) ADDRESS of the child's page table.
16742                            <1>       ;           (with 'read only' attribute of page table entries)
16743                            <1>       ;     20/02/2017
16744                            <1>       ;     EBP = Next linear page address (for 'duplicate_page_dir')
16745                            <1>       ;
16746                            <1>       ;     CF = 1 -> error
16747                            <1>       ;
16748                            <1>       ; Modified Registers -> EBP (except EAX)
16749                            <1>       ;
16750 00004DE0 E895FDFFFF        <1>       call   allocate_page
16751 00004DE5 726A              <1>       jc     short dpt_err
16752                            <1>       ;
16753 00004DE7 50                <1>       push   eax ; *
16754 00004DE8 56                <1>       push   esi
16755 00004DE9 57                <1>       push   edi
16756 00004DEA 52                <1>       push   edx
16757 00004DEB 51                <1>       push   ecx
16758                            <1>       ;
16759 00004DEC 89DE              <1>       mov    esi, ebx
16760 00004DEE 89C7              <1>       mov    edi, eax
16761 00004DF0 89C2              <1>       mov    edx, eax
16762 00004DF2 81C200100000      <1>       add    edx, PAGE_SIZE
16763                            <1> dpt_0:
16764 00004DF8 AD                <1>       lodsd
16765 00004DF9 21C0              <1>       and    eax, eax
16766 00004DFB 7444              <1>       jz     short dpt_3
16767 00004DFD A801              <1>       test   al, PTE_A_PRESENT ;  bit 0 =  1
16768 00004DFF 7507              <1>       jnz    short dpt_1
16769                            <1>       ; 20/07/2015
16770                            <1>       ; ebp = virtual (linear) address of the memory page
16771 00004E01 E83F050000        <1>       call   reload_page ; 28/04/2015
16772 00004E06 7244              <1>       jc     short dpt_p_err
16773                            <1> dpt_1:
16774                            <1>       ; 21/09/2015
16775 00004E08 89C1              <1>       mov    ecx, eax
16776 00004E0A 662500F0          <1>       and    ax, PTE_A_CLEAR ; 0F000h ; clear attribute bits
16777 00004E0E F6C102            <1>       test   cl, PTE_A_WRITE ; writable page ?
16778 00004E11 7525              <1>       jnz    short dpt_2
16779                            <1>       ; Read only (parent) page
16780                            <1>       ;      - there is a third process which uses this page -
16781                            <1>       ; Allocate a new page for the child process
16782 00004E13 E862FDFFFF        <1>       call   allocate_page
16783 00004E18 7232              <1>       jc     short dpt_p_err
16784 00004E1A 57                <1>       push   edi
16785 00004E1B 56                <1>       push   esi
16786 00004E1C 89CE              <1>       mov    esi, ecx
```

```
16787 00004E1E 89C7              <1>       mov    edi, eax
16788 00004E20 B900040000        <1>       mov    ecx, PAGE_SIZE/4
16789 00004E25 F3A5              <1>       rep    movsd  ; copy page (4096 bytes)
16790 00004E27 5E                <1>       pop    esi
16791 00004E28 5F                <1>       pop    edi
16792                            <1>       ;
16793 00004E29 53                <1>       push   ebx
16794 00004E2A 50                <1>       push   eax
16795                            <1>       ; 20/07/2015
16796 00004E2B 89EB              <1>       mov    ebx, ebp
16797                            <1>       ; ebx = virtual (linear) address of the memory page
16798 00004E2D E887030000        <1>       call   add_to_swap_queue
16799 00004E32 58                <1>       pop    eax
16800 00004E33 5B                <1>       pop    ebx
16801                            <1>       ; 21/09/2015
16802 00004E34 0C07              <1>       or     al, PTE_A_USER+PTE_A_WRITE+PTE_A_PRESENT
16803                            <1>              ; user + writable + present page
16804 00004E36 EB09              <1>       jmp    short dpt_3
16805                            <1> dpt_2:
16806                            <1>       ;or    ax, PTE_A_USER+PTE_A_PRESENT
16807 00004E38 0C05              <1>       or     al, PTE_A_USER+PTE_A_PRESENT
16808                            <1>                  ; (read only page!)
16809 00004E3A 8946FC            <1>       mov    [esi-4], eax ; update parent's PTE
16810 00004E3D 660D0002          <1>       or      ax, PTE_DUPLICATED  ; (read only page & duplicated PTE!)
16811                            <1> dpt_3:
16812 00004E41 AB                <1>       stosd  ; EDI points to child's PTE
16813                            <1>       ;
16814 00004E42 81C500100000      <1>       add    ebp, 4096 ; 20/07/2015 (next page)
16815                            <1>       ;
16816 00004E48 39D7              <1>       cmp    edi, edx
16817 00004E4A 72AC              <1>       jb     short dpt_0
16818                            <1> dpt_p_err:
16819 00004E4C 59                <1>       pop    ecx
16820 00004E4D 5A                <1>       pop    edx
16821 00004E4E 5F                <1>       pop    edi
16822 00004E4F 5E                <1>       pop    esi
16823 00004E50 58                <1>       pop    eax ; *
16824                            <1> dpt_err:
16825 00004E51 C3                <1>       retn
16826                            <1>
16827                            <1> page_fault_handler:      ; CPU EXCEPTION 0Eh (14) : Page Fault !
16828                            <1>       ; 21/09/2015
16829                            <1>       ; 19/09/2015
16830                            <1>       ; 17/09/2015
16831                            <1>       ; 28/08/2015
16832                            <1>       ; 20/07/2015
16833                            <1>       ; 28/06/2015
16834                            <1>       ; 03/05/2015
16835                            <1>       ; 30/04/2015
16836                            <1>       ; 18/04/2015
16837                            <1>       ; 12/04/2015
16838                            <1>       ; 30/10/2014
16839                            <1>       ; 11/09/2014
16840                            <1>       ; 10/09/2014 (Retro UNIX 386 v1 - beginning)
16841                            <1>       ;
16842                            <1>       ; Note: This is not an interrupt/exception handler.
16843                            <1>       ;     This is a 'page fault remedy' subroutine
16844                            <1>       ;     which will be called by standard/uniform
16845                            <1>       ;     exception handler.
16846                            <1>       ;
16847                            <1>       ; INPUT ->
16848                            <1>       ;     [error_code] = 32 bit ERROR CODE (lower 5 bits are valid)
16849                            <1>       ;
16850                            <1>       ;     cr2 = the virtual (linear) address
16851                            <1>       ;           which has caused to page fault (19/09/2015)
16852                            <1>       ;
16853                            <1>       ; OUTPUT ->
16854                            <1>       ;     (corresponding PAGE TABLE ENTRY is mapped/set)
16855                            <1>       ;     EAX = 0 -> no error
16856                            <1>       ;     EAX > 0 -> error code in EAX (also CF = 1)
16857                            <1>       ;
16858                            <1>       ; Modified Registers -> none (except EAX)
16859                            <1>       ;
16860                            <1>         ;
16861                            <1>         ; ERROR CODE:
16862                            <1>       ;       31  .....   4   3  2   1  0
16863                            <1>       ;      +---+-- --+---+---+---+---+---+
16864                            <1>       ;      |   Reserved  | I | R | U | W | P |
16865                            <1>       ;      +---+-- --+---+---+---+---+---+
16866                            <1>       ;
16867                            <1>       ; P : PRESENT -     When set, the page fault was caused by
16868                            <1>       ;             a page-protection violation. When not set,
16869                            <1>       ;             it was caused by a non-present page.
16870                            <1>       ; W : WRITE   -     When set, the page fault was caused by
16871                            <1>       ;             a page write. When not set, it was caused
16872                            <1>       ;             by a page read.
16873                            <1>       ; U : USER    -     When set, the page fault was caused
16874                            <1>       ;             while CPL = 3.
16875                            <1>       ;             This does not necessarily mean that
16876                            <1>       ;             the page fault was a privilege violation.
16877                            <1>       ; R : RESERVD -     When set, the page fault was caused by
16878                            <1>       ;     WRITE  reading a 1 in a reserved field.
16879                            <1>       ; I : INSTRUC -     When set, the page fault was caused by
16880                            <1>       ;     FETCH  an instruction fetch
16881                            <1>       ;
16882                            <1>       ;; x86 (32 bit) VIRTUAL ADDRESS TRANSLATION
16883                            <1>       ;  31               22                  12 11                     0
16884                            <1>       ; +------------------+------------------+-----------------------+
16885                            <1>             ; | PAGE DIR. ENTRY # | PAGE TAB. ENTRY # |        OFFSET         |
16886                            <1>             ; +------------------+------------------+-----------------------+
16887                            <1>       ;
16888                            <1>
16889                            <1>       ;; CR3 REGISTER (Control Register 3)
```

```
16890                                  <1>       ;  31                                   12            5 4 3 2   0
16891                                  <1>       ; +---------------------------------------+------------+---+-----+
16892                                  <1>            ; |                                       |         |P|P|     |
16893                                  <1>            ; |   PAGE DIRECTORY TABLE BASE ADDRESS   |  reserved     |C|W|rsvrd|
16894                                  <1>            ; |                                       |         |D|T|     |
16895                                  <1>       ; +---------------------------------------+------------+---+-----+
16896                                  <1>       ;
16897                                  <1>       ;     PWT    - WRITE THROUGH
16898                                  <1>       ;     PCD    - CACHE DISABLE
16899                                  <1>       ;
16900                                  <1>       ;
16901                                  <1>       ;; x86 PAGE DIRECTORY ENTRY (4 KByte Page)
16902                                  <1>       ;  31                                   12 11  9 8 7 6 5 4 3 2 1 0
16903                                  <1>       ; +---------------------------------------+-----+---+-+-+---+-+-+-+
16904                                  <1>            ; |                                       |     | | | | |P|P|U|R| |
16905                                  <1>            ; |     PAGE TABLE BASE ADDRESS 31..12    | AVL |G|0|D|A|C|W|/|/|P|
16906                                  <1>            ; |                                       |     | | | | |D|T|S|W| |
16907                                  <1>       ; +---------------------------------------+-----+---+-+-+---+-+-+-+
16908                                  <1>       ;
16909                                  <1>         ;       P      - PRESENT
16910                                  <1>         ;       R/W    - READ/WRITE
16911                                  <1>         ;       U/S    - USER/SUPERVISOR
16912                                  <1>       ;     PWT    - WRITE THROUGH
16913                                  <1>       ;     PCD    - CACHE DISABLE
16914                                  <1>       ;     A      - ACCESSED
16915                                  <1>         ;       D      - DIRTY (IGNORED)
16916                                  <1>       ;     PAT    - PAGE ATTRIBUTE TABLE INDEX (CACHE BEHAVIOR)
16917                                  <1>       ;     G      - GLOBAL     (IGNORED)
16918                                  <1>         ;       AVL    - AVAILABLE FOR SYSTEMS PROGRAMMER USE
16919                                  <1>       ;
16920                                  <1>       ;
16921                                  <1>       ;; x86 PAGE TABLE ENTRY (4 KByte Page)
16922                                  <1>       ;  31                                   12 11  9 8 7 6 5 4 3 2 1 0
16923                                  <1>       ; +---------------------------------------+-----+---+-+-+---+-+-+-+
16924                                  <1>            ; |                                       |     | |P| | |P|P|U|R| |
16925                                  <1>            ; |     PAGE FRAME BASE ADDRESS 31..12    | AVL |G|A|D|A|C|W|/|/|P|
16926                                  <1>            ; |                                       |     | |T| | |D|T|S|W| |
16927                                  <1>       ; +---------------------------------------+-----+---+-+-+---+-+-+-+
16928                                  <1>       ;
16929                                  <1>         ;       P      - PRESENT
16930                                  <1>         ;       R/W    - READ/WRITE
16931                                  <1>         ;       U/S    - USER/SUPERVISOR
16932                                  <1>       ;     PWT    - WRITE THROUGH
16933                                  <1>       ;     PCD    - CACHE DISABLE
16934                                  <1>       ;     A      - ACCESSED
16935                                  <1>         ;       D      - DIRTY
16936                                  <1>       ;     PAT    - PAGE ATTRIBUTE TABLE INDEX (CACHE BEHAVIOR)
16937                                  <1>       ;     G      - GLOBAL
16938                                  <1>         ;       AVL    - AVAILABLE FOR SYSTEMS PROGRAMMER USE
16939                                  <1>       ;
16940                                  <1>       ;
16941                                  <1>       ;; 80386 PAGE TABLE ENTRY (4 KByte Page)
16942                                  <1>       ;  31                                   12 11  9 8 7 6 5 4 3 2 1 0
16943                                  <1>       ; +---------------------------------------+-----+-+-+-+-+---+-+-+-+
16944                                  <1>            ; |                                       |     | | | | | | |U|R| |
16945                                  <1>            ; |     PAGE FRAME BASE ADDRESS 31..12    | AVL |0|0|D|A|0|0|/|/|P|
16946                                  <1>            ; |                                       |     | | | | | | |S|W| |
16947                                  <1>            ; +---------------------------------------+-----+-+-+-+-+---+-+-+-+
16948                                  <1>       ;
16949                                  <1>         ;       P      - PRESENT
16950                                  <1>         ;       R/W    - READ/WRITE
16951                                  <1>         ;       U/S    - USER/SUPERVISOR
16952                                  <1>         ;       D      - DIRTY
16953                                  <1>         ;       AVL    - AVAILABLE FOR SYSTEMS PROGRAMMER USE
16954                                  <1>       ;
16955                                  <1>         ;       NOTE: 0 INDICATES INTEL RESERVED. DO NOT DEFINE.
16956                                  <1>       ;
16957                                  <1>       ;
16958                                  <1>       ;; Invalid Page Table Entry
16959                                  <1>       ; 31                                                      1 0
16960                                  <1>            ; +-------------------------------------------------------------+-+
16961                                  <1>            ; |                                                             | |
16962                                  <1>            ; |                          AVAILABLE                          |0|
16963                                  <1>            ; |                                                             | |
16964                                  <1>            ; +-------------------------------------------------------------+-+
16965                                  <1>       ;
16966                                  <1>
16967 00004E52 53                      <1>       push   ebx
16968 00004E53 52                      <1>       push   edx
16969 00004E54 51                      <1>       push   ecx
16970                                  <1>       ;
16971                                  <1>       ; 21/09/2015 (debugging)
16972 00004E55 FF05[CC030300]          <1>       inc    dword [u.pfcount] ; page fault count for running process
16973 00004E5B FF05[80050300]          <1>       inc    dword [PF_Count] ; total page fault count
16974                                  <1>       ; 28/06/2015
16975                                  <1>       ;mov   edx, [error_code] ; Lower 5 bits are valid
16976 00004E61 8A15[78050300]          <1>       mov    dl, [error_code]
16977                                  <1>       ;
16978 00004E67 F6C201                  <1>       test   dl, 1  ; page fault was caused by a non-present page
16979                                  <1>                     ; sign
16980 00004E6A 7422                    <1>       jz     short pfh_alloc_np
16981                                  <1>       ;
16982                                  <1>       ; If it is not a 'write on read only page' type page fault
16983                                  <1>       ; major page fault error with minor reason must be returned without
16984                                  <1>       ; fixing the problem. 'sys_exit with error' will be needed
16985                                  <1>       ; after return here!
16986                                  <1>       ; Page fault will be remedied, by copying page contents
16987                                  <1>       ; to newly allocated page with write permission;
16988                                  <1>       ; sys_fork -> sys_exec -> copy on write, demand paging method is
16989                                  <1>       ; used for working with minimum possible memory usage.
16990                                  <1>       ; sys_fork will duplicate page directory and tables of parent
16991                                  <1>       ; process with 'read only' flag. If the child process attempts to
16992                                  <1>       ; write on these read only pages, page fault will be directed here
```

```
16993                                  <1>       ; for allocating a new page with same data/content.
16994                                  <1>       ;
16995                                  <1>       ; IMPORTANT : Retro UNIX 386 v1 (and SINGLIX and TR-DOS)
16996                                  <1>       ; will not force to separate CODE and DATA space
16997                                  <1>       ; in a process/program...
16998                                  <1>       ; CODE segment/section may contain DATA!
16999                                  <1>       ; It is flat, smoth and simplest programming method already as in
17000                                  <1>       ; Retro UNIX 8086 v1 and MS-DOS programs.
17001                                  <1>       ;
17002 00004E6C F6C202                  <1>       test   dl, 2  ; page fault was caused by a page write
17003                                  <1>                     ; sign
17004 00004E6F 0F84AB000000            <1>         jz      pfh_p_err
17005                                  <1>       ; 31/08/2015
17006 00004E75 F6C204                  <1>       test   dl, 4  ; page fault was caused while CPL = 3 (user mode)
17007                                  <1>                     ; sign.  (U+W+P = 4+2+1 = 7)
17008 00004E78 0F84A2000000            <1>         jz   pfh_pv_err
17009                                  <1>       ;
17010                                  <1>       ; make a new page and copy the parent's page content
17011                                  <1>       ; as the child's new page content
17012                                  <1>       ;
17013 00004E7E 0F20D3                  <1>       mov    ebx, cr2 ; CR2 contains the linear address
17014                                  <1>                       ; which has caused to page fault
17015 00004E81 E8A2000000              <1>       call   copy_page
17016 00004E86 0F828D000000            <1>         jc      pfh_im_err ; insufficient memory
17017                                  <1>       ;
17018 00004E8C EB7D                    <1>         jmp     pfh_cpp_ok
17019                                  <1>       ;
17020                                  <1> pfh_alloc_np:
17021 00004E8E E8E7FCFFFF              <1>       call   allocate_page; (allocate a new page)
17022 00004E93 0F8280000000            <1>         jc      pfh_im_err      ; 'insufficient memory' error
17023                                  <1> pfh_chk_cpl:
17024                                  <1>       ; EAX = Physical (base) address of the allocated (new) page
17025                                  <1>             ; (Lower 12 bits are ZERO, because
17026                                  <1>             ;      the address is on a page boundary)
17027 00004E99 80E204                  <1>       and    dl, 4  ; CPL = 3 ?
17028 00004E9C 7505                    <1>       jnz    short pfh_um
17029                                  <1>                     ; Page fault handler for kernel/system mode (CPL=0)
17030 00004E9E 0F20DB                  <1>       mov    ebx, cr3 ; CR3 (Control Register 3) contains physical address
17031                                  <1>                       ; of the current/active page directory
17032                                  <1>                       ; (Always kernel/system mode page directory, here!)
17033                                  <1>                       ; Note: Lower 12 bits are 0. (page boundary)
17034 00004EA1 EB06                    <1>       jmp    short pfh_get_pde
17035                                  <1>       ;
17036                                  <1> pfh_um:                    ; Page fault handler for user/appl. mode (CPL=3)
17037 00004EA3 8B1D[B8030300]          <1>       mov    ebx, [u.pgdir] ; Page directory of current/active process
17038                                  <1>                     ; Physical address of the USER's page directory
17039                                  <1>                     ; Note: Lower 12 bits are 0. (page boundary)
17040                                  <1> pfh_get_pde:
17041 00004EA9 80CA03                  <1>       or     dl, 3  ; USER + WRITE + PRESENT or SYSTEM + WRITE + PRESENT
17042 00004EAC 0F20D1                  <1>       mov    ecx, cr2 ; CR2 contains the virtual address
17043                                  <1>                       ; which has been caused to page fault
17044                                  <1>                       ;
17045 00004EAF C1E914                  <1>       shr    ecx, 20       ; shift 20 bits right
17046 00004EB2 80E1FC                  <1>       and    cl, 0FCh ; mask lower 2 bits to get PDE offset
17047                                  <1>       ;
17048 00004EB5 01CB                    <1>       add    ebx, ecx ; now, EBX points to the relevant page dir entry
17049 00004EB7 8B0B                    <1>       mov    ecx, [ebx] ; physical (base) address of the page table
17050 00004EB9 F6C101                  <1>       test   cl, 1   ; check bit 0 is set (1) or not (0).
17051 00004EBC 740B                    <1>       jz     short pfh_set_pde ; Page directory entry is not valid,
17052                                  <1>                              ; set/validate page directory entry
17053 00004EBE 6681E100F0              <1>       and    cx, PDE_A_CLEAR ; 0F000h ; Clear attribute bits
17054 00004EC3 89CB                    <1>       mov    ebx, ecx ; Physical address of the page table
17055 00004EC5 89C1                    <1>       mov    ecx, eax ; new page address (physical)
17056 00004EC7 EB16                    <1>       jmp    short pfh_get_pte
17057                                  <1> pfh_set_pde:
17058                                  <1>       ;; NOTE: Page directories and page tables never be swapped out!
17059                                  <1>       ;;     (So, we know this PDE is empty or invalid)
17060                                  <1>       ;
17061 00004EC9 08D0                    <1>       or     al, dl  ; lower 3 bits are used as U/S, R/W, P flags
17062 00004ECB 8903                    <1>       mov    [ebx], eax ; Let's put the new page directory entry here !
17063 00004ECD 30C0                    <1>       xor    al, al  ; clear lower (3..8) bits
17064 00004ECF 89C3                    <1>       mov    ebx, eax
17065 00004ED1 E8A4FCFFFF              <1>       call   allocate_page ; (allocate a new page)
17066 00004ED6 7241                    <1>       jc     short pfh_im_err   ; 'insufficient memory' error
17067                                  <1> pfh_spde_1:
17068                                  <1>       ; EAX = Physical (base) address of the allocated (new) page
17069 00004ED8 89C1                    <1>       mov    ecx, eax
17070 00004EDA E815FDFFFF              <1>       call   clear_page ; Clear page content
17071                                  <1> pfh_get_pte:
17072 00004EDF 0F20D0                  <1>       mov    eax, cr2 ; virtual address
17073                                  <1>                       ; which has been caused to page fault
17074 00004EE2 89C7                    <1>       mov    edi, eax ; 20/07/2015
17075 00004EE4 C1E80C                  <1>       shr    eax, 12       ; shift 12 bit right to get
17076                                  <1>                       ; higher 20 bits of the page fault address
17077 00004EE7 25FF030000              <1>       and    eax, 3FFh ; mask PDE# bits, the result is PTE# (0 to 1023)
17078 00004EEC C1E002                  <1>       shl    eax, 2 ; shift 2 bits left to get PTE offset
17079 00004EEF 01C3                    <1>       add    ebx, eax ; now, EBX points to the relevant page table entry
17080 00004EF1 8B03                    <1>       mov    eax, [ebx] ; get previous value of pte
17081                                  <1>             ; bit 0 of EAX is always 0 (otherwise we would not be here)
17082 00004EF3 21C0                    <1>       and    eax, eax
17083 00004EF5 7410                    <1>       jz     short pfh_gpte_1
17084                                  <1>       ; 20/07/2015
17085 00004EF7 87D9                    <1>       xchg   ebx, ecx ; new page address (physical)
17086 00004EF9 55                      <1>       push   ebp ; 20/07/2015
17087 00004EFA 0F20D5                  <1>       mov    ebp, cr2
17088                                  <1>             ; ECX = physical address of the page table entry
17089                                  <1>             ; EBX = Memory page address (physical!)
17090                                  <1>             ; EAX = Swap disk (offset) address
17091                                  <1>             ; EBP = virtual address (page fault address)
17092 00004EFD E8B7000000              <1>       call   swap_in
17093 00004F02 5D                      <1>       pop    ebp
17094 00004F03 7210                    <1>       jc      short pfh_err_retn
17095 00004F05 87CB                    <1>       xchg   ecx, ebx
```

```
17096                                  <1>             ; EBX = physical address of the page table entry
17097                                  <1>             ; ECX = new page
17098                                  <1> pfh_gpte_1:
17099 00004F07 08D1                    <1>             or      cl, dl ; lower 3 bits are used as U/S, R/W, P flags
17100 00004F09 890B                    <1>             mov     [ebx], ecx ; Let's put the new page table entry here !
17101                                  <1> pfh_cpp_ok:
17102                                  <1>             ; 20/07/2015
17103 00004F0B 0F20D3                  <1>             mov     ebx, cr2
17104 00004F0E E8A6020000              <1>             call    add_to_swap_queue
17105                                  <1>             ;
17106                                  <1>             ; The new PTE (which contains the new page) will be added to
17107                                  <1>             ; the swap queue, here.
17108                                  <1>             ; (Later, if memory will become insufficient,
17109                                  <1>             ; one page will be swapped out which is at the head of
17110                                  <1>             ; the swap queue by using FIFO and access check methods.)
17111                                  <1>             ;
17112 00004F13 31C0                    <1>             xor     eax, eax  ; 0
17113                                  <1>             ;
17114                                  <1> pfh_err_retn:
17115 00004F15 59                      <1>             pop     ecx
17116 00004F16 5A                      <1>             pop     edx
17117 00004F17 5B                      <1>             pop     ebx
17118 00004F18 C3                      <1>             retn
17119                                  <1>
17120                                  <1> pfh_im_err:
17121 00004F19 B8E4000000              <1>             mov     eax, ERR_MAJOR_PF + ERR_MINOR_IM ; Error code in AX
17122                                  <1>                             ; Major (Primary) Error: Page Fault
17123                                  <1>                             ; Minor (Secondary) Error: Insufficient Memory !
17124 00004F1E EBF5                    <1>             jmp     short pfh_err_retn
17125                                  <1>
17126                                  <1>
17127                                  <1> pfh_p_err: ; 09/03/2015
17128                                  <1> pfh_pv_err:
17129                                  <1>             ; Page fault was caused by a protection-violation
17130 00004F20 B8E6000000              <1>             mov     eax, ERR_MAJOR_PF + ERR_MINOR_PV ; Error code in AX
17131                                  <1>                             ; Major (Primary) Error: Page Fault
17132                                  <1>                             ; Minor (Secondary) Error: Protection violation !
17133 00004F25 F9                      <1>             stc
17134 00004F26 EBED                    <1>             jmp     short pfh_err_retn
17135                                  <1>
17136                                  <1> copy_page:
17137                                  <1>             ; 22/09/2015
17138                                  <1>             ; 21/09/2015
17139                                  <1>             ; 19/09/2015
17140                                  <1>             ; 07/09/2015
17141                                  <1>             ; 31/08/2015
17142                                  <1>             ; 20/07/2015
17143                                  <1>             ; 05/05/2015
17144                                  <1>             ; 03/05/2015
17145                                  <1>             ; 18/04/2015
17146                                  <1>             ; 12/04/2015
17147                                  <1>             ; 30/10/2014
17148                                  <1>             ; 18/10/2014 (Retro UNIX 386 v1 - beginning)
17149                                  <1>             ;
17150                                  <1>             ; INPUT ->
17151                                  <1>             ;       EBX = Virtual (linear) address of source page
17152                                  <1>             ;             (Page fault address)
17153                                  <1>             ; OUTPUT ->
17154                                  <1>             ;       EAX = PHYSICAL (real/flat) ADDRESS OF THE ALLOCATED PAGE
17155                                  <1>             ;       (corresponding PAGE TABLE ENTRY is mapped/set)
17156                                  <1>             ;       EAX = 0 (CF = 1)
17157                                  <1>             ;               if there is not a free page to be allocated
17158                                  <1>             ;       (page content of the source page will be copied
17159                                  <1>             ;       onto the target/new page)
17160                                  <1>             ;
17161                                  <1>             ; Modified Registers -> ecx, ebx (except EAX)
17162                                  <1>             ;
17163 00004F28 56                      <1>             push    esi
17164 00004F29 57                      <1>             push    edi
17165                                  <1>             ;push   ebx
17166                                  <1>             ;push   ecx
17167 00004F2A 31F6                    <1>             xor     esi, esi
17168 00004F2C C1EB0C                  <1>             shr     ebx, 12 ; shift 12 bits right to get PDE & PTE numbers
17169 00004F2F 89D9                    <1>             mov     ecx, ebx ; save page fault address (as 12 bit shifted)
17170 00004F31 C1EB08                  <1>             shr     ebx, 8  ; shift 8 bits right and then
17171 00004F34 80E3FC                  <1>             and     bl, 0FCh ; mask lower 2 bits to get PDE offset
17172 00004F37 89DF                    <1>             mov     edi, ebx ; save it for the parent of current process
17173 00004F39 031D[B8030300]          <1>             add     ebx, [u.pgdir] ; EBX points to the relevant page dir entry
17174 00004F3F 8B03                    <1>             mov     eax, [ebx] ; physical (base) address of the page table
17175 00004F41 662500F0                <1>             and     ax, PTE_A_CLEAR ; 0F000h ; clear attribute bits
17176 00004F45 89CB                    <1>             mov     ebx, ecx   ; (restore higher 20 bits of page fault address)
17177 00004F47 81E3FF030000            <1>             and     ebx, 3FFh  ; mask PDE# bits, the result is PTE# (0 to 1023)
17178 00004F4D 66C1E302                <1>             shl     bx, 2     ; shift 2 bits left to get PTE offset
17179 00004F51 01C3                    <1>             add     ebx, eax   ; EBX points to the relevant page table entry
17180                                  <1>             ; 07/09/2015
17181 00004F53 66F7030002              <1>               test    word [ebx], PTE_DUPLICATED ; (Does current process share this
17182                                  <1>                                        ; read only page as a child process?)
17183 00004F58 7509                    <1>             jnz     short cpp_0 ; yes
17184 00004F5A 8B0B                    <1>             mov     ecx, [ebx] ; PTE value
17185 00004F5C 6681E100F0              <1>             and     cx, PTE_A_CLEAR ; 0F000h  ; clear page attributes
17186 00004F61 EB32                    <1>             jmp     short cpp_1
17187                                  <1> cpp_0:
17188 00004F63 89FE                    <1>             mov     esi, edi
17189 00004F65 0335[BC030300]          <1>             add     esi, [u.ppgdir] ; the parent's page directory entry
17190 00004F6B 8B06                    <1>             mov     eax, [esi] ; physical (base) address of the page table
17191 00004F6D 662500F0                <1>             and     ax, PTE_A_CLEAR ; 0F000h ; clear attribute bits
17192 00004F71 89CE                    <1>             mov     esi, ecx   ; (restore higher 20 bits of page fault address)
17193 00004F73 81E6FF030000            <1>             and     esi, 3FFh  ; mask PDE# bits, the result is PTE# (0 to 1023)
17194 00004F79 66C1E602                <1>             shl     si, 2     ; shift 2 bits left to get PTE offset
17195 00004F7D 01C6                    <1>             add     esi, eax   ; EDX points to the relevant page table entry
17196 00004F7F 8B0E                    <1>             mov     ecx, [esi] ; PTE value of the parent process
17197                                  <1>             ; 21/09/2015
17198 00004F81 8B03                    <1>             mov     eax, [ebx] ; PTE value of the child process
```

```
17199 00004F83 662500F0            <1>       and   ax, PTE_A_CLEAR ; 0F000h ; clear page attributes
17200                              <1>       ;
17201 00004F87 F6C101              <1>       test  cl, PTE_A_PRESENT ; is it a present/valid page ?
17202 00004F8A 7424                <1>       jz    short cpp_3 ; the parent's page is not same page
17203                              <1>       ;
17204 00004F8C 6681E100F0          <1>       and   cx, PTE_A_CLEAR ; 0F000h ; clear page attributes
17205 00004F91 39C8                <1>       cmp   eax, ecx   ; Same page?
17206 00004F93 751B                <1>       jne   short cpp_3 ; Parent page and child page are not same
17207                              <1>                         ; Convert child's page to writable page
17208                              <1> cpp_1:
17209 00004F95 E8E0FBFFFF          <1>       call  allocate_page
17210 00004F9A 721A                <1>       jc    short cpp_4 ; 'insufficient memory' error
17211 00004F9C 21F6                <1>       and   esi, esi    ; check ESI is valid or not
17212 00004F9E 7405                <1>       jz    short cpp_2
17213                              <1>             ; Convert read only page to writable page
17214                              <1>             ;(for the parent of the current process)
17215                              <1>       ;and  word [esi], PTE_A_CLEAR ; 0F000h
17216                              <1>       ; 22/09/2015
17217 00004FA0 890E                <1>       mov   [esi], ecx
17218 00004FA2 800E07              <1>       or    byte [esi], PTE_A_PRESENT + PTE_A_WRITE + PTE_A_USER
17219                              <1>                           ; 1+2+4 = 7
17220                              <1> cpp_2:
17221 00004FA5 89C7                <1>       mov   edi, eax ; new page address of the child process
17222                              <1>       ; 07/09/2015
17223 00004FA7 89CE                <1>       mov   esi, ecx ; the page address of the parent process
17224 00004FA9 B900040000          <1>       mov   ecx, PAGE_SIZE / 4
17225 00004FAE F3A5                <1>       rep   movsd ; 31/08/2015
17226                              <1> cpp_3:
17227 00004FB0 0C07                <1>       or    al, PTE_A_PRESENT + PTE_A_WRITE + PTE_A_USER ; 1+2+4 = 7
17228 00004FB2 8903                <1>       mov   [ebx], eax ; Update PTE
17229 00004FB4 28C0                <1>       sub   al, al ; clear attributes
17230                              <1> cpp_4:
17231                              <1>       ;pop  ecx
17232                              <1>       ;pop  ebx
17233 00004FB6 5F                  <1>       pop   edi
17234 00004FB7 5E                  <1>       pop   esi
17235 00004FB8 C3                  <1>       retn
17236                              <1>
17237                              <1> ;; 28/04/2015
17238                              <1> ;; 24/10/2014
17239                              <1> ;; 21/10/2014 (Retro UNIX 386 v1 - beginning)
17240                              <1> ;; SWAP_PAGE_QUEUE (4096 bytes)
17241                              <1> ;;
17242                              <1> ;;   0000   0001   0002   0003   ....   1020   1021   1022   1023
17243                              <1> ;; +------+------+------+------+-    -+------+------+------+------+
17244                              <1> ;; |  pg1 |  pg2 |  pg3 |  pg4 | .... |pg1021|pg1022|pg1023|pg1024|
17245                              <1> ;; +------+------+------+------+-    -+------+------+------+------+
17246                              <1> ;;
17247                              <1> ;; [swpq_last] = 0 to 4096 (step 4) -> the last position on the queue
17248                              <1> ;;
17249                              <1> ;; Method:
17250                              <1> ;;    Swap page queue is a list of allocated pages with physical
17251                              <1> ;;    addresses (system mode virtual adresses = physical addresses).
17252                              <1> ;;    It is used for 'swap_in' and 'swap_out' procedures.
17253                              <1> ;;    When a new page is being allocated, swap queue is updated
17254                              <1> ;;    by 'swap_queue_shift' procedure, header of the queue (offset 0)
17255                              <1> ;;    is checked for 'accessed' flag. If the 1st page on the queue
17256                              <1> ;;    is 'accessed' or 'read only', it is dropped from the list;
17257                              <1> ;;    other pages from the 2nd to the last (in [swpq_last]) shifted
17258                              <1> ;;    to head then the 2nd page becomes the 1st and '[swpq_last]'
17259                              <1> ;;    offset value becomes it's previous offset value - 4.
17260                              <1> ;;    If the 1st page of the swap page queue is not 'accessed'
17261                              <1> ;;    the queue/list is not shifted.
17262                              <1> ;;    After the queue/list shift, newly allocated page is added
17263                              <1> ;;    to the tail of the queue at the [swpq_count*4] position.
17264                              <1> ;;    But, if [swpq_count] > 1023, the newly allocated page
17265                              <1> ;;    will not be added to the tail of swap page queue.
17266                              <1> ;;
17267                              <1> ;;    During 'swap_out' procedure, swap page queue is checked for
17268                              <1> ;;    the first non-accessed, writable page in the list,
17269                              <1> ;;    from the head to the tail. The list is shifted to left
17270                              <1> ;;    (to the head) till a non-accessed page will be found in the list.
17271                              <1> ;;    Then, this page     is swapped out (to disk) and then it is dropped
17272                              <1> ;;    from the list by a final swap queue shift. [swpq_count] value
17273                              <1> ;;    is changed. If all pages on the queue' are 'accessed',
17274                              <1> ;;    'insufficient memory' error will be returned ('swap_out'
17275                              <1> ;;    procedure will be failed)...
17276                              <1> ;;
17277                              <1> ;;    Note: If the 1st page of the queue is an 'accessed' page,
17278                              <1> ;;    'accessed' flag of the page will be reset (0) and that page
17279                              <1> ;;    (PTE) will be added to the tail of the queue after
17280                              <1> ;;    the check, if [swpq_count] < 1023. If [swpq_count] = 1024
17281                              <1> ;;    the queue will be rotated and the PTE in the head will be
17282                              <1> ;;    added to the tail after resetting 'accessed' bit.
17283                              <1> ;;
17284                              <1> ;;
17285                              <1> ;;
17286                              <1> ;; SWAP DISK/FILE (with 4096 bytes swapped page blocks)
17287                              <1> ;;
17288                              <1> ;;  00000000  00000004  00000008  0000000C   ...   size-8    size-4
17289                              <1> ;; +---------+---------+---------+---------+-- --+---------+---------+
17290                              <1> ;; |descriptr| page(1) | page(2) | page(3) | ... |page(n-1)| page(n) |
17291                              <1> ;; +---------+---------+---------+---------+-- --+---------+---------+
17292                              <1> ;;
17293                              <1> ;; [swpd_next] = the first free block address in swapped page records
17294                              <1> ;;                  for next free block search by 'swap_out' procedure.
17295                              <1> ;; [swpd_size] = swap disk/file size in sectors (512 bytes)
17296                              <1> ;;           NOTE: max. possible swap disk size is 1024 GB
17297                              <1> ;;           (entire swap space must be accessed by using
17298                              <1> ;;           31 bit offset address)
17299                              <1> ;; [swpd_free] = free block (4096 bytes) count in swap disk/file space
17300                              <1> ;; [swpd_start] = absolute/start address of the swap disk/file
17301                              <1> ;;            0 for file, or beginning sector of the swap partition
```

```
17302                           <1> ;; [swp_drv] = logical drive description table addr. of swap disk/file
17303                           <1> ;;
17304                           <1> ;;
17305                           <1> ;; Method:
17306                           <1> ;;    When the memory (ram) becomes insufficient, page allocation
17307                           <1> ;;    procedure swaps out a page from memory to the swap disk
17308                           <1> ;;    (partition) or swap file to get a new free page at the memory.
17309                           <1> ;;    Swapping out is performed by using swap page queue.
17310                           <1> ;;
17311                           <1> ;;    Allocation block size of swap disk/file is equal to page size
17312                           <1> ;;    (4096 bytes). Swapping address (in sectors) is recorded
17313                           <1> ;;    into relevant page file entry as 31 bit physical (logical)
17314                           <1> ;;    offset address as 1 bit shifted to left for present flag (0).
17315                           <1> ;;    Swapped page address is between 1 and swap disk/file size - 4.
17316                           <1> ;;    Absolute physical (logical) address of the swapped page is
17317                           <1> ;;    calculated by adding offset value to the swap partition's
17318                           <1> ;;    start address. If the swap device (disk) is a virtual disk
17319                           <1> ;;    or it is a file, start address of the swap disk/volume is 0,
17320                           <1> ;;    and offset value is equal to absolute (physical or logical)
17321                           <1> ;;    address/position. (It has not to be ZERO if the swap partition
17322                           <1> ;;    is in a partitioned virtual hard disk.)
17323                           <1> ;;
17324                           <1> ;;    Note: Swap addresses are always specified/declared in sectors,
17325                           <1> ;;    not in bytes or     in blocks/zones/clusters (4096 bytes) as unit.
17326                           <1> ;;
17327                           <1> ;;    Swap disk/file allocation is mapped via 'Swap Allocation Table'
17328                           <1> ;;    at memory as similar to 'Memory Allocation Table'.
17329                           <1> ;;
17330                           <1> ;;    Every bit of Swap Allocation Table repsesents one swap block
17331                           <1> ;;    (equal to page size) respectively. Bit 0 of the S.A.T. byte 0
17332                           <1> ;;    is reserved for swap disk/file block 0 as descriptor block
17333                           <1> ;;    (also for compatibility with PTE). If bit value is ZERO,
17334                           <1> ;;    it means relevant (respective) block is in use, and,
17335                           <1> ;;    of course, if bit value is 1, it means relevant (respective)
17336                           <1> ;;      swap disk/file block is free.
17337                           <1> ;;    For example: bit 1 of the byte 128 repsesents block 1025
17338                           <1> ;;    (128*8+1) or sector (offset) 8200 on the swap disk or
17339                           <1> ;;    byte (offset/position) 4198400 in the swap file.
17340                           <1> ;;    4GB swap space is represented via 128KB Swap Allocation Table.
17341                           <1> ;;    Initial layout of Swap Allocation Table is as follows:
17342                           <1> ;;    ------------------------------------------------------------
17343                           <1> ;;    01111111111111111111111111 .... 11111111111111111111111111111111
17344                           <1> ;;    ------------------------------------------------------------
17345                           <1> ;;    (0 is reserved block, 1s represent free blocks respectively.)
17346                           <1> ;;    (Note: Allocation cell/unit of the table is bit, not byte)
17347                           <1> ;;
17348                           <1> ;;    ............................................................
17349                           <1> ;;
17350                           <1> ;;    'swap_out' procedure checks 'free_swap_blocks' count at first,
17351                           <1> ;;    then it searches Swap Allocation Table if free count is not
17352                           <1> ;;    zero. From begining the [swpd_next] dword value, the first bit
17353                           <1> ;;    position with value of 1 on the table is converted to swap
17354                           <1> ;;    disk/file offset address, in sectors (not 4096 bytes block).
17355                           <1> ;;    'ldrv_write' procedure is called with ldrv (logical drive
17356                           <1> ;;    number of physical swap disk or virtual swap disk)
17357                           <1> ;;    number, sector offset (not absolute sector -LBA- number),
17358                           <1> ;;    and sector count (8, 512*8 = 4096) and buffer adress
17359                           <1> ;;    (memory page). That will be a direct disk write procedure.
17360                           <1> ;;    (for preventing late memory allocation, significant waiting).
17361                           <1> ;;    If disk write procedure returns with error or free count of
17362                           <1> ;;    swap blocks is ZERO, 'swap_out' procedure will return with
17363                           <1> ;;    'insufficient memory error' (cf=1).
17364                           <1> ;;
17365                           <1> ;;    (Note: Even if free swap disk/file blocks was not zero,
17366                           <1> ;;    any disk write error will not be fixed by 'swap_out' procedure,
17367                           <1> ;;    in other words, 'swap_out' will not check the table for other
17368                           <1> ;;    free blocks after a disk write error. It will return to
17369                           <1> ;;    the caller with error (CF=1) which means swapping is failed.
17370                           <1> ;;
17371                           <1> ;;    After writing the page on to swap disk/file address/sector,
17372                           <1> ;;    'swap_out' procesure returns with that swap (offset) sector
17373                           <1> ;;    address (cf=0).
17374                           <1> ;;
17375                           <1> ;;    ............................................................
17376                           <1> ;;
17377                           <1> ;;    'swap_in' procedure loads addressed (relevant) swap disk or
17378                           <1> ;;    file sectors at specified memory page. Then page allocation
17379                           <1> ;;    procedure updates relevant page table entry with 'present'
17380                           <1> ;;    attribute. If swap disk or file reading fails there is nothing
17381                           <1> ;;    to do, except to terminate the process which is the owner of
17382                           <1> ;;    the swapped page.
17383                           <1> ;;
17384                           <1> ;;    'swap_in' procedure sets the relevant/respective bit value
17385                           <1> ;;    in the Swap Allocation Table (as free block). 'swap_in' also
17386                           <1> ;;    updates [swpd_first] pointer if it is required.
17387                           <1> ;;
17388                           <1> ;;    ............................................................
17389                           <1> ;;
17390                           <1> ;;    Note: If [swap_enabled] value is ZERO, that means there is not
17391                           <1> ;;    a swap disk or swap file in use... 'swap_in' and 'swap_out'
17392                           <1> ;;    procedures ans 'swap page que' procedures will not be active...
17393                           <1> ;;    'Insufficient memory' error will be returned by 'swap_out'
17394                           <1> ;;    and 'general protection fault' will be returned by 'swap_in'
17395                           <1> ;;    procedure, if it is called mistakenly (a wrong value in a PTE).
17396                           <1> ;;
17397                           <1>
17398                           <1> swap_in:
17399                           <1>       ; 31/08/2015
17400                           <1>       ; 20/07/2015
17401                           <1>       ; 28/04/2015
17402                           <1>       ; 18/04/2015
17403                           <1>       ; 24/10/2014 (Retro UNIX 386 v1 - beginning)
17404                           <1>       ;
```

```
17405                                  <1>       ; INPUT ->
17406                                  <1>       ;     EBX = PHYSICAL (real/flat) ADDRESS OF THE MEMORY PAGE
17407                                  <1>       ;     EBP = VIRTUAL (LINEAR) ADDRESS (page fault address)
17408                                  <1>       ;     EAX = Offset Address for the swapped page on the
17409                                  <1>       ;           swap disk or in the swap file.
17410                                  <1>       ;
17411                                  <1>       ; OUTPUT ->
17412                                  <1>       ;     EAX = 0 if loading at memory has been successful
17413                                  <1>       ;
17414                                  <1>       ;     CF = 1 -> swap disk reading error (disk/file not present
17415                                  <1>       ;             or sector not present or drive not ready
17416                                  <1>       ;          EAX = Error code
17417                                  <1>       ;          [u.error] = EAX
17418                                  <1>       ;                  = The last error code for the process
17419                                  <1>       ;                    (will be reset after returning to user)
17420                                  <1>       ;
17421                                  <1>       ; Modified Registers -> EAX
17422                                  <1>       ;
17423                                  <1>
17424 00004FB9 833D[62050300]00       <1>         cmp     dword [swp_drv], 0
17425 00004FC0 7646                   <1>       jna   short swpin_dnp_err
17426                                  <1>
17427 00004FC2 3B05[66050300]         <1>       cmp   eax, [swpd_size]
17428 00004FC8 734A                   <1>       jnb   short swpin_snp_err
17429                                  <1>
17430 00004FCA 56                     <1>       push  esi
17431 00004FCB 53                     <1>       push  ebx
17432 00004FCC 51                     <1>       push  ecx
17433 00004FCD 8B35[62050300]         <1>       mov   esi, [swp_drv]
17434 00004FD3 B908000000             <1>       mov   ecx, PAGE_SIZE / LOGIC_SECT_SIZE  ; 8 !
17435                                  <1>             ; Note: Even if corresponding physical disk's sector
17436                                  <1>             ; size different than 512 bytes, logical disk sector
17437                                  <1>             ; size is 512 bytes and disk reading procedure
17438                                  <1>             ; will be performed for reading 4096 bytes
17439                                  <1>             ; (2*2048, 8*512).
17440                                  <1>       ; ESI = Logical disk description table address
17441                                  <1>       ; EBX = Memory page (buffer) address (physical!)
17442                                  <1>       ; EAX = Sector adress (offset address, logical sector number)
17443                                  <1>       ; ECX = Sector count ; 8 sectors
17444 00004FD8 50                     <1>       push  eax
17445 00004FD9 E8AF020000             <1>       call  logical_disk_read
17446 00004FDE 58                     <1>       pop   eax
17447 00004FDF 730C                   <1>       jnc   short swpin_read_ok
17448                                  <1>       ;
17449 00004FE1 B828000000             <1>       mov   eax, SWP_DISK_READ_ERR ; drive not ready or read error
17450 00004FE6 A3[C8030300]           <1>       mov   [u.error], eax
17451 00004FEB EB17                   <1>       jmp   short swpin_retn
17452                                  <1>       ;
17453                                  <1> swpin_read_ok:
17454                                  <1>       ; EAX = Offset address (logical sector number)
17455 00004FED E80D020000             <1>       call  unlink_swap_block  ; Deallocate swap block
17456                                  <1>       ;
17457                                  <1>       ; EBX = Memory page (buffer) address (physical!)
17458                                  <1>       ; 20/07/2015
17459 00004FF2 89EB                   <1>       mov   ebx, ebp ; virtual address (page fault address)
17460 00004FF4 6681E300F0             <1>         and     bx, ~PAGE_OFF ; ~0FFFh ; reset bits, 0 to 11
17461 00004FF9 8A1D[B3030300]         <1>       mov   bl, [u.uno] ; current process number
17462                                  <1>       ; EBX = Virtual (Linear) address & process number combination
17463 00004FFF E8DB000000             <1>       call  swap_queue_shift
17464                                  <1>       ; eax = 0 ; 10/06/2016 (if ebx input > 0, eax output = 0)
17465                                  <1>       ;sub   eax, eax  ; 0 ; Error Code = 0  (no error)
17466                                  <1>       ; zf = 1
17467                                  <1> swpin_retn:
17468 00005004 59                     <1>       pop   ecx
17469 00005005 5B                     <1>       pop   ebx
17470 00005006 5E                     <1>       pop   esi
17471 00005007 C3                     <1>       retn
17472                                  <1>
17473                                  <1> swpin_dnp_err:
17474 00005008 B829000000             <1>       mov   eax, SWP_DISK_NOT_PRESENT_ERR
17475                                  <1> swpin_err_retn:
17476 0000500D A3[C8030300]           <1>       mov   [u.error], eax
17477 00005012 F9                     <1>       stc
17478 00005013 C3                     <1>       retn
17479                                  <1>
17480                                  <1> swpin_snp_err:
17481 00005014 B82A000000             <1>       mov   eax, SWP_SECTOR_NOT_PRESENT_ERR
17482 00005019 EBF2                   <1>       jmp   short swpin_err_retn
17483                                  <1>
17484                                  <1> swap_out:
17485                                  <1>       ; 10/06/2016
17486                                  <1>       ; 07/06/2016
17487                                  <1>         ; 23/05/2016
17488                                  <1>       ; 19/05/2016 - TRDOS 386 (TRDOS v2.0)
17489                                  <1>       ; 24/10/2014 - 31/08/2015 (Retro UNIX 386 v1)
17490                                  <1>       ;
17491                                  <1>       ; INPUT ->
17492                                  <1>       ;     none
17493                                  <1>       ;
17494                                  <1>       ; OUTPUT ->
17495                                  <1>       ;     EAX = Physical page address (which is swapped out
17496                                  <1>       ;           for allocating a new page)
17497                                  <1>       ;     CF = 1 -> swap disk writing error (disk/file not present
17498                                  <1>       ;             or sector not present or drive not ready
17499                                  <1>       ;          EAX = Error code
17500                                  <1>       ;          [u.error] = EAX
17501                                  <1>       ;                  = The last error code for the process
17502                                  <1>       ;                    (will be reset after returning to user)
17503                                  <1>       ;
17504                                  <1>       ; Modified Registers -> none (except EAX)
17505                                  <1>       ;
17506 0000501B 66833D[60050300]01     <1>       cmp   word [swpq_count], 1
17507 00005023 0F82AF000000           <1>         jc      swpout_im_err ; 'insufficient memory'
```

```
                                <1>
                                <1>         ;cmp    dword [swp_drv], 1
                                <1>       ;jc    short swpout_dnp_err ; 'swap disk/file not present'
                                <1>
00005029 833D[6A050300]01       <1>         cmp     dword [swpd_free], 1
00005030 0F828F000000           <1>         jc      swpout_nfspc_err ; 'no free space on swap disk'
                                <1>
00005036 53                     <1>       push   ebx ; *
                                <1> swpout_1:
                                <1>       ; 10/06/2016
00005037 31DB                   <1>       xor    ebx, ebx ; shift the queue and return a PTE value
00005039 E8A1000000             <1>       call   swap_queue_shift
0000503E 21C0                   <1>       and    eax, eax     ; 0 = empty queue (improper entries)
00005040 0F848A000000           <1>         jz      swpout_npts_err        ; There is not any proper PTE
                                <1>                                      ; pointer in the swap queue
                                <1>       ; EAX = PTE value of the page
                                <1>       ; EBX = PTE address of the page
00005046 662500F0               <1>       and    ax, PTE_A_CLEAR ; 0F000h ; clear attribute bits
                                <1>       ;
                                <1>       ; 07/06/2016
                                <1>       ; 19/05/2016
                                <1>       ; check this page is in timer events or not
                                <1>
                                <1> swpout_timer_page_0:
0000504A 52                     <1>       push   edx ; **
                                <1>
                                <1>       ; 07/06/2016
0000504B 803D[B75F0100]00       <1>       cmp    byte [timer_events], 0
00005052 762F                   <1>       jna    short swpout_2
                                <1>       ;
00005054 8A15[B75F0100]         <1>       mov    dl, [timer_events]
                                <1>
0000505A 51                     <1>       push   ecx ; ***
0000505B 53                     <1>       push   ebx ; ****
0000505C BB[60040300]           <1>       mov    ebx, timer_set ; beginning address of timer event
                                <1>                             ; structures
                                <1> swpout_timer_page_1:
00005061 8A0B                   <1>       mov    cl, [ebx]
00005063 08C9                   <1>       or     cl, cl ; 0 = free, >0 = process number
00005065 7415                   <1>       jz     short swpout_timer_page_3
00005067 8B4B0C                 <1>       mov    ecx, [ebx+12] ; response (signal return) address
0000506A 6681E100F0             <1>       and    cx, PTE_A_CLEAR ; clear offset part (right 12 bits)
                                <1>                            ; of the response byte address, to
                                <1>                            ; get beginning of the page address)
0000506F 39C8                   <1>       cmp    eax, ecx
00005071 7505                   <1>       jne    short swpout_timer_page_2 ; not same page
                                <1>
                                <1>       ; !same page!
                                <1>       ;
                                <1>       ; NOTE: // 19/05/2016 // - TRDOS 386 feature only ! -
                                <1>       ; This page will be used by the kernel to put timer event
                                <1>       ; response (signal return) byte at the requested address;
                                <1>       ; in order to prevent a possible wrong write (while
                                <1>       ; this page is swapped out) on physical memory,
                                <1>       ; we must protect this page against to be swapped out!
                                <1>       ;
00005073 5B                     <1>       pop    ebx ; ****
00005074 59                     <1>       pop    ecx ; ***
00005075 5A                     <1>       pop    edx ; **
00005076 EBBF                   <1>       jmp    short swpout_1      ; do not swap out this page !
                                <1>
                                <1> swpout_timer_page_2:
                                <1>       ; 07/06/2016
00005078 FECA                   <1>       dec    dl
0000507A 7405                   <1>       jz     short swpout_timer_page_4
                                <1> swpout_timer_page_3:
                                <1>       ;cmp   ebx, timer_set + 240 ; last timer event (15*16)
                                <1>       ;jnb   short swpout_timer_page_4
0000507C 83C310                 <1>       add    ebx, 16
0000507F EBE0                   <1>       jmp    short swpout_timer_page_1
                                <1>
                                <1> swpout_timer_page_4:
00005081 5B                     <1>       pop    ebx ; ****
00005082 59                     <1>       pop    ecx ; ***
                                <1> swpout_2:
00005083 89DA                   <1>       mov    edx, ebx            ; Page table entry address
00005085 89C3                   <1>       mov    ebx, eax            ; Buffer (Page) Address
                                <1>       ;
00005087 E8A6010000             <1>       call   link_swap_block
0000508C 7304                   <1>       jnc    short swpout_3             ; It may not be needed here
                                <1>                                       ; because [swpd_free] value
                                <1>                                       ; was checked at the beginging.
0000508E 5A                     <1>       pop    edx ; **
0000508F 5B                     <1>       pop    ebx ; *
00005090 EB33                   <1>       jmp    short swpout_nfspc_err
                                <1> swpout_3:
00005092 A900000080             <1>       test   eax, 80000000h ; test bit 31 (this may not be needed!)
00005097 752C                   <1>       jnz    short swpout_nfspc_err  ; 10/06/2016 (bit 31 = 1 !)
                                <1>       ;
00005099 56                     <1>       push   esi ; **
0000509A 51                     <1>       push   ecx ; ***
0000509B 50                     <1>       push   eax ; sector address ; (31 bit !, bit 31 = 0)
0000509C 8B35[62050300]         <1>       mov    esi, [swp_drv]
000050A2 B908000000             <1>       mov    ecx, PAGE_SIZE / LOGIC_SECT_SIZE  ; 8 !
                                <1>              ; Note: Even if corresponding physical disk's sector
                                <1>              ; size different than 512 bytes, logical disk sector
                                <1>              ; size is 512 bytes and disk writing procedure
                                <1>              ; will be performed for writing 4096 bytes
                                <1>              ; (2*2048, 8*512).
                                <1>       ; ESI = Logical disk description table address
                                <1>       ; EBX = Buffer (Page) address
                                <1>       ; EAX = Sector adress (offset address, logical sector number)
                                <1>       ; ECX = Sector count ; 8 sectors
```

```
17611                                  <1>       ; edx = PTE address
17612 000050A7 E8E2010000              <1>       call   logical_disk_write
17613                                  <1>       ; edx = PTE address
17614 000050AC 59                      <1>       pop    ecx ; sector address
17615 000050AD 730C                    <1>       jnc    short swpout_write_ok
17616                                  <1>       ;
17617                                  <1>       ;; call      unlink_swap_block ; this block must be left as 'in use'
17618                                  <1> swpout_dw_err:
17619 000050AF B82C000000              <1>       mov    eax, SWP_DISK_WRITE_ERR ; drive not ready or write error
17620 000050B4 A3[C8030300]            <1>       mov    [u.error], eax
17621 000050B9 EB06                    <1>       jmp    short swpout_retn
17622                                  <1>       ;
17623                                  <1> swpout_write_ok:
17624                                  <1>       ; EBX = Buffer (page) address
17625                                  <1>       ; EDX = Page Table Entry address
17626                                  <1>       ; ECX = Swap disk sector (file block) address (31 bit)
17627 000050BB D1E1                    <1>       shl    ecx, 1  ; 31 bit sector address from bit 1 to bit 31
17628 000050BD 890A                    <1>       mov    [edx], ecx
17629                                  <1>              ; bit 0 = 0 (swapped page)
17630 000050BF 89D8                    <1>       mov    eax, ebx
17631                                  <1> swpout_retn:
17632 000050C1 59                      <1>       pop    ecx ; ***
17633 000050C2 5E                      <1>       pop    esi ; **
17634 000050C3 5B                      <1>       pop    ebx ; *
17635 000050C4 C3                      <1>       retn
17636                                  <1>
17637                                  <1> ;swpout_dnp_err:
17638                                  <1> ;     mov    eax, SWP_DISK_NOT_PRESENT_ERR ; disk not present
17639                                  <1> ;     jmp    short swpout_err_retn
17640                                  <1> swpout_nfspc_err:
17641 000050C5 B82B000000              <1>       mov    eax, SWP_NO_FREE_SPACE_ERR ; no free space
17642                                  <1> swpout_err_retn:
17643 000050CA A3[C8030300]            <1>       mov    [u.error], eax
17644                                  <1>       ;stc
17645 000050CF C3                      <1>       retn
17646                                  <1> swpout_npts_err:
17647 000050D0 B82D000000              <1>       mov    eax, SWP_NO_PAGE_TO_SWAP_ERR
17648 000050D5 5B                      <1>       pop    ebx
17649 000050D6 EBF2                    <1>       jmp    short swpout_err_retn
17650                                  <1> swpout_im_err:
17651 000050D8 B804000000              <1>       mov    eax, ERR_MINOR_IM ; insufficient (out of) memory
17652 000050DD EBEB                    <1>       jmp    short swpout_err_retn
17653                                  <1>
17654                                  <1> swap_queue_shift:
17655                                  <1>       ; 26/03/2017
17656                                  <1>       ; 10/06/2016
17657                                  <1>       ; 09/06/2016 - TRDOS 386 (TRDOS v2.0)
17658                                  <1>       ; 23/10/2014 - 20/07/2015 (Retro UNIX 386 v1)
17659                                  <1>       ;
17660                                  <1>       ; INPUT ->
17661                                  <1>       ;     EBX = Virtual (linear) address (bit 12 to 31)
17662                                  <1>       ;           and process number combination (bit 0 to 11)
17663                                  <1>       ;     EBX = 0 -> shift/drop from the head (offset 0)
17664                                  <1>       ;
17665                                  <1>       ; OUTPUT ->
17666                                  <1>       ;     If EBX input > 0
17667                                  <1>       ;        the queue will be shifted 4 bytes (dword),
17668                                  <1>       ;        from the tail to the head, up to entry offset
17669                                  <1>       ;        which points to EBX input value or nothing
17670                                  <1>       ;        to do if EBX value is not found on the queue.
17671                                  <1>       ;        (The entry -with EBX value- will be removed
17672                                  <1>       ;        from the queue if it is found.)
17673                                  <1>       ;
17674                                  <1>       ;        EAX = 0
17675                                  <1>       ;
17676                                  <1>       ;     If EBX input = 0
17677                                  <1>       ;        the queue will be shifted 4 bytes (dword),
17678                                  <1>       ;        from the tail to the head, if the PTE address
17679                                  <1>       ;        which is pointed in head of the queue is marked
17680                                  <1>       ;        as "accessed" or it is marked as "non present".
17681                                  <1>       ;        (If "accessed" flag of the PTE -which is pointed
17682                                  <1>       ;        in the head- is set -to 1-, it will be reset
17683                                  <1>       ;        -to 0- and then, the queue will be rotated
17684                                  <1>       ;        -without dropping pointer of the PTE from
17685                                  <1>       ;        the queue- for 4 bytes on head to tail direction.
17686                                  <1>       ;        Pointer in the head will be moved into the tail,
17687                                  <1>       ;        other PTEs will be shifted on head direction.)
17688                                  <1>       ;
17689                                  <1>       ;        Swap queue will be shifted up to the first
17690                                  <1>       ;        'present' or 'non accessed' page will be found
17691                                  <1>       ;        (as pointed) on the queue head (then it will be
17692                                  <1>         ;         removed/dropped from the queue).
17693                                  <1>       ;
17694                                  <1>       ;        EAX (> 0) = PTE value of the page which is
17695                                  <1>       ;             (it's pointer -virtual address-) dropped
17696                                  <1>       ;             (removed) from swap queue.
17697                                  <1>       ;        EBX = PTE address of the page (if EAX > 0)
17698                                  <1>       ;              which is (it's pointer -virtual address-)
17699                                  <1>       ;             dropped (removed) from swap queue.
17700                                  <1>       ;
17701                                  <1>       ;        EAX = 0 -> empty swap queue !
17702                                  <1>       ;
17703                                  <1>       ; Modified Registers -> EAX, EBX
17704                                  <1>       ;
17705 000050DF 0FB705[60050300]        <1>       movzx   eax, word [swpq_count]  ; Max. 1024
17706 000050E6 6621C0                  <1>       and    ax, ax
17707 000050E9 7431                    <1>       jz     short swpqs_retn
17708 000050EB 57                      <1>       push   edi
17709 000050EC 56                      <1>       push   esi
17710 000050ED 51                      <1>       push   ecx
17711 000050EE BE00E00800              <1>       mov    esi, swap_queue
17712 000050F3 89C1                    <1>       mov    ecx, eax
17713 000050F5 09DB                    <1>       or     ebx, ebx
```

```
17714 000050F7 7424                <1>       jz     short swpqs_7
17715                              <1> swpqs_1:
17716 000050F9 AD                  <1>       lodsd
17717 000050FA 39D8                <1>       cmp    eax, ebx
17718 000050FC 7406                <1>       je     short swpqs_2
17719 000050FE E2F9                <1>       loop   swpqs_1
17720                              <1>       ; 10/06/2016
17721 00005100 29C0                <1>       sub    eax, eax
17722 00005102 EB15                <1>       jmp    short swpqs_6
17723                              <1> swpqs_2:
17724 00005104 89F7                <1>       mov    edi, esi
17725 00005106 83EF04              <1>       sub    edi, 4
17726                              <1> swpqs_3:
17727 00005109 66FF0D[60050300]    <1>       dec    word [swpq_count]
17728 00005110 7403                <1>       jz     short swpqs_5
17729                              <1> swpqs_4:
17730 00005112 49                  <1>       dec    ecx
17731 00005113 F3A5                <1>       rep    movsd  ; shift up (to the head)
17732                              <1> swpqs_5:
17733 00005115 31C0                <1>       xor    eax, eax
17734 00005117 8907                <1>       mov    [edi], eax
17735                              <1> swpqs_6:
17736 00005119 59                  <1>       pop    ecx
17737 0000511A 5E                  <1>       pop    esi
17738 0000511B 5F                  <1>       pop    edi
17739                              <1> swpqs_retn:
17740 0000511C C3                  <1>       retn
17741                              <1> swpqs_7:
17742 0000511D 89F7                <1>       mov    edi, esi ; head
17743 0000511F AD                  <1>       lodsd
17744                              <1>       ; 20/07/2015
17745 00005120 89C3                <1>       mov    ebx, eax
17746 00005122 81E300F0FFFF        <1>       and    ebx, ~PAGE_OFF ; ~0FFFh
17747                              <1>                ; ebx = virtual address (at page boundary)
17748 00005128 25FF0F0000          <1>       and    eax, PAGE_OFF ; 0FFFh
17749                              <1>                ; ax = process number (1 to 4095)
17750 0000512D 3A05[B3030300]      <1>       cmp    al, [u.uno]
17751                              <1>              ; Max. 16 (nproc) processes for Retro UNIX 386 v1
17752 00005133 7507                <1>       jne    short swpqs_8
17753 00005135 A1[B8030300]        <1>       mov    eax, [u.pgdir]
17754 0000513A EB28                <1>       jmp    short swpqs_9
17755                              <1> swpqs_8:
17756                              <1>       ; 09/06/2016
17757 0000513C 80B8[AF000300]00    <1>       cmp    byte [eax+p.stat-1], 0
17758 00005143 76C4                <1>       jna    short swpqs_3     ; free (or terminated) process
17759 00005145 80B8[AF000300]02    <1>       cmp    byte [eax+p.stat-1], 2 ; waiting
17760 0000514C 77BB                <1>       ja     short swpqs_3        ; zombie (3) or undefined ?
17761                              <1>
17762                              <1>       ;shl   ax, 2
17763 0000514E C0E002              <1>       shl    al, 2
17764 00005151 8B80[BC000300]      <1>       mov    eax, [eax+p.upage-4]
17765 00005157 09C0                <1>       or     eax, eax
17766 00005159 74AE                <1>       jz     short swpqs_3 ; invalid upage
17767 0000515B 83C05C              <1>       add    eax, u.pgdir - user
17768                              <1>                      ; u.pgdir value for the process
17769                              <1>                      ; is in [eax]
17770 0000515E 8B00                <1>       mov    eax, [eax]
17771 00005160 21C0                <1>       and    eax, eax
17772 00005162 74A5                <1>       jz     short swpqs_3 ; invalid page directory
17773                              <1> swpqs_9:
17774 00005164 52                  <1>       push   edx
17775                              <1>       ; eax = page directory
17776                              <1>       ; ebx = virtual address
17777 00005165 E82BFBFFFF          <1>       call   get_pte
17778 0000516A 89D3                <1>       mov    ebx, edx     ; PTE address
17779 0000516C 5A                  <1>       pop    edx
17780                              <1>       ; 10/06/2016
17781 0000516D 723A                <1>       jc     short swpqs_13 ; empty PDE
17782                              <1>       ; EAX = PTE value
17783 0000516F A801                <1>       test   al, PTE_A_PRESENT ; bit 0 = 1
17784 00005171 7436                <1>       jz     short swpqs_13  ; Drop non-present page
17785                              <1>                              ; from the queue (head)
17786 00005173 A802                <1>       test   al, PTE_A_WRITE     ; bit 1 = 0 (read only)
17787 00005175 7432                <1>       jz     short swpqs_13  ; Drop read only page
17788                              <1>                              ; from the queue (head)
17789                              <1>       ;test  al, PTE_A_ACCESS ; bit 5 = 1 (Accessed)
17790                              <1>       ;jnz   short swpqs_11  ; present
17791                              <1>                              ; accessed page
17792 00005177 0FBAF005            <1>         btr     eax, PTE_A_ACCESS_BIT ; reset 'accessed' bit
17793 0000517B 7210                <1>       jc     short swpqs_11  ; accessed page
17794                              <1>
17795 0000517D 49                  <1>       dec    ecx
17796 0000517E 66890D[60050300]    <1>       mov    [swpq_count], cx
17797 00005185 7402                <1>         jz      short swpqs_10
17798                              <1>              ; esi = head + 4
17799                              <1>              ; edi = head
17800 00005187 F3A5                <1>       rep    movsd  ; n = 1 to k-1, [n - 1] = [n]
17801                              <1> swpqs_10:
17802 00005189 890F                <1>       mov    [edi], ecx ; 0
17803 0000518B EB8C                <1>       jmp    short swpqs_6 ; 26/03/2017
17804                              <1>
17805                              <1> swpqs_11:
17806 0000518D 8903                <1>       mov    [ebx], eax     ; save changed attribute
17807                              <1>       ; Rotation (head -> tail)
17808 0000518F 49                  <1>       dec    ecx     ; entry count -> last entry number
17809 00005190 74F7                <1>       jz     short swpqs_10
17810                              <1>              ; esi = head + 4
17811                              <1>              ; edi = head
17812 00005192 8B07                <1>       mov    eax, [edi] ; 20/07/2015
17813 00005194 F3A5                <1>       rep    movsd  ; n = 1 to k-1, [n - 1] = [n]
17814 00005196 8907                <1>       mov    [edi], eax ; head -> tail ; [k] = [1]
17815                              <1>
17816 00005198 668B0D[60050300]    <1>       mov    cx, [swpq_count]
```

```
                                <1>
                                <1> swpqs_12:
0000519F BE00E00800             <1>       mov    esi, swap_queue ; head
000051A4 E974FFFFFF             <1>         jmp     swpqs_7
                                <1>
                                <1> swpqs_13:
000051A9 49                     <1>       dec    ecx
000051AA 66890D[60050300]       <1>       mov    [swpq_count], cx
000051B1 0F845EFFFFFF           <1>         jz      swpqs_5
000051B7 EBE6                   <1>       jmp    short swpqs_12
                                <1>
                                <1> add_to_swap_queue:
                                <1> ; temporary - 16/09/2015
000051B9 C3                     <1> retn
                                <1>       ; 20/02/2017
                                <1>       ; 20/07/2015
                                <1>       ; 24/10/2014 (Retro UNIX 386 v1 - beginning)
                                <1>       ;
                                <1>       ; Adds new page to swap queue
                                <1>       ; (page directories and page tables must not be added
                                <1>       ; to swap queue)
                                <1>       ;
                                <1>       ; INPUT ->
                                <1>       ;     EBX = Linear (Virtual) addr for current process
                                <1>       ;     [u.uno]
                                <1>       ;     20/02/2017
                                <1>       ;     (Linear address = CORE + user's virtual address)
                                <1>       ;
                                <1>       ; OUTPUT ->
                                <1>       ;     EAX = [swpq_count]
                                <1>       ;           (after the PTE has been added)
                                <1>       ;     EAX = 0 -> Swap queue is full, (1024 entries)
                                <1>       ;           the PTE could not be added.
                                <1>       ;
                                <1>       ; Modified Registers -> EAX
                                <1>       ;
000051BA 53                     <1>       push   ebx
000051BB 6681E300F0             <1>         and     bx, ~PAGE_OFF ; ~0FFFh ; reset bits, 0 to 11
000051C0 8A1D[B3030300]         <1>       mov    bl, [u.uno] ; current process number
000051C6 E814FFFFFF             <1>       call   swap_queue_shift ; drop from the queue if
                                <1>                               ; it is already on the queue
                                <1>             ; then add it to the tail of the queue
000051CB 0FB705[60050300]       <1>       movzx  eax, word [swpq_count]
000051D2 663D0004               <1>       cmp    ax, 1024
000051D6 7205                   <1>       jb     short atsq_1
000051D8 6629C0                 <1>       sub    ax, ax
000051DB 5B                     <1>       pop    ebx
000051DC C3                     <1>       retn
                                <1> atsq_1:
000051DD 56                     <1>       push   esi
000051DE BE00E00800             <1>       mov    esi, swap_queue
000051E3 6621C0                 <1>       and    ax, ax
000051E6 740A                   <1>       jz     short atsq_2
000051E8 66C1E002               <1>       shl    ax, 2  ; convert to offset
000051EC 01C6                   <1>       add    esi, eax
000051EE 66C1E802               <1>       shr    ax, 2
                                <1> atsq_2:
000051F2 6640                   <1>       inc    ax
000051F4 891E                   <1>       mov    [esi], ebx ; Virtual address + [u.uno] combination
000051F6 66A3[60050300]         <1>       mov    [swpq_count], ax
000051FC 5E                     <1>       pop    esi
000051FD 5B                     <1>       pop    ebx
000051FE C3                     <1>       retn
                                <1>
                                <1> unlink_swap_block:
                                <1>       ; 15/09/2015
                                <1>       ; 30/04/2015
                                <1>       ; 18/04/2015
                                <1>       ; 24/10/2014 (Retro UNIX 386 v1 - beginning)
                                <1>       ;
                                <1>       ; INPUT ->
                                <1>       ;     EAX = swap disk/file offset address
                                <1>       ;           (bit 1 to bit 31)
                                <1>       ; OUTPUT ->
                                <1>       ;     [swpd_free] is increased
                                <1>       ;     (corresponding SWAP DISK ALLOC. TABLE bit is SET)
                                <1>       ;
                                <1>       ; Modified Registers -> EAX
                                <1>       ;
000051FF 53                     <1>       push   ebx
00005200 52                     <1>       push   edx
                                <1>       ;
00005201 C1E804                 <1>       shr    eax, SECTOR_SHIFT+1  ;3+1 ; shift sector address to
                                <1>                                   ; 3 bits right
                                <1>                                   ; to get swap block/page number
00005204 89C2                   <1>       mov    edx, eax
                                <1>       ; 15/09/2015
00005206 C1EA03                 <1>       shr    edx, 3            ; to get offset to S.A.T.
                                <1>                                   ; (1 allocation bit = 1 page)
                                <1>                                   ; (1 allocation bytes = 8 pages)
00005209 80E2FC                 <1>       and    dl, 0FCh          ; clear lower 2 bits
                                <1>                                   ; (to get 32 bit position)
                                <1>       ;
0000520C BB00000D00             <1>       mov    ebx, swap_alloc_table ; Swap Allocation Table address
00005211 01D3                   <1>       add    ebx, edx
00005213 83E01F                 <1>       and    eax, 1Fh          ; lower 5 bits only
                                <1>                                   ; (allocation bit position)
00005216 3B05[6E050300]         <1>       cmp    eax, [swpd_next]     ; is the new free block addr. lower
                                <1>                                   ; than the address in 'swpd_next' ?
                                <1>                                   ; (next/first free block value)
0000521C 7305                   <1>       jnb    short uswpbl_1          ; no
0000521E A3[6E050300]           <1>       mov    [swpd_next], eax     ; yes
                                <1> uswpbl_1:
```

```
17920 00005223 0FAB03              <1>       bts    [ebx], eax        ; unlink/release/deallocate block
17921                              <1>                                  ; set relevant bit to 1.
17922                              <1>                                  ; set CF to the previous bit value
17923 00005226 F5                  <1>       cmc                        ; complement carry flag
17924 00005227 7206                <1>       jc     short uswpbl_2           ; do not increase swfd_free count
17925                              <1>                                  ; if the block is already deallocated
17926                              <1>                                  ; before.
17927 00005229 FF05[6A050300]      <1>         inc     dword [swpd_free]
17928                              <1> uswpbl_2:
17929 0000522F 5A                  <1>       pop    edx
17930 00005230 5B                  <1>       pop    ebx
17931 00005231 C3                  <1>       retn
17932                              <1>
17933                              <1> link_swap_block:
17934                              <1>       ; 01/07/2015
17935                              <1>       ; 18/04/2015
17936                              <1>       ; 24/10/2014 (Retro UNIX 386 v1 - beginning)
17937                              <1>       ;
17938                              <1>       ; INPUT -> none
17939                              <1>       ;
17940                              <1>       ; OUTPUT ->
17941                              <1>       ;     EAX = OFFSET ADDRESS OF THE ALLOCATED BLOCK (4096 bytes)
17942                              <1>       ;           in sectors (corresponding
17943                              <1>       ;           SWAP DISK ALLOCATION TABLE bit is RESET)
17944                              <1>       ;
17945                              <1>       ;     CF = 1 and EAX = 0
17946                              <1>       ;               if there is not a free block to be allocated
17947                              <1>       ;
17948                              <1>       ; Modified Registers -> none (except EAX)
17949                              <1>       ;
17950                              <1>
17951                              <1>       ;mov   eax, [swpd_free]
17952                              <1>       ;and   eax, eax
17953                              <1>       ;jz    short out_of_swpspc
17954                              <1>       ;
17955 00005232 53                  <1>       push   ebx
17956 00005233 51                  <1>       push   ecx
17957                              <1>       ;
17958 00005234 BB00000D00          <1>       mov    ebx, swap_alloc_table ; Swap Allocation Table offset
17959 00005239 89D9                <1>       mov    ecx, ebx
17960 0000523B 031D[6E050300]      <1>       add    ebx, [swpd_next] ; Free block searching starts from here
17961                              <1>                               ; next_free_swap_block >> 5
17962 00005241 030D[72050300]      <1>       add    ecx, [swpd_last] ; Free block searching ends here
17963                              <1>                               ; (total_swap_blocks - 1) >> 5
17964                              <1> lswbl_scan:
17965 00005247 39CB                <1>       cmp    ebx, ecx
17966 00005249 770A                <1>       ja     short lswbl_notfound
17967                              <1>       ;
17968 0000524B 0FBC03              <1>       bsf    eax, [ebx] ; Scans source operand for first bit set (1).
17969                              <1>                         ; Clears ZF if a bit is found set (1) and
17970                              <1>                         ; loads the destination with an index to
17971                              <1>                         ; first set bit. (0 -> 31)
17972                              <1>                         ; Sets ZF to 1 if no bits are found set.
17973                              <1>       ; 01/07/2015
17974 0000524E 751C                <1>       jnz    short lswbl_found ; ZF = 0 -> a free block has been found
17975                              <1>                          ;
17976                              <1>                          ; NOTE:  a Swap Disk Allocation Table bit
17977                              <1>                          ;       with value of 1 means
17978                              <1>                          ;       the corresponding page is free
17979                              <1>                          ;       (Retro UNIX 386 v1 feaure only!)
17980 00005250 83C304              <1>       add    ebx, 4
17981                              <1>                          ; We return back for searching next page block
17982                              <1>                          ; NOTE: [swpd_free] is not ZERO; so,
17983                              <1>                          ;     we always will find at least 1 free block here.
17984 00005253 EBF2                <1>       jmp           short lswbl_scan
17985                              <1>       ;
17986                              <1> lswbl_notfound:
17987 00005255 81E900000D00        <1>       sub    ecx, swap_alloc_table
17988 0000525B 890D[6E050300]      <1>       mov    [swpd_next], ecx ; next/first free page = last page
17989                              <1>                               ; (unlink_swap_block procedure will change it)
17990 00005261 31C0                <1>       xor    eax, eax
17991 00005263 A3[6A050300]        <1>       mov    [swpd_free], eax
17992 00005268 F9                  <1>       stc
17993                              <1> lswbl_ok:
17994 00005269 59                  <1>       pop    ecx
17995 0000526A 5B                  <1>       pop    ebx
17996 0000526B C3                  <1>       retn
17997                              <1>       ;
17998                              <1> ;out_of_swpspc:
17999                              <1> ;     stc
18000                              <1> ;     retn
18001                              <1>
18002                              <1> lswbl_found:
18003 0000526C 89D9                <1>       mov    ecx, ebx
18004 0000526E 81E900000D00        <1>       sub    ecx, swap_alloc_table
18005 00005274 890D[6E050300]      <1>       mov    [swpd_next], ecx ; Set first free block searching start
18006                              <1>                               ; address/offset (to the next)
18007 0000527A FF0D[6A050300]      <1>         dec     dword [swpd_free] ; 1 block has been allocated (X = X-1)
18008                              <1>       ;
18009 00005280 0FB303              <1>       btr    [ebx], eax     ; The destination bit indexed by the source value
18010                              <1>                               ; is copied into the Carry Flag and then cleared
18011                              <1>                               ; in the destination.
18012                              <1>                               ;
18013                              <1>                               ; Reset the bit which is corresponding to the
18014                              <1>                               ; (just) allocated block.
18015 00005283 C1E105              <1>       shl    ecx, 5         ; (block offset * 32) + block index
18016 00005286 01C8                <1>       add    eax, ecx       ; = block number
18017 00005288 C1E003              <1>       shl    eax, SECTOR_SHIFT ; 3, sector (offset) address of the block
18018                              <1>                               ; 1 block =  8 sectors
18019                              <1>       ;
18020                              <1>       ; EAX = offset address of swap disk/file sector (beginning of the block)
18021                              <1>       ;
18022                              <1>       ; NOTE: The relevant page table entry will be updated
```

```
18023                                  <1>       ;       according to this EAX value...
18024                                  <1>       ;
18025 0000528B EBDC                    <1>       jmp    short lswbl_ok
18026                                  <1>
18027                                  <1> logical_disk_read:
18028                                  <1>       ; 20/07/2015
18029                                  <1>       ; 09/03/2015 (temporary code here)
18030                                  <1>       ;
18031                                  <1>       ; INPUT ->
18032                                  <1>       ;     ESI = Logical disk description table address
18033                                  <1>       ;     EBX = Memory page (buffer) address (physical!)
18034                                  <1>       ;     EAX = Sector adress (offset address, logical sector number)
18035                                  <1>       ;     ECX = Sector count
18036                                  <1>       ;
18037                                  <1>       ;
18038 0000528D C3                      <1>       retn
18039                                  <1>
18040                                  <1> logical_disk_write:
18041                                  <1>       ; 20/07/2015
18042                                  <1>       ; 09/03/2015 (temporary code here)
18043                                  <1>       ;
18044                                  <1>       ; INPUT ->
18045                                  <1>       ;     ESI = Logical disk description table address
18046                                  <1>       ;     EBX = Memory page (buffer) address (physical!)
18047                                  <1>       ;     EAX = Sector adress (offset address, logical sector number)
18048                                  <1>       ;     ECX = Sector count
18049                                  <1>       ;
18050 0000528E C3                      <1>       retn
18051                                  <1>
18052                                  <1> get_physical_addr:
18053                                  <1>       ; 26/03/2017
18054                                  <1>       ; 20/02/2017
18055                                  <1>       ; 27/05/2016 - TRDOS 386 (TRDOS v2.0)
18056                                  <1>       ; 18/10/2015
18057                                  <1>       ; 29/07/2015
18058                                  <1>       ; 20/07/2015
18059                                  <1>       ; 04/06/2015
18060                                  <1>       ; 20/05/2015
18061                                  <1>       ; 28/04/2015
18062                                  <1>       ; 18/04/2015
18063                                  <1>       ; Get physical address
18064                                  <1>       ;     (allocates a new page for user if it is not present)
18065                                  <1>       ;
18066                                  <1>       ; (This subroutine is needed for mapping user's virtual
18067                                  <1>       ; (buffer) address to physical address (of the buffer).)
18068                                  <1>       ; ('sys write', 'sys read' system calls...)
18069                                  <1>       ;
18070                                  <1>       ; INPUT ->
18071                                  <1>       ;     EBX = virtual address
18072                                  <1>       ;     u.pgdir = page directory (physical) address
18073                                  <1>       ;
18074                                  <1>       ; OUTPUT ->
18075                                  <1>       ;     EAX = physical address
18076                                  <1>       ;     EBX = linear address
18077                                  <1>       ;     EDX = physical address of the page frame
18078                                  <1>       ;           (with attribute bits)
18079                                  <1>       ;     ECX = byte count within the page frame
18080                                  <1>       ;
18081                                  <1>       ; Modified Registers -> EAX, EBX, ECX, EDX
18082                                  <1>       ;
18083 0000528F 81C300004000            <1>       add    ebx, CORE ; 18/10/2015
18084                                  <1> get_physical_addr_x: ; 27/05/2016
18085 00005295 A1[B8030300]            <1>       mov    eax, [u.pgdir]
18086 0000529A E8F6F9FFFF              <1>       call   get_pte
18087                                  <1>              ; EDX = Page table entry address (if CF=0)
18088                                  <1>              ;       Page directory entry address (if CF=1)
18089                                  <1>              ;       (Bit 0 value is 0 if PT is not present)
18090                                  <1>              ; EAX = Page table entry value (page address)
18091                                  <1>              ;     CF = 1 -> PDE not present or invalid ?
18092 0000529F 731C                    <1>       jnc    short gpa_1
18093                                  <1>       ;
18094 000052A1 E8D4F8FFFF              <1>       call   allocate_page
18095 000052A6 7248                    <1>       jc     short gpa_im_err  ; 'insufficient memory' error
18096                                  <1> gpa_0:
18097 000052A8 E847F9FFFF              <1>       call   clear_page
18098                                  <1>       ; EAX = Physical (base) address of the allocated (new) page
18099 000052AD 0C07                    <1>       or     al, PDE_A_PRESENT + PDE_A_WRITE + PDE_A_USER ; 4+2+1 = 7
18100                                  <1>                        ; lower 3 bits are used as U/S, R/W, P flags
18101                                  <1>                        ; (user, writable, present page)
18102 000052AF 8902                    <1>       mov    [edx], eax ; Let's put the new page directory entry here !
18103 000052B1 A1[B8030300]            <1>       mov    eax, [u.pgdir]
18104 000052B6 E8DAF9FFFF              <1>       call   get_pte
18105 000052BB 7233                    <1>       jc     short gpa_im_err ; 'insufficient memory' error
18106                                  <1> gpa_1:
18107                                  <1>       ; EAX = PTE value, EDX = PTE address
18108 000052BD A801                    <1>       test   al, PTE_A_PRESENT
18109 000052BF 751F                    <1>       jnz    short gpa_3 ; 26/03/2017
18110 000052C1 09C0                    <1>       or     eax, eax
18111 000052C3 7456                    <1>       jz     short gpa_7  ; Allocate a new page
18112                                  <1>       ; 20/07/2015
18113 000052C5 55                      <1>       push   ebp
18114 000052C6 89DD                    <1>       mov    ebp, ebx ; virtual (linear) address
18115                                  <1>       ; reload swapped page
18116 000052C8 E878000000              <1>       call   reload_page ; 28/04/2015
18117 000052CD 5D                      <1>       pop    ebp
18118 000052CE 724A                    <1>       jc     short gpa_retn
18119                                  <1> gpa_2:
18120                                  <1>       ; 26/03/2017
18121                                  <1>       ; 20/02/2017
18122                                  <1>       ; If a page will contain a Signal Response Byte
18123                                  <1>       ; it must not be swapped out, because
18124                                  <1>       ; timer service or irq callback service
18125                                  <1>       ; will write a signal return/response byte
```

```
18126                                  <1>       ; directly by using physical address of Signal
18127                                  <1>       ; Response Byte.(Even if process is not running,
18128                                  <1>       ; or it is running with swapped out pages.)
18129                                  <1>       ;
18130                                  <1>       ; 'no_page_swap' will be set by 'systimer' or
18131                                  <1>       ; 'syscalbac' sistem functions/calls. (*)
18132                                  <1>       ;
18133 000052D0 803D[F6640100]00        <1>       cmp    byte [no_page_swap], 0
18134 000052D7 761D                    <1>       jna    short gpa_4 ; this page can be swapped out
18135                                  <1>       ; this page must not be swapped out
18136                                  <1>       ; but 'no_page_swap' must be reset here
18137                                  <1>       ; imediately for other callers (*)
18138                                  <1>       ; (otherwise, swap queue would not be long enough)
18139 000052D9 E84B000000              <1>       call   gpa_8 ; 26/03/2017
18140 000052DE EB1D                    <1>       jmp    short gpa_5
18141                                  <1> gpa_3:
18142                                  <1>       ; 26/03/2017
18143 000052E0 803D[F6640100]00        <1>       cmp    byte [no_page_swap], 0
18144 000052E7 7618                    <1>       jna    short gpa_6 ; this page can be swapped out
18145 000052E9 E83B000000              <1>       call   gpa_8
18146 000052EE EB11                    <1>       jmp    short gpa_6
18147                                  <1>
18148                                  <1> gpa_im_err:
18149 000052F0 B804000000              <1>       mov    eax, ERR_MINOR_IM ; Insufficient memory (minor) error!
18150                                  <1>                                ; Major error = 0 (No protection fault)
18151 000052F5 C3                      <1>       retn
18152                                  <1> gpa_4:
18153                                  <1>       ; 20/07/2015
18154                                  <1>       ; 20/05/2015
18155                                  <1>       ; add this page to swap queue
18156 000052F6 50                      <1>       push   eax
18157                                  <1>       ; EBX = Linear (CORE+virtual) address ; 20/02/2017
18158 000052F7 E8BDFEFFFF              <1>       call   add_to_swap_queue
18159 000052FC 58                      <1>       pop    eax
18160                                  <1> gpa_5:
18161                                  <1>             ; PTE address in EDX
18162                                  <1>             ; virtual address in EBX
18163                                  <1>       ; EAX = memory page address
18164 000052FD 0C07                    <1>       or     al, PTE_A_PRESENT + PTE_A_USER + PTE_A_WRITE
18165                                  <1>                               ; present flag, bit 0 = 1
18166                                  <1>                               ; user flag, bit 2 = 1
18167                                  <1>                               ; writable flag, bit 1 = 1
18168 000052FF 8902                    <1>       mov    [edx], eax  ; Update PTE value
18169                                  <1> gpa_6:
18170                                  <1>       ; 18/10/2015
18171 00005301 89D9                    <1>       mov    ecx, ebx
18172 00005303 81E1FF0F0000            <1>       and    ecx, PAGE_OFF
18173 00005309 89C2                    <1>       mov    edx, eax
18174 0000530B 662500F0                <1>       and    ax, PTE_A_CLEAR
18175 0000530F 01C8                    <1>       add    eax, ecx
18176 00005311 F7D9                    <1>       neg    ecx ; 1 -> -1 (0FFFFFFFFh), 4095 (0FFFh) -> -4095
18177 00005313 81C100100000            <1>       add    ecx, PAGE_SIZE
18178 00005319 F8                      <1>       clc
18179                                  <1> gpa_retn:
18180 0000531A C3                      <1>       retn
18181                                  <1> gpa_7:
18182 0000531B E85AF8FFFF              <1>       call   allocate_page
18183 00005320 72CE                    <1>       jc     short gpa_im_err ; 'insufficient memory' error
18184 00005322 E8CDF8FFFF              <1>       call   clear_page
18185 00005327 EBA7                    <1>       jmp    short gpa_2
18186                                  <1>
18187                                  <1> gpa_8: ; 26/03/2017
18188 00005329 C605[F6640100]00        <1>       mov    byte [no_page_swap], 0
18189 00005330 53                      <1>       push   ebx
18190 00005331 50                      <1>       push   eax ; 26/03/2017
18191 00005332 6681E300F0              <1>         and     bx, ~PAGE_OFF ; ~0FFFh ; reset bits, 0 to 11
18192 00005337 8A1D[B3030300]          <1>       mov    bl, [u.uno] ; current process number
18193 0000533D E89DFDFFFF              <1>       call   swap_queue_shift ; drop from the queue if
18194                                  <1>                               ; it is already on the queue
18195 00005342 58                      <1>       pop    eax ; 26/03/2017
18196 00005343 5B                      <1>       pop    ebx
18197 00005344 C3                      <1>       retn
18198                                  <1>
18199                                  <1> reload_page:
18200                                  <1>       ; 20/07/2015
18201                                  <1>       ; 28/04/2015 (Retro UNIX 386 v1 - beginning)
18202                                  <1>       ;
18203                                  <1>       ; Reload (Restore) swapped page at memory
18204                                  <1>       ;
18205                                  <1>       ; INPUT ->
18206                                  <1>       ;     EBP = Virtual (linear) memory address
18207                                  <1>       ;     EAX = PTE value (swap disk sector address)
18208                                  <1>       ;     (Swap disk sector address = bit 1 to bit 31 of EAX)
18209                                  <1>       ; OUTPUT ->
18210                                  <1>       ;     EAX = PHYSICAL (real/flat) ADDRESS OF RELOADED PAGE
18211                                  <1>       ;
18212                                  <1>       ;     CF = 1 and EAX = error code
18213                                  <1>       ;
18214                                  <1>       ; Modified Registers -> none (except EAX)
18215                                  <1>       ;
18216 00005345 D1E8                    <1>       shr    eax, 1   ; Convert PTE value to swap disk address
18217 00005347 53                      <1>       push   ebx      ;
18218 00005348 89C3                    <1>       mov    ebx, eax ; Swap disk (offset) address
18219 0000534A E82BF8FFFF              <1>       call   allocate_page
18220 0000534F 720C                    <1>       jc     short rlp_im_err
18221 00005351 93                      <1>       xchg   eax, ebx
18222                                  <1>       ; EBX = Physical memory (page) address
18223                                  <1>       ; EAX = Swap disk (offset) address
18224                                  <1>       ; EBP = Virtual (linear) memory address
18225 00005352 E862FCFFFF              <1>       call   swap_in
18226 00005357 720B                    <1>       jc     short rlp_swp_err  ; (swap disk/file read error)
18227 00005359 89D8                    <1>       mov    eax, ebx
18228                                  <1> rlp_retn:
```

```
18229 0000535B 5B                <1>       pop    ebx
18230 0000535C C3                <1>       retn
18231                            <1>
18232                            <1> rlp_im_err:
18233 0000535D B804000000        <1>       mov    eax, ERR_MINOR_IM ; Insufficient memory (minor) error!
18234                            <1>                                ; Major error = 0 (No protection fault)
18235 00005362 EBF7              <1>       jmp    short rlp_retn
18236                            <1>
18237                            <1> rlp_swp_err:
18238 00005364 B828000000        <1>       mov    eax, SWP_DISK_READ_ERR ; Swap disk read error !
18239 00005369 EBF0              <1>       jmp    short rlp_retn
18240                            <1>
18241                            <1>
18242                            <1> copy_page_dir:
18243                            <1>       ; 19/09/2015
18244                            <1>       ; temporary - 07/09/2015
18245                            <1>       ; 07/09/2015 (Retro UNIX 386 v1 - beginning)
18246                            <1>       ;
18247                            <1>       ; INPUT ->
18248                            <1>       ;      [u.pgdir] = PHYSICAL (real/flat) ADDRESS of the parent's
18249                            <1>       ;                  page directory.
18250                            <1>       ; OUTPUT ->
18251                            <1>       ;      EAX =  PHYSICAL (real/flat) ADDRESS of the child's
18252                            <1>       ;             page directory.
18253                            <1>       ;      (New page directory with new page table entries.)
18254                            <1>       ;      (New page tables with read only copies of the parent's
18255                            <1>       ;      pages.)
18256                            <1>       ;      EAX = 0 -> Error (CF = 1)
18257                            <1>       ;
18258                            <1>       ; Modified Registers -> none (except EAX)
18259                            <1>       ;
18260 0000536B E80AF8FFFF        <1>       call   allocate_page
18261 00005370 723E              <1>       jc     short cpd_err
18262                            <1>       ;
18263 00005372 55                <1>       push   ebp ; 20/07/2015
18264 00005373 56                <1>       push   esi
18265 00005374 57                <1>       push   edi
18266 00005375 53                <1>       push   ebx
18267 00005376 51                <1>       push   ecx
18268 00005377 8B35[B8030300]    <1>       mov    esi, [u.pgdir]
18269 0000537D 89C7              <1>       mov    edi, eax
18270 0000537F 50                <1>       push   eax ; save child's page directory address
18271                            <1>       ; copy PDE 0 from the parent's page dir to the child's page dir
18272                            <1>       ; (use same system space for all user page tables)
18273 00005380 A5                <1>       movsd
18274 00005381 BD00004000        <1>       mov    ebp, 1024*4096 ; pass the 1st 4MB (system space)
18275 00005386 B9FF030000        <1>       mov    ecx, (PAGE_SIZE / 4) - 1 ; 1023
18276                            <1> cpd_0:
18277 0000538B AD                <1>       lodsd
18278                            <1>       ;or    eax, eax
18279                            <1>         ;jnz     short cpd_1
18280 0000538C A801              <1>       test   al, PDE_A_PRESENT ;  bit 0 =  1
18281 0000538E 7508              <1>       jnz    short cpd_1
18282                            <1>       ; (virtual address at the end of the page table)
18283 00005390 81C500004000      <1>       add    ebp, 1024*4096 ; page size * PTE count
18284 00005396 EB0F              <1>       jmp    short cpd_2
18285                            <1> cpd_1:
18286 00005398 662500F0          <1>       and    ax, PDE_A_CLEAR ; 0F000h ; clear attribute bits
18287 0000539C 89C3              <1>       mov    ebx, eax
18288                            <1>       ; EBX = Parent's page table address
18289 0000539E E81F000000        <1>       call   copy_page_table
18290 000053A3 720C              <1>       jc     short cpd_p_err
18291                            <1>       ; EAX = Child's page table address
18292 000053A5 0C07              <1>       or     al, PDE_A_PRESENT + PDE_A_WRITE + PDE_A_USER
18293                            <1>                      ; set bit 0, bit 1 and bit 2 to 1
18294                            <1>                      ; (present, writable, user)
18295                            <1> cpd_2:
18296 000053A7 AB                <1>       stosd
18297 000053A8 E2E1              <1>       loop   cpd_0
18298                            <1>       ;
18299 000053AA 58                <1>       pop    eax  ; restore child's page directory address
18300                            <1> cpd_3:
18301 000053AB 59                <1>       pop    ecx
18302 000053AC 5B                <1>       pop    ebx
18303 000053AD 5F                <1>       pop    edi
18304 000053AE 5E                <1>       pop    esi
18305 000053AF 5D                <1>       pop    ebp
18306                            <1> cpd_err:
18307 000053B0 C3                <1>       retn
18308                            <1> cpd_p_err:
18309                            <1>       ; release the allocated pages missing (recover free space)
18310 000053B1 58                <1>       pop    eax  ; the new page directory address (physical)
18311 000053B2 8B1D[B8030300]    <1>       mov    ebx, [u.pgdir] ; parent's page directory address
18312 000053B8 E8F6F8FFFF        <1>       call   deallocate_page_dir
18313 000053BD 29C0              <1>       sub    eax, eax ; 0
18314 000053BF F9                <1>       stc
18315 000053C0 EBE9              <1>       jmp    short cpd_3
18316                            <1>
18317                            <1> copy_page_table:
18318                            <1>       ; 19/09/2015
18319                            <1>       ; temporary - 07/09/2015
18320                            <1>       ; 07/09/2015 (Retro UNIX 386 v1 - beginning)
18321                            <1>       ;
18322                            <1>       ; INPUT ->
18323                            <1>       ;      EBX = PHYSICAL (real/flat) ADDRESS of the parent's page table.
18324                            <1>       ;      EBP = page table entry index (from 'copy_page_dir')
18325                            <1>       ; OUTPUT ->
18326                            <1>       ;      EAX = PHYSICAL (real/flat) ADDRESS of the child's page table.
18327                            <1>       ;      EBP = (recent) page table index (for 'add_to_swap_queue')
18328                            <1>       ;      CF = 1 -> error
18329                            <1>       ;
18330                            <1>       ; Modified Registers -> EBP (except EAX)
18331                            <1>       ;
```

```
18332 000053C2 E8B3F7FFFF          <1>       call   allocate_page
18333 000053C7 725A                <1>       jc     short cpt_err
18334                              <1>       ;
18335 000053C9 50                  <1>       push   eax ; *
18336                              <1>       ;push  ebx
18337 000053CA 56                  <1>       push   esi
18338 000053CB 57                  <1>       push   edi
18339 000053CC 52                  <1>       push   edx
18340 000053CD 51                  <1>       push   ecx
18341                              <1>       ;
18342 000053CE 89DE                <1>       mov    esi, ebx
18343 000053D0 89C7                <1>       mov    edi, eax
18344 000053D2 89C2                <1>       mov    edx, eax
18345 000053D4 81C200100000        <1>       add    edx, PAGE_SIZE
18346                              <1> cpt_0:
18347 000053DA AD                  <1>       lodsd
18348 000053DB A801                <1>       test   al, PTE_A_PRESENT ;  bit 0 =  1
18349 000053DD 750B                <1>       jnz    short cpt_1
18350 000053DF 21C0                <1>       and    eax, eax
18351 000053E1 7430                <1>       jz     short cpt_2
18352                              <1>       ; ebp = virtual (linear) address of the memory page
18353 000053E3 E85DFFFFFF          <1>       call   reload_page ; 28/04/2015
18354 000053E8 7234                <1>       jc     short cpt_p_err
18355                              <1> cpt_1:
18356 000053EA 662500F0            <1>       and    ax, PTE_A_CLEAR ; 0F000h ; clear attribute bits
18357 000053EE 89C1                <1>       mov    ecx, eax
18358                              <1>       ; Allocate a new page for the child process
18359 000053F0 E885F7FFFF          <1>       call   allocate_page
18360 000053F5 7227                <1>       jc     short cpt_p_err
18361 000053F7 57                  <1>       push   edi
18362 000053F8 56                  <1>       push   esi
18363 000053F9 89CE                <1>       mov    esi, ecx
18364 000053FB 89C7                <1>       mov    edi, eax
18365 000053FD B900040000          <1>       mov    ecx, PAGE_SIZE/4
18366 00005402 F3A5                <1>       rep    movsd  ; copy page (4096 bytes)
18367 00005404 5E                  <1>       pop    esi
18368 00005405 5F                  <1>       pop    edi
18369                              <1>       ;
18370 00005406 53                  <1>       push   ebx
18371 00005407 50                  <1>       push   eax
18372 00005408 89EB                <1>       mov    ebx, ebp
18373                              <1>       ; ebx = virtual address of the memory page
18374 0000540A E8AAFDFFFF          <1>       call   add_to_swap_queue
18375 0000540F 58                  <1>       pop    eax
18376 00005410 5B                  <1>       pop    ebx
18377                              <1>       ;
18378                              <1>       ;or    ax, PTE_A_USER+PTE_A_PRESENT
18379 00005411 0C07                <1>       or     al, PTE_A_USER+PTE_A_WRITE+PTE_A_PRESENT
18380                              <1> cpt_2:
18381 00005413 AB                  <1>       stosd  ; EDI points to child's PTE
18382                              <1>       ;
18383 00005414 81C500100000        <1>       add    ebp, 4096 ; 20/07/2015 (next page)
18384                              <1>       ;
18385 0000541A 39D7                <1>       cmp    edi, edx
18386 0000541C 72BC                <1>       jb     short cpt_0
18387                              <1> cpt_p_err:
18388 0000541E 59                  <1>       pop    ecx
18389 0000541F 5A                  <1>       pop    edx
18390 00005420 5F                  <1>       pop    edi
18391 00005421 5E                  <1>       pop    esi
18392                              <1>       ;pop   ebx
18393 00005422 58                  <1>       pop    eax ; *
18394                              <1> cpt_err:
18395 00005423 C3                  <1>       retn
18396                              <1>
18397                              <1> allocate_memory_block:
18398                              <1>       ; 01/05/2017
18399                              <1>       ; 28/04/2017
18400                              <1>       ; 25/04/2017
18401                              <1>       ; 01/04/2016, 02/04/2016, 03/04/2016
18402                              <1>       ; 13/03/2016, 14/03/2016
18403                              <1>       ; 12/03/2016 (TRDOS 386 = TRDOS v2.0)
18404                              <1>       ; Allocating contiguous memory pages (in the kernel's memory space)
18405                              <1>       ;
18406                              <1>       ; INPUT ->
18407                              <1>       ;     EAX = Beginning address (physical)
18408                              <1>       ;     EAX = 0 -> Allocate memory block from the first proper aperture
18409                              <1>       ;     ECX = Number of bytes to be allocated
18410                              <1>       ;
18411                              <1>       ; OUTPUT ->
18412                              <1>       ;     1) cf = 0 -> successful
18413                              <1>       ;     EAX = Beginning (physical) address of the allocated memory block
18414                              <1>       ;     ECX = Number of allocated bytes (rounded up to page borders)
18415                              <1>       ;     2) cf = 1 -> unsuccessful
18416                              <1>       ;      2.1) If EAX > 0 ->
18417                              <1>       ;           (Number of requested pages is more than # of free pages
18418                              <1>       ;            but contiguous free pages -the aperture- is not enough!)
18419                              <1>       ;           EAX = Beginning address of available aperture
18420                              <1>       ;                (one of all aperture with max. aperture size/length)
18421                              <1>       ;           ECX = Size of available aperture (memory block) in bytes
18422                              <1>       ;      2.2) If EAX = 0 -> Out of memory error
18423                              <1>       ;                  (number of free pages is less than requested number)
18424                              <1>       ;           ECX = Total number of free bytes (free pages * 4096)
18425                              <1>       ;                (It is not number of contiguous free bytes)
18426                              <1>       ;
18427                              <1>       ; (Modified Registers -> EAX, ECX)
18428                              <1>       ;
18429                              <1>       ; PURPOSE: Loading a file at memory for copying or running etc.
18430                              <1>       ; If this procedure returns with cf is set, ECX contains maximum
18431                              <1>       ; available space and EAX contains the beginning address of it.
18432                              <1>       ; If EAX has zero, ECX contains total number of free bytes.
18433                              <1>       ; If requested block has been successfully allocated (by rounding up to
18434                              <1>       ; the last page border), it must be deallocated later by using
```

```
18435                                  <1>      ; 'deallocate_memory_block' procedure.
18436                                  <1>
18437 00005424 52                      <1>      push   edx ; *
18438 00005425 BAFF0F0000              <1>      mov    edx, PAGE_SIZE - 1   ; 4095
18439 0000542A 01D0                    <1>      add    eax, edx
18440 0000542C 01D1                    <1>      add    ecx, edx
18441 0000542E C1E90C                  <1>      shr    ecx, PAGE_SHIFT         ; 12
18442                                  <1>
18443                                  <1>      ; ECX = number of contiguous pages to be allocated
18444 00005431 8B15[28520100]          <1>      mov    edx, [free_pages]
18445                                  <1>      ; 01/05/2017
18446                                  <1>      ;or    ecx, ecx
18447                                  <1>      ;jz    short amb3
18448                                  <1>      ; If ECX=0, set cf to 1 and return with max. available mem block size
18449                                  <1>
18450 00005437 39D1                    <1>      cmp    ecx, edx
18451 00005439 7760                    <1>      ja     short amb_3
18452                                  <1>
18453 0000543B C1E80C                  <1>      shr    eax, PAGE_SHIFT      ; 12
18454                                  <1>
18455 0000543E 89C2                    <1>      mov    edx, eax          ; page number
18456 00005440 C1EA03                  <1>      shr    edx, 3            ; to get offset to M.A.T.
18457                                  <1>                               ; (1 allocation bit = 1 page)
18458                                  <1>                               ; (1 allocation bytes = 8 pages)
18459 00005443 80E2FC                  <1>      and    dl, 0FCh          ; clear lower 2 bits
18460                                  <1>                               ; (to get 32 bit position)
18461 00005446 53                      <1>      push   ebx ; **
18462                                  <1> amb_0:
18463 00005447 890D[E05E0100]          <1>      mov    [mem_ipg_count], ecx ; initial (reset) value of page count
18464 0000544D 890D[E45E0100]          <1>      mov    [mem_pg_count], ecx
18465 00005453 31C9                    <1>      xor    ecx, ecx ; 0
18466 00005455 890D[E85E0100]          <1>      mov    [mem_aperture], ecx ; 0
18467 0000545B 890D[EC5E0100]          <1>      mov    [mem_max_aperture], ecx ; 0
18468                                  <1>
18469 00005461 BB00001000              <1>      mov    ebx, MEM_ALLOC_TBL   ; Memory Allocation Table address.
18470 00005466 3B15[2C520100]          <1>      cmp    edx, [next_page]     ; Is the beginning page address lower
18471                                  <1>                                 ; than the address in 'next_page' ?
18472                                  <1>                                 ; (the first/next free page of user space)
18473 0000546C 7208                    <1>      jb     short amb_1
18474 0000546E 3B15[30520100]          <1>      cmp    edx, [last_page]     ; is the beginning page address higher
18475                                  <1>                                 ; than the address in 'last_page' ?
18476                                  <1>                                 ; (end of the memory)
18477 00005474 7606                    <1>      jna    short amb_2       ; no
18478                                  <1> amb_1:
18479 00005476 8B15[2C520100]          <1>      mov    edx, [next_page]     ; M.A.T. offset (1 M.A.T. byte = 8 pages)
18480                                  <1> amb_2:
18481 0000547C 01D3                    <1>      add    ebx, edx
18482                                  <1>
18483                                  <1>      ; 28/04/2017
18484                                  <1>      ;xor   ecx, ecx
18485 0000547E 0FBC0B                  <1>      bsf    ecx, [ebx]        ; 0 to 31
18486 00005481 89D0                    <1>      mov    eax, edx
18487 00005483 C1E003                  <1>      shl    eax, 3            ; *8
18488 00005486 01C8                    <1>      add    eax, ecx          ; beginning page number
18489                                  <1>
18490 00005488 A3[F05E0100]            <1>      mov    [mem_pg_pos], eax    ; beginning page no (for curr. mem. aperture)
18491 0000548D A3[F45E0100]            <1>      mov    [mem_max_pg_pos], eax ; beginning page no for max. mem. aperture
18492                                  <1>
18493 00005492 83E01F                  <1>      and    eax, 1Fh          ; lower 5 bits only (0 to 31)
18494                                  <1>                               ; (allocation bit position)
18495 00005495 750E                    <1>      jnz    short amb_4       ; 0
18496 00005497 B120                    <1>      mov    cl, 32
18497 00005499 EB4B                    <1>      jmp    short amb_10
18498                                  <1>
18499                                  <1> amb_3:      ; out_of_memory
18500 0000549B 31C0                    <1>      xor    eax, eax ; 0
18501 0000549D 89D1                    <1>      mov    ecx, edx ; free pages
18502 0000549F C1E10C                  <1>      shl    ecx, PAGE_SHIFT
18503 000054A2 5A                      <1>      pop    edx ; *
18504 000054A3 F9                      <1>      stc
18505 000054A4 C3                      <1>      retn
18506                                  <1> amb_4:
18507 000054A5 8B13                    <1>      mov    edx, [ebx]
18508 000054A7 88C1                    <1>      mov    cl, al ; 1 to 31
18509 000054A9 D3EA                    <1>      shr    edx, cl
18510 000054AB 89D0                    <1>      mov    eax, edx
18511                                  <1> amb_5:
18512 000054AD D1E8                    <1>      shr    eax, 1 ; (***)
18513 000054AF 7317                    <1>      jnc    short amb_7
18514 000054B1 FF05[E85E0100]          <1>      inc    dword [mem_aperture]
18515 000054B7 FF0D[E45E0100]          <1>      dec    dword [mem_pg_count]
18516 000054BD 7470                    <1>      jz     short amb_15
18517                                  <1> amb_6:
18518                                  <1>      ; 28/04/2017
18519 000054BF FEC1                    <1>      inc    cl
18520 000054C1 80F920                  <1>      cmp    cl, 32
18521 000054C4 730D                    <1>      jnb    short amb_9
18522 000054C6 EBE5                    <1>      jmp    short amb_5
18523                                  <1> amb_7:
18524 000054C8 50                      <1>      push   eax ; (***) allocation bits (in shifted status)
18525 000054C9 E81B010000              <1>      call   amb_26 ; set maximum memory aperture (free memory block size)
18526 000054CE 58                      <1>      pop    eax ; (***)
18527 000054CF EBEE                    <1>      jmp    short amb_6
18528                                  <1> amb_8:
18529                                  <1>      ; 28/04/2017
18530 000054D1 B120                    <1>      mov    cl, 32
18531                                  <1> amb_9:
18532 000054D3 89DA                    <1>      mov    edx, ebx
18533 000054D5 81EA00001000            <1>      sub    edx, MEM_ALLOC_TBL
18534 000054DB 3B15[30520100]          <1>      cmp    edx, [last_page]
18535 000054E1 7336                    <1>      jnb    short amb_14 ; contiguous pages not enough
18536 000054E3 83C304                  <1>      add    ebx, 4
18537                                  <1> amb_10:
```

```
18538 000054E6 8B03              <1>       mov    eax, [ebx]
18539 000054E8 21C0              <1>       and    eax, eax
18540 000054EA 7408              <1>         jz      short amb_11 ; there is not a free page bit in this alloc dword
18541 000054EC 40                <1>       inc    eax ; 0FFFFFFFFh -> 0
18542 000054ED 740C              <1>       jz     short amb_12 ; all of bits are set (32 free pages)
18543 000054EF 48                <1>       dec    eax
18544 000054F0 28C9              <1>       sub    cl, cl ; 0
18545 000054F2 EBB9              <1>       jmp    short amb_5
18546                            <1> amb_11:
18547 000054F4 E8F0000000        <1>       call   amb_26 ; set maximum memory aperture (free memory block size)
18548 000054F9 EBD8              <1>       jmp    short amb_9
18549                            <1> amb_12:
18550 000054FB 390D[E45E0100]    <1>       cmp    [mem_pg_count], ecx ; 32
18551 00005501 7306              <1>       jnb    short amb_13
18552 00005503 8B0D[E45E0100]    <1>       mov    ecx, [mem_pg_count]
18553                            <1> amb_13:
18554 00005509 010D[E85E0100]    <1>       add    [mem_aperture], ecx
18555 0000550F 290D[E45E0100]    <1>       sub    [mem_pg_count], ecx
18556 00005515 7618              <1>       jna    short amb_15
18557 00005517 EBBA              <1>       jmp    short amb_9 ; 01/05/2017
18558                            <1> amb_14:
18559 00005519 E8CB000000        <1>       call   amb_26 ; 28/04/2017
18560 0000551E A1[F45E0100]      <1>       mov    eax, [mem_max_pg_pos] ; begin address of max. mem aperture
18561 00005523 8B0D[EC5E0100]    <1>       mov    ecx, [mem_max_aperture] ; max. (largest) memory aperture
18562 00005529 F9                <1>       stc
18563 0000552A E9AF000000        <1>         jmp     amb_25
18564                            <1>
18565                            <1> amb_15: ; OK !
18566 0000552F A1[F05E0100]      <1>       mov    eax, [mem_pg_pos]    ; Beginning address as page number
18567 00005534 8B0D[E85E0100]    <1>       mov    ecx, [mem_aperture]  ; Free contiguous page count (>=1)
18568                            <1> amb_16:
18569                            <1>       ; allocate contiguous memory pages (via memory allocation table bits)
18570 0000553A 89C2              <1>       mov    edx, eax
18571                            <1>       ; 25/04/2017
18572 0000553C C1EA03            <1>       shr    edx, 3        ; 8 pages in one allocation byte
18573 0000553F 80E2FC            <1>       and    dl, 0FCh      ; clear lower 2 bits
18574                            <1>                            ; (for dword/32bit positioning)
18575                            <1>
18576 00005542 BB00001000        <1>       mov    ebx, MEM_ALLOC_TBL
18577 00005547 01D3              <1>       add    ebx, edx
18578 00005549 83E01F            <1>       and    eax, 1Fh ; 31
18579                            <1>       ; 03/04/2016
18580 0000554C BA20000000        <1>       mov    edx, 32
18581 00005551 28C2              <1>       sub    dl, al
18582 00005553 39CA              <1>       cmp    edx, ecx      ; ecx >= 1
18583 00005555 7602              <1>       jna    short amb_17
18584 00005557 89CA              <1>       mov    edx, ecx
18585                            <1> amb_17:
18586 00005559 29D1              <1>       sub    ecx, edx
18587 0000555B 51                <1>       push   ecx ; ***
18588 0000555C 89D1              <1>       mov    ecx, edx
18589                            <1> amb_18:
18590 0000555E 0FB303            <1>       btr    [ebx], eax    ; The destination bit indexed by the source value
18591                            <1>                            ; is copied into the Carry Flag and then cleared
18592                            <1>                            ; in the destination.
18593 00005561 FF0D[28520100]    <1>       dec     dword [free_pages] ; 1 page has been allocated (X = X-1)
18594 00005567 49                <1>       dec    ecx
18595 00005568 7404              <1>       jz     short amb_19
18596 0000556A FEC0              <1>       inc    al
18597 0000556C EBF0              <1>       jmp    short amb_18
18598                            <1> amb_19:
18599 0000556E 59                <1>       pop    ecx ; ***
18600 0000556F 21C9              <1>       and    ecx, ecx ; 0 ?
18601 00005571 741E              <1>       jz     short amb_22
18602                            <1>       ; 01/04/2016
18603 00005573 B020              <1>       mov    al, 32
18604                            <1> amb_20:
18605 00005575 83C304            <1>       add    ebx, 4
18606 00005578 39C1              <1>       cmp    ecx, eax ; 32
18607 0000557A 7305              <1>       jnb    short amb_21
18608                            <1>       ; ECX < 32
18609 0000557C 28C0              <1>       sub    al, al ; 0
18610 0000557E 50                <1>       push   eax ; 0 ***
18611 0000557F EBDD              <1>       jmp    short amb_18
18612                            <1> amb_21:
18613 00005581 2905[28520100]    <1>       sub    [free_pages], eax   ; [free_pages] = [free_pages] - 32
18614 00005587 C70300000000      <1>       mov    dword [ebx], 0         ; reset 32 bits
18615 0000558D 29C1              <1>       sub    ecx, eax ; 32
18616 0000558F 75E4              <1>       jnz    short amb_20
18617                            <1> amb_22:
18618 00005591 A1[F05E0100]      <1>       mov    eax, [mem_pg_pos]   ; Beginning address as page number
18619 00005596 8B0D[E85E0100]    <1>       mov    ecx, [mem_aperture] ; Free contiguous page count
18620                            <1>       ; [next_page] update
18621 0000559C 89C2              <1>       mov    edx, eax
18622                            <1>       ; 03/04/2016
18623 0000559E C1EA03            <1>       shr    edx, 3            ; to get offset to M.A.T.
18624                            <1>                                ; (1 allocation bit = 1 page)
18625                            <1>                                ; (1 allocation bytes = 8 pages)
18626 000055A1 80E2FC            <1>       and    dl, 0FCh          ; clear lower 2 bits
18627                            <1>                                ; (to get 32 bit position)
18628 000055A4 3B15[2C520100]    <1>       cmp    edx, [next_page] ; first free page pointer offset
18629 000055AA 7732              <1>       ja     short amb_25
18630 000055AC BB00001000        <1>       mov    ebx, MEM_ALLOC_TBL
18631 000055B1 833C1300          <1>       cmp    dword [ebx+edx], 0
18632 000055B5 7721              <1>       ja     short amb_24
18633 000055B7 89C2              <1>       mov    edx, eax
18634 000055B9 01CA              <1>       add    edx, ecx
18635 000055BB C1EA03            <1>       shr    edx, 3
18636 000055BE 80E2FC            <1>       and    dl, 0FCh
18637                            <1> amb_23:
18638 000055C1 833C1300          <1>       cmp    dword [ebx+edx], 0
18639 000055C5 7711              <1>       ja     short amb_24
18640 000055C7 83C204            <1>       add    edx, 4
```

```
18641 000055CA 3B15[30520100]      <1>       cmp    edx, [last_page]    ; last page pointer offset
18642 000055D0 76EF                <1>       jna    short amb_23
18643 000055D2 8B15[34520100]      <1>       mov    edx, [first_page]   ; (for) beginning of user's space
18644                              <1> amb_24:
18645 000055D8 8915[2C520100]      <1>       mov    [next_page], edx
18646                              <1> amb_25:
18647 000055DE 9C                  <1>       pushf
18648 000055DF C1E00C              <1>       shl    eax, PAGE_SHIFT         ; convert to phy. address in bytes
18649 000055E2 C1E10C              <1>       shl    ecx, PAGE_SHIFT         ; convert to byte counts
18650 000055E5 9D                  <1>       popf
18651 000055E6 5B                  <1>       pop    ebx ; **
18652 000055E7 5A                  <1>       pop    edx ; *
18653 000055E8 C3                  <1>       retn
18654                              <1>
18655                              <1> amb_26:     ; set maximum free memory aperture (free memory block size)
18656 000055E9 89DA                <1>       mov    edx, ebx ; current address
18657 000055EB 81EA00001000        <1>       sub    edx, MEM_ALLOC_TBL ; MAT beginning address
18658                              <1>       ; 02/04/2016
18659 000055F1 C1E203              <1>       shl    edx, 3 ; MAT byte offset * 8 = page number base
18660 000055F4 01CA                <1>       add    edx, ecx ; current page number (ecx = 0 to 32)
18661                              <1>       ;
18662 000055F6 A1[E85E0100]        <1>       mov    eax, [mem_aperture]
18663 000055FB 21C0                <1>       and    eax, eax
18664 000055FD 7421                <1>         jz     short amb_27
18665 000055FF C705[E85E0100]0000- <1>         mov    dword [mem_aperture], 0
18666 00005607 0000                <1>
18667 00005609 3B05[EC5E0100]      <1>       cmp    eax, [mem_max_aperture]
18668 0000560F 760F                <1>       jna    short amb_27
18669 00005611 A3[EC5E0100]        <1>       mov    [mem_max_aperture], eax
18670                              <1>       ; 25/04/2017
18671 00005616 A1[F05E0100]        <1>       mov    eax, [mem_pg_pos]
18672                              <1>       ; EAX = Beginning page number of the max. aperture
18673 0000561B A3[F45E0100]        <1>       mov    [mem_max_pg_pos], eax
18674                              <1> amb_27:
18675 00005620 8915[F05E0100]      <1>       mov    [mem_pg_pos], edx ; current page
18676                              <1>
18677 00005626 A1[E05E0100]        <1>       mov    eax, [mem_ipg_count] ; initial (reset) value of page count
18678 0000562B A3[E45E0100]        <1>       mov    [mem_pg_count], eax
18679                              <1>
18680 00005630 C3                  <1>       retn
18681                              <1>
18682                              <1> deallocate_memory_block:
18683                              <1>       ; 03/04/2016
18684                              <1>       ; 14/03/2016 (TRDOS 386 = TRDOS v2.0)
18685                              <1>       ; Deallocating contiguous memory pages (in the kernel's memory space)
18686                              <1>       ;
18687                              <1>       ; INPUT ->
18688                              <1>       ;     EAX = Beginning address (physical)
18689                              <1>       ;     ECX = Number of bytes to be deallocated
18690                              <1>       ;
18691                              <1>       ; OUTPUT ->
18692                              <1>       ;     Memory Allocation Table bits will be updated
18693                              <1>       ;     [free_pages] will be changed (increased)
18694                              <1>       ;
18695                              <1>       ; (Modified Registers -> EAX, ECX)
18696                              <1>       ;
18697                              <1>       ; PURPOSE: Unloading/Freeing a file -or an allocated memory block-
18698                              <1>       ; at memory after copying, running, saving, reading, writing etc.
18699                              <1>       ;
18700                              <1>
18701 00005631 52                  <1>       push   edx ; *
18702 00005632 53                  <1>       push   ebx ; **
18703                              <1>
18704 00005633 C1E80C              <1>       shr    eax, PAGE_SHIFT         ; 12
18705 00005636 C1E90C              <1>       shr    ecx, PAGE_SHIFT         ; 12
18706                              <1>
18707                              <1>       ; EAX = Beginning page number
18708                              <1>       ; ECX = Number of contiguous pages to be deallocated
18709                              <1> damb_0:
18710                              <1>       ; deallocate contiguous memory pages (via memory allocation table bits)
18711 00005639 89C2                <1>       mov    edx, eax
18712 0000563B C1EA03              <1>       shr    edx, 3           ; to get offset to M.A.T.
18713                              <1>                               ; (1 allocation bit = 1 page)
18714                              <1>                               ; (1 allocation bytes = 8 pages)
18715 0000563E 80E2FC              <1>       and    dl, 0FCh         ; clear lower 2 bits
18716                              <1>                               ; (to get 32 bit position)
18717 00005641 3B15[2C520100]      <1>       cmp    edx, [next_page] ; next free page
18718 00005647 7306                <1>       jnb    short damb_1
18719 00005649 8915[2C520100]      <1>       mov    [next_page], edx
18720                              <1> damb_1:
18721 0000564F BB00001000          <1>       mov    ebx, MEM_ALLOC_TBL
18722 00005654 01D3                <1>       add    ebx, edx
18723 00005656 83E01F              <1>       and    eax, 1Fh ; 31
18724                              <1>
18725                              <1>       ; 03/04/2016
18726 00005659 BA20000000          <1>       mov    edx, 32
18727 0000565E 28C2                <1>       sub    dl, al
18728 00005660 39CA                <1>       cmp    edx, ecx
18729 00005662 7602                <1>       jna    short damb_2
18730 00005664 89CA                <1>       mov    edx, ecx
18731                              <1> damb_2:
18732 00005666 29D1                <1>       sub    ecx, edx
18733 00005668 51                  <1>       push   ecx ; ***
18734 00005669 89D1                <1>       mov    ecx, edx
18735                              <1> damb_3:
18736 0000566B 0FAB03              <1>       bts    [ebx], eax       ; unlink/release/deallocate page
18737                              <1>                               ; set relevant bit to 1.
18738                              <1>                               ; set CF to the previous bit value
18739 0000566E FF05[28520100]      <1>       inc     dword [free_pages]   ; 1 page has been deallocated (X = X+1)
18740 00005674 49                  <1>       dec    ecx
18741 00005675 7404                <1>       jz     short damb_4
18742 00005677 FEC0                <1>       inc    al
18743 00005679 EBF0                <1>       jmp    short damb_3
```

```
18744                                  <1> damb_4:
18745 0000567B 59                      <1>         pop    ecx ; ***
18746 0000567C 21C9                    <1>         and    ecx, ecx ; 0 ?
18747 0000567E 741E                    <1>         jz     short damb_7
18748                                  <1>         ; 03/04/2016
18749 00005680 B020                    <1>         mov    al, 32
18750                                  <1> damb_5:
18751 00005682 83C304                  <1>         add    ebx, 4
18752 00005685 39C1                    <1>         cmp    ecx, eax ; 32
18753 00005687 7305                    <1>         jnb    short damb_6
18754                                  <1>         ; ECX < 32
18755 00005689 28C0                    <1>         sub    al, al ; 0
18756 0000568B 50                      <1>         push   eax ; 0 ***
18757 0000568C EBDD                    <1>         jmp    short damb_3
18758                                  <1> damb_6:
18759 0000568E 0105[28520100]          <1>         add    [free_pages], eax ; [free_pages] = [free_pages] + 32
18760 00005694 C703FFFFFFFF            <1>         mov    dword [ebx], 0FFFFFFFFh ; set 32 bits
18761 0000569A 29C1                    <1>         sub    ecx, eax ; 32
18762 0000569C 75E4                    <1>         jnz    short damb_5
18763                                  <1> damb_7:
18764 0000569E 5B                      <1>         pop    ebx ; **
18765 0000569F 5A                      <1>         pop    edx ; *
18766 000056A0 C3                      <1>         retn
18767                                  <1>
18768                                  <1> direct_memory_access:
18769                                  <1>         ; 22/07/2017
18770                                  <1>         ; 12/05/2017
18771                                  <1>         ; 16/07/2016
18772                                  <1>         ; 12/07/2016 (TRDOS 386 = TRDOS v2.0)
18773                                  <1>         ; This processure will be called to map
18774                                  <1>         ; user's (ring 3) page tables to access phsical
18775                                  <1>         ; (flat/linear) memory addresses, directly (without
18776                                  <1>         ;  kernel's data transfer functions).
18777                                  <1>         ;
18778                                  <1>         ; Purpose: Video memory access and shared memory access.
18779                                  <1>         ;
18780                                  <1>         ; INPUT ->
18781                                  <1>         ;     EAX = Beginning address (physical).
18782                                  <1>         ;     EBX = User's buffer address ; 12/05/2017
18783                                  <1>         ;     ECX = Number of contiguous pages to be mapped.
18784                                  <1>         ; OUTPUT ->
18785                                  <1>         ;     User's page directory and pages tables
18786                                  <1>         ;     will be updated.
18787                                  <1>         ;
18788                                  <1>         ;     If an old page table entry has valid page address,
18789                                  <1>         ;     that page will be deallocated just before PTE will
18790                                  <1>         ;     be changed for direct (1 to 1) memory page access.
18791                                  <1>         ;
18792                                  <1>         ;     If old PTE value points to a swapped page,
18793                                  <1>           ;       that page (block) will be unlinked on swap disk.
18794                                  <1>         ;
18795                                  <1>         ;     Newly allocated pages (except page tables) will not
18796                                  <1>         ;     be applied to Memory Allocation Table.
18797                                  <1>         ;     AVL bit 1 (PTE bit 10) of page table entry will be
18798                                  <1>         ;     used to indicate shared (direct) memory page; then,
18799                                  <1>         ;     this page will not be deallocated later during
18800                                  <1>         ;     process termination. (Memory Allocation Table and
18801                                  <1>         ;     free memory count will not be affected.
18802                                  <1>         ;     (Except deallocating page table's itself.)
18803                                  <1>         ;
18804                                  <1>         ;      CF = 1 -> error (EAX = error code)
18805                                  <1>         ;      CF = 0 -> success (EAX = beginning address)
18806                                  <1>         ;
18807                                  <1>         ;; (Modified Registers -> none)
18808                                  <1>         ; Modified registers: ebp, edx, ecx, ebx, esi, edi
18809                                  <1>         ;
18810                                  <1>
18811                                  <1>         ;push  ebp
18812                                  <1>         ;push  ebx
18813                                  <1>         ;push  ecx
18814                                  <1>         ;push  edx
18815 000056A1 662500F0               <1>         and    ax, PTE_A_CLEAR ; clear page offset
18816 000056A5 50                      <1>         push   eax
18817                                  <1>         ;and   ecx, ecx ; page count
18818                                  <1>         ;jz    dmem_acc_7  ; 'insufficient memory' error
18819 000056A6 89C5                    <1>         mov    ebp, eax
18820 000056A8 81C300004000            <1>         add    ebx, CORE ; 12/05/2017
18821                                  <1> dmem_acc_0:
18822 000056AE 891D[E0690100]          <1>         mov    [base_addr], ebx ; 12/05/2017
18823 000056B4 A1[B8030300]            <1>         mov    eax, [u.pgdir] ; page dir address (physical)
18824 000056B9 E8D7F5FFFF              <1>         call   get_pte
18825                                  <1>                ; EDX = Page table entry address (if CF=0)
18826                                  <1>                  ;       Page directory entry address (if CF=1)
18827                                  <1>                ;       (Bit 0 value is 0 if PT is not present)
18828                                  <1>                ; EAX = Page table entry value (page address)
18829                                  <1>                ;     CF = 1 -> PDE not present or invalid ?
18830 000056BE 7324                    <1>         jnc    short dmem_acc_1
18831                                  <1>         ;
18832 000056C0 E8B5F4FFFF              <1>         call   allocate_page
18833 000056C5 0F82AB000000            <1>           jc     dmem_acc_7  ; 'insufficient memory' error
18834                                  <1>         ;
18835 000056CB E824F5FFFF              <1>         call   clear_page
18836                                  <1>         ; EAX = Physical (base) address of the allocated (new) page
18837 000056D0 0C07                    <1>         or     al, PDE_A_PRESENT + PDE_A_WRITE + PDE_A_USER ; 4+2+1 = 7
18838                                  <1>                           ; lower 3 bits are used as U/S, R/W, P flags
18839                                  <1>                           ; (user, writable, present page)
18840 000056D2 8902                    <1>         mov    [edx], eax ; Let's put the new page directory entry here !
18841 000056D4 A1[B8030300]            <1>         mov    eax, [u.pgdir]
18842 000056D9 E8B7F5FFFF              <1>         call   get_pte
18843 000056DE 0F8292000000            <1>           jc     dmem_acc_7 ; 'insufficient memory' error
18844                                  <1> dmem_acc_1:
18845                                  <1>         ; EAX = PTE value, EDX = PTE address
18846 000056E4 A801                    <1>         test   al, PTE_A_PRESENT
```

```
18847 000056E6 750D                <1>       jnz    short dmem_acc_2
18848 000056E8 09C0                <1>       or     eax, eax
18849 000056EA 7468                <1>       jz     short dmem_acc_6   ; Change PTE
18850 000056EC D1E8                <1>       shr    eax, 1       ; swap disk block (8 sectors) address
18851                              <1>       ; unlink swap disk block
18852 000056EE E80CFBFFFF          <1>       call   unlink_swap_block
18853 000056F3 EB5F                <1>       jmp    short dmem_acc_6
18854                              <1>
18855                              <1> dmem_acc_2:
18856 000056F5 A802                <1>       test   al, PTE_A_WRITE   ; bit 1, writable (r/w) flag
18857                              <1>                                ; (must be 1)
18858 000056F7 7550                <1>       jnz    short dmem_acc_4
18859                              <1>       ; Read only -duplicated- page (belongs to a parent or a child)
18860 000056F9 66A90002            <1>         test    ax, PTE_DUPLICATED ; Was this page duplicated
18861                              <1>                                    ; as child's page ?
18862 000056FD 7455                <1>       jz     short dmem_acc_5 ; Change PTE but don't deallocate the page!
18863                              <1>
18864                              <1>       ;push  edi
18865                              <1>       ;push  esi
18866                              <1>
18867 000056FF 51                  <1>       push   ecx
18868                              <1>       ;push  ebx
18869 00005700 8B1D[BC030300]      <1>       mov    ebx, [u.ppgdir] ; parent's page dir address (physical)
18870                              <1>
18871                              <1>       ; check the parent's PTE value is read only & same page or not..
18872 00005706 89EF                <1>       mov    edi, ebp
18873 00005708 C1EF16              <1>       shr    edi, PAGE_D_SHIFT ; 22
18874                              <1>       ; EDI = page directory entry index (0-1023)
18875 0000570B 89EE                <1>       mov    esi, ebp
18876 0000570D C1EE0C              <1>       shr    esi, PAGE_SHIFT ; 12
18877 00005710 81E6FF030000        <1>       and    esi, PTE_MASK
18878                              <1>       ; ESI = page table entry index (0-1023)
18879                              <1>
18880 00005716 66C1E702            <1>       shl    di, 2 ; * 4
18881 0000571A 01FB                <1>       add    ebx, edi ; PDE offset (for the parent)
18882 0000571C 8B0F                <1>       mov    ecx, [edi]
18883 0000571E F6C101              <1>       test   cl, PDE_A_PRESENT ; present (valid) or not ?
18884 00005721 7425                <1>       jz     short dmem_acc_3    ; parent process does not use this page
18885 00005723 6681E100F0          <1>       and    cx, PDE_A_CLEAR ; 0F000h ; Clear attribute bits
18886 00005728 66C1E602            <1>       shl    si, 2 ; *4
18887 0000572C 01CE                <1>       add    esi, ecx ; PTE offset (for the parent)
18888 0000572E 8B1E                <1>       mov    ebx, [esi]
18889 00005730 F6C301              <1>       test   bl, PTE_A_PRESENT ; present or not ?
18890 00005733 7413                <1>       jz     short dmem_acc_3    ; parent process does not use this page
18891 00005735 662500F0            <1>       and    ax, PTE_A_CLEAR ; 0F000h ; Clear attribute bits
18892 00005739 6681E300F0          <1>       and    bx, PTE_A_CLEAR ; 0F000h ; Clear attribute bits
18893 0000573E 39D8                <1>       cmp    eax, ebx     ; parent's and child's pages are same ?
18894 00005740 7506                <1>       jne    short dmem_acc_3    ; not same page
18895                              <1>                                ; deallocate the child's page
18896 00005742 800E02              <1>         or     byte [esi], PTE_A_WRITE ; convert to writable page (parent)
18897                              <1>       ;pop   ebx
18898 00005745 59                  <1>       pop    ecx
18899 00005746 EB0C                <1>       jmp    short dmem_acc_5
18900                              <1> dmem_acc_3:
18901                              <1>       ;pop   ebx
18902 00005748 59                  <1>       pop    ecx
18903                              <1> dmem_acc_4:
18904 00005749 66A90004            <1>       test   ax, PTE_SHARED ; shared or direct memory access indicator
18905 0000574D 7505                <1>       jnz    short dmem_acc_5   ; AVL bit 1 = 1, do not deallocate this page!
18906                              <1>       ;
18907                              <1>       ;and   ax, PTE_A_CLEAR ; 0F000h ; clear lower 12 (attribute) bits
18908 0000574F E804F6FFFF          <1>       call   deallocate_page
18909                              <1> dmem_acc_5:
18910                              <1>       ;pop   esi
18911                              <1>       ;pop   edi
18912                              <1> dmem_acc_6:
18913 00005754 89E8                <1>       mov    eax, ebp ; physical page (offset=0) address
18914                              <1>       ; EAX = memory page address
18915                              <1>       ; EDX = PTE entry address (physical)
18916 00005756 660D0704            <1>       or     ax, PTE_A_PRESENT+PTE_A_USER+PTE_A_WRITE+PTE_SHARED
18917                              <1>                    ; present flag, bit 0 = 1
18918                              <1>                    ; user flag, bit 2 = 1
18919                              <1>                    ; writable flag, bit 1 = 1
18920                              <1>                    ; direct memory access flag, bit 10 = 1
18921                              <1>                    ; (This page must not be deallocated!)
18922 0000575A 8902                <1>       mov    [edx], eax  ; Update PTE value
18923 0000575C 49                  <1>       dec    ecx ; remain count of contiguous pages
18924 0000575D 741E                <1>       jz     short dmem_acc_8
18925 0000575F 81C500100000        <1>       add    ebp, PAGE_SIZE ; next physical page address
18926                              <1>       ; 22/07/2017
18927                              <1>       ;mov   eax, ebp
18928                              <1>       ; 12/05/2017
18929 00005765 8B1D[E0690100]      <1>       mov    ebx, [base_addr] ; linear address (virtual+CORE)
18930 0000576B 81C300100000        <1>       add    ebx, PAGE_SIZE      ; next linear address
18931 00005771 E938FFFFFF          <1>         jmp     dmem_acc_0
18932                              <1> dmem_acc_7:  ; ERROR !
18933 00005776 C7042404000000      <1>       mov    dword [esp], ERR_MINOR_IM
18934                              <1>              ; Insufficient memory (minor) error!
18935                              <1>              ; Major error = 0 (No protection fault)
18936                              <1>       ; cf = 1
18937                              <1> dmem_acc_8:
18938 0000577D 58                  <1>       pop    eax
18939                              <1>       ;pop   edx
18940                              <1>       ;pop   ecx
18941                              <1>       ;pop   ebx
18942                              <1>       ;pop   ebp
18943 0000577E C3                  <1>       retn
18944                              <1>
18945                              <1> deallocate_user_pages:
18946                              <1>       ; 20/05/2017
18947                              <1>       ; 15/05/2017
18948                              <1>       ; 20/02/2017
18949                              <1>       ; 19/02/2017 (TRDOS 386 = TRDOS v2.0)
```

```
18950                                  <1>      ;
18951                                  <1>      ; Deallocate virtually contiguous user pages (memory block)
18952                                  <1>      ; (caller: 'sysdalloc' system call)
18953                                  <1>      ;
18954                                  <1>      ; INPUT ->
18955                                  <1>      ;     EBX = VIRTUAL ADDRESS (beginning address)
18956                                  <1>      ;     ECX = byte count
18957                                  <1>      ;     [u.pgdir] = user's page directory
18958                                  <1>      ;     [u.ppdir] = parent's page directory
18959                                  <1>      ;
18960                                  <1>      ; OUTPUT ->
18961                                  <1>      ;    If CF = 0
18962                                  <1>      ;     EAX = Deallocated memory bytes
18963                                  <1>      ;       (Even if shared or read only pages will not be
18964                                  <1>      ;        deallocated on M.A.T., this byte count will be
18965                                  <1>      ;        returned as virtually deallocated bytes; in fact
18966                                  <1>      ;        virtually deallocated user pages * 4096.)
18967                                  <1>      ;     EBX = Virtual address (as rounded up)
18968                                  <1>      ;    If CF = 1
18969                                  <1>      ;     EAX = 0 (there is not any deallocated pages)
18970                                  <1>      ;
18971                                  <1>      ; Note: Empty page tables will not be deallocated!!!
18972                                  <1>      ;     (they will be deallocated at process termination stage)
18973                                  <1>      ;
18974                                  <1>      ; Modified Registers -> EAX, EDX, ESI, EDI, EBX, ECX, EBP
18975                                  <1>      ;
18976 0000577F 89DE                   <1>      mov    esi, ebx
18977 00005781 89F7                   <1>      mov    edi, esi
18978 00005783 01CF                   <1>      add    edi, ecx
18979 00005785 81C6FF0F0000           <1>      add    esi, PAGE_SIZE - 1  ; 4095 (round up)
18980 0000578B C1EE0C                 <1>      shr    esi, PAGE_SHIFT
18981 0000578E C1EF0C                 <1>      shr    edi, PAGE_SHIFT
18982 00005791 89F8                   <1>      mov    eax, edi ; end page
18983 00005793 29F0                   <1>      sub    eax, esi ; end page - start page
18984 00005795 0F86D5000000           <1>      jna    da_u_pd_err  ; < 1
18985 0000579B 89F3                   <1>      mov    ebx, esi
18986 0000579D C1E30C                 <1>      shl    ebx, PAGE_SHIFT ; virtual address (as rounded up)
18987 000057A0 53                     <1>      push   ebx ; *
18988 000057A1 89C1                   <1>      mov    ecx, eax ; page count
18989 000057A3 C1E00C                 <1>      shl    eax, PAGE_SHIFT ; byte count as adjusted
18990 000057A6 50                     <1>      push   eax ; **
18991 000057A7 8B1D[B8030300]         <1>      mov    ebx, [u.pgdir] ; physical addr of user's page dir
18992 000057AD 81C600040000           <1>      add    esi, CORE/PAGE_SIZE
18993 000057B3 89F7                   <1>      mov    edi, esi
18994 000057B5 81E7FF030000           <1>      and    edi, PTE_MASK ; PTE entry in the page table
18995 000057BB 57                     <1>      push   edi ; *** ; PTE index (of page directory)
18996 000057BC C1EE0A                 <1>      shr    esi, PAGE_D_SHIFT - PAGE_SHIFT ; 22-12=10
18997 000057BF 89F2                   <1>      mov    edx, esi
18998                                  <1>      ; EDX = PDE index
18999 000057C1 C1E602                 <1>      shl    esi, 2 ; convert PDE index to dword offset
19000 000057C4 01DE                   <1>      add    esi, ebx ; add page directory address
19001                                  <1> da_u_pd_1:
19002 000057C6 AD                     <1>      lodsd
19003                                  <1>      ;
19004 000057C7 89F5                   <1>      mov    ebp, esi ; 20/02/2017
19005                                  <1>      ; EBP = next PDE address
19006                                  <1>      ;
19007 000057C9 A801                   <1>      test   al, PDE_A_PRESENT ; bit 0, present flag (must be 1)
19008 000057CB 0F8494000000           <1>      jz     da_u_pd_3 ; 20/05/2017
19009 000057D1 662500F0               <1>      and    ax, PDE_A_CLEAR ; 0F000h ; clear lower 12 (attribute) bits
19010                                  <1>      ; EAX = PHYSICAL (flat) ADDRESS OF THE PAGE TABLE
19011 000057D5 8B3C24                 <1>      mov    edi, [esp] ; ***
19012                                  <1>      ; EDI = PTE index (of complete page directory)
19013                                  <1>      ;and   edi, PTE_MASK ; PTE entry in the page table
19014 000057D8 C1E702                 <1>      shl    edi, 2 ; convert PTE index to dword offset
19015 000057DB 89FE                   <1>      mov    esi, edi ; PTE offset in page table (0-4092)
19016 000057DD 01C6                   <1>      add    esi, eax ; now, esi points to requested PTE
19017                                  <1> da_u_pt_0:
19018 000057DF AD                     <1>      lodsd
19019 000057E0 A801                   <1>      test   al, PTE_A_PRESENT ; bit 0, present flag (must be 1)
19020 000057E2 743F                   <1>      jz     short da_u_pt_1
19021                                  <1>      ;
19022 000057E4 A802                   <1>      test   al, PTE_A_WRITE   ; bit 1, writable (r/w) flag
19023                                  <1>                               ; (must be 1)
19024 000057E6 7549                   <1>      jnz    short da_u_pt_3
19025                                  <1>      ; Read only -duplicated- page (belongs to a parent or a child)
19026 000057E8 66A90002               <1>        test    ax, PTE_DUPLICATED ; Was this page duplicated
19027                                  <1>                                 ; as child's page ?
19028 000057EC 744E                   <1>      jz     short da_u_pt_4 ; Clear PTE but don't deallocate the page!
19029                                  <1>      ;
19030                                  <1>      ; check the parent's PTE value is read only & same page or not..
19031                                  <1>      ; EDX = page directory entry index (0-1023)
19032 000057EE 52                     <1>      push   edx ; ****
19033                                  <1>      ; EDI = page table entry offset (0-4092)
19034 000057EF 8B1D[BC030300]         <1>      mov    ebx, [u.ppgdir] ; page directory of the parent process
19035 000057F5 66C1E202               <1>      shl    dx, 2 ; *4
19036 000057F9 01D3                   <1>      add    ebx, edx ; PDE address (for the parent)
19037 000057FB 8B13                   <1>      mov    edx, [ebx] ; page table address
19038 000057FD F6C201                 <1>      test   dl, PDE_A_PRESENT ; present (valid) or not ?
19039 00005800 742E                   <1>      jz     short da_u_pt_2     ; parent process does not use this page
19040 00005802 6681E200F0             <1>      and    dx, PDE_A_CLEAR ; 0F000h ; Clear attribute bits
19041                                  <1>      ; EDI = page table entry offset (0-4092)
19042 00005807 01D7                   <1>      add    edi, edx     ; PTE address (for the parent)
19043 00005809 8B1F                   <1>      mov    ebx, [edi]
19044 0000580B F6C301                 <1>      test   bl, PTE_A_PRESENT ; present or not ?
19045 0000580E 7420                   <1>      jz     short da_u_pt_2     ; parent process does not use this page
19046 00005810 662500F0               <1>      and    ax, PTE_A_CLEAR ; 0F000h ; Clear attribute bits
19047 00005814 6681E300F0             <1>      and    bx, PTE_A_CLEAR ; 0F000h ; Clear attribute bits
19048 00005819 39D8                   <1>      cmp    eax, ebx     ; parent's and child's pages are same ?
19049 0000581B 7513                   <1>      jne    short da_u_pt_2     ; not same page
19050                                  <1>                                ; deallocate the child's page
19051 0000581D 800F02                 <1>        or      byte [edi], PTE_A_WRITE ; convert to writable page (parent)
19052 00005820 5A                     <1>      pop    edx ; ****
```

```
19053 00005821 EB19              <1>       jmp    short da_u_pt_4
19054                            <1> da_u_pt_1:
19055 00005823 09C0              <1>       or     eax, eax     ; swapped page ?
19056 00005825 741C              <1>       jz     short da_u_pt_5     ; no
19057                            <1>                                  ; yes
19058 00005827 D1E8              <1>       shr    eax, 1
19059 00005829 E8D1F9FFFF        <1>       call   unlink_swap_block ; Deallocate swapped page block
19060                            <1>                                ; on the swap disk (or in file)
19061 0000582E EB13              <1>       jmp    short da_u_pt_5
19062                            <1> da_u_pt_2:
19063 00005830 5A                <1>       pop    edx ; ****
19064                            <1> da_u_pt_3:
19065 00005831 66A90004          <1>       test   ax, PTE_SHARED      ; shared or direct memory access indicator
19066 00005835 7505              <1>       jnz    short da_u_pt_4     ; AVL bit 1 = 1, do not deallocate this page!
19067                            <1>       ;
19068                            <1>       ;and   ax, PTE_A_CLEAR ; 0F000h ; clear lower 12 (attribute) bits
19069 00005837 E81CF5FFFF        <1>       call   deallocate_page ; set the mem allocation bit of this page
19070                            <1> da_u_pt_4:
19071 0000583C C746FC00000000    <1>       mov    dword [esi-4], 0 ; clear/reset PTE (child, dupl. as parent)
19072                            <1> da_u_pt_5:
19073                            <1>       ; 20/05/2017
19074 00005843 58                <1>       pop    eax ; *** PTE index (of page directory)
19075 00005844 49                <1>       dec    ecx ; remain page count
19076 00005845 7426              <1>       jz     short da_u_pd_4
19077 00005847 40                <1>       inc    eax ; next PTE
19078 00005848 6625FF03          <1>       and    ax, PTE_MASK ; PTE entry index in the page table
19079 0000584C 50                <1>       push   eax ; *** (save again)
19080                            <1>       ;mov   edi, eax
19081                            <1>       ;and   di, PTE_MASK
19082                            <1>       ;cmp   edi, PAGE_SIZE / 4 ; 1024
19083                            <1>       ;jnb   short da_u_pd_2
19084 0000584D 89C7              <1>       mov    edi, eax
19085 0000584F C1E702            <1>       shl    edi, 2 ; convert index to dword offset
19086                            <1>       ;test  ax, PTE_MASK ; 3FFh
19087 00005852 09C0              <1>       or     eax, eax
19088 00005854 7589              <1>       jnz    short da_u_pt_0 ; 1-1023
19089                            <1> da_u_pd_2:
19090 00005856 42                <1>       inc    edx
19091                            <1>       ; 20/05/2017
19092 00005857 6681E2FF03        <1>       and    dx, PTE_MASK  ; 3FFh
19093 0000585C 740F              <1>       jz     short da_u_pd_4  ; 0 (1024)
19094                            <1>       ;cmp   edx, 1024
19095                            <1>       ;jnb   short da_u_pd_4
19096 0000585E 89EE              <1>       mov    esi, ebp ; 20/02/2017
19097 00005860 E961FFFFFF        <1>       jmp    da_u_pd_1
19098                            <1> da_u_pd_3:
19099                            <1>       ; 15/05/2017 (empty page directory entry)
19100 00005865 81E900040000      <1>       sub    ecx, 1024
19101 0000586B 77E9              <1>       ja     short da_u_pd_2 ; 20/05/2017
19102                            <1> da_u_pd_4:
19103 0000586D 58                <1>       pop    eax ; **
19104 0000586E 5B                <1>       pop    ebx ; *
19105 0000586F C3                <1>       retn
19106                            <1>
19107                            <1> da_u_pd_err:
19108 00005870 31C0              <1>       xor    eax, eax
19109 00005872 F9                <1>       stc
19110 00005873 C3                <1>       retn
19111                            <1>
19112                            <1> allocate_user_pages:
19113                            <1>       ; 20/05/2017
19114                            <1>       ; 01/05/2017, 02/05/2017, 15/05/2017
19115                            <1>       ; 04/03/2017
19116                            <1>       ; 20/02/2017 (TRDOS 386 = TRDOS v2.0)
19117                            <1>       ;
19118                            <1>       ; Allocate physically contiguous user pages (memory block)
19119                            <1>       ; (caller: 'sysalloc' system call)
19120                            <1>       ;
19121                            <1>       ; Note: This procedure does not alloc a page's itself
19122                            <1>       ;     (page bit) on Memory Allocation Table.
19123                            <1>       ;     (allocate_memory_block is needed before this proc)
19124                            <1>       ;
19125                            <1>       ; INPUT ->
19126                            <1>       ;     EAX = PHYSICAL ADDRESS (beginning address)
19127                            <1>       ;     EBX = VIRTUAL ADDRESS (beginning address)
19128                            <1>       ;     ECX = byte count (>=4096)
19129                            <1>       ;     [u.pgdir] = user's page directory
19130                            <1>       ;
19131                            <1>       ;     Note: All addresses are (must be) already adjusted
19132                            <1>       ;     to page      borders, otherwise, lower 12bits of addresses
19133                            <1>       ;     and byte count would be truncated.
19134                            <1>       ;
19135                            <1>       ; OUTPUT ->
19136                            <1>       ;     none
19137                            <1>       ;
19138                            <1>       ;     CF = 1 -> insufficient memory error
19139                            <1>       ;
19140                            <1>       ; Note: All pages will be allocated in physical page order
19141                            <1>       ;     from the beginning page address.
19142                            <1>       ;     * A new page table will be added to the page dir
19143                            <1>       ;       when the requested PDE is invalid.
19144                            <1>       ;     * Those pages will not be added to swap queue
19145                            <1>       ;       because main purpose of this allocation is to
19146                            <1>       ;       set a direct memory access (DMA controller) buffer.
19147                            <1>       ;      (Swapping out a page in a DMA buffer would be wrong!)
19148                            <1>       ;     * Previous content of page tables (PTEs) would be
19149                            <1>       ;       (should be) deallocated before entering this
19150                            <1>       ;       procedure. So, new page table entries (PTEs)
19151                            <1>       ;       directly will be written without checking
19152                            <1>       ;       their previous content.
19153                            <1>       ;     * Only solution to increase free memory by removing
19154                            <1>       ;       that non-swappable memory block is to terminate
19155                            <1>       ;       the process or to wait until the process will
```

```
19156                                  <1>       ;       deallocate that memory block as itself. ('sysdalloc')
19157                                  <1>       ;       (No problem, if the process does not grab all of
19158                                  <1>       ;       -very big amount of- free memory by using
19159                                  <1>       ;       'sysalloc' system call!?)
19160                                  <1>       ;       (Even if the process has grabbed all of free memory,
19161                                  <1>       ;       no problem if the process is not running in
19162                                  <1>       ;       multitasking mode. No problem in multitasking
19163                                  <1>       ;       mode if there is not another process which is running
19164                                  <1>       ;       or waiting or sleeping for an event as it's pages
19165                                  <1>       ;       are swapped-out. But a new process can not start to
19166                                  <1>       ;       run if all of free memory has beeen allocated
19167                                  <1>       ;       by running processes. Deallocation -'sysdalloc'-
19168                                  <1>       ;       or terminate a running process is needed
19169                                  <1>       ;       in order to run a new process.)
19170                                  <1>       ;
19171                                  <1>       ; Modified Registers -> EAX, EDX, ESI, EDI, EBX, ECX, EBP
19172                                  <1>       ;
19173                                  <1>
19174                                  <1>       ; 01/05/2017
19175 00005874 662500F0              <1>       and    ax, ~PAGE_OFF
19176 00005878 6681E300F0            <1>       and    bx, ~PAGE_OFF
19177                                  <1>       ; 02/05/2017
19178 0000587D BD00F0FFFF            <1>       mov    ebp, 0FFFFF000h ; 4 Giga Bytes - 4096 Bytes (for Stack)
19179 00005882 C1E90C                <1>       shr    ecx, PAGE_SHIFT ; page count
19180 00005885 83F901                <1>       cmp    ecx, 1
19181 00005888 7251                  <1>       jb     short a_u_im_retn
19182 0000588A 89C2                  <1>       mov    edx, eax
19183 0000588C 01CA                  <1>       add    edx, ecx
19184 0000588E 724B                  <1>       jc     short a_u_im_retn
19185 00005890 39D5                  <1>       cmp    ebp, edx
19186 00005892 7247                  <1>       jb     short a_u_im_retn
19187 00005894 89DA                  <1>       mov    edx, ebx
19188 00005896 81C200004000          <1>       add    edx, CORE
19189 0000589C 723D                  <1>       jc     short a_u_im_retn
19190 0000589E 01CA                  <1>       add    edx, ecx
19191 000058A0 7239                  <1>       jc     short a_u_im_retn
19192 000058A2 39D5                  <1>       cmp    ebp, edx
19193 000058A4 7235                  <1>       jb     short a_u_im_retn
19194                                  <1>       ;
19195 000058A6 89C5                  <1>       mov    ebp, eax ; physical address
19196 000058A8 89DE                  <1>       mov    esi, ebx
19197 000058AA 81C600004000          <1>       add    esi, CORE ; start of user's memory (4M)
19198 000058B0 C1EE0C                <1>       shr    esi, PAGE_SHIFT ; higher 20 bits of the linear address
19199                                  <1>       ;shr   ecx, PAGE_SHIFT ; page count
19200 000058B3 8B1D[B8030300]        <1>       mov    ebx, [u.pgdir] ; physical addr of user's page dir
19201 000058B9 89F7                  <1>       mov    edi, esi
19202 000058BB 81E7FF030000          <1>       and    edi, PTE_MASK ; PTE entry index in the page table
19203 000058C1 57                    <1>       push   edi  ; * ; PTE index (in page directory)
19204 000058C2 C1EE0A                <1>       shr    esi, PAGE_D_SHIFT - PAGE_SHIFT ; 22-12=10
19205 000058C5 89F2                  <1>       mov    edx, esi
19206                                  <1>       ; EDX = PDE index
19207 000058C7 C1E602                <1>       shl    esi, 2 ; convert PDE index to dword offset
19208 000058CA 01DE                  <1>       add    esi, ebx ; add page directory address
19209                                  <1> a_u_pd_0:
19210 000058CC AD                    <1>       lodsd
19211                                  <1>       ;
19212 000058CD 89F3                  <1>       mov    ebx, esi ; next PDE address
19213                                  <1>       ;
19214 000058CF A801                  <1>       test   al, PDE_A_PRESENT ; bit 0, present flag (must be 1)
19215 000058D1 7513                  <1>       jnz    short a_u_pd_2
19216                                  <1>       ;
19217                                  <1>       ; empty PDE (it does not point to valid page table address)
19218 000058D3 E8A2F2FFFF            <1>       call   allocate_page  ; (allocate a new page table)
19219 000058D8 7302                  <1>       jnc    short a_u_pd_1 ; OK... now, we have a new page table.
19220                                  <1>       ; cf = 1
19221                                  <1>       ; There is not a free memory page to allocate a new page table !!!
19222 000058DA 5E                    <1>       pop    esi ; *
19223                                  <1> a_u_im_retn:
19224 000058DB C3                    <1>       retn   ; return to 'sysalloc' with 'insufficient memory' error
19225                                  <1>       ;
19226                                  <1> a_u_pd_1: ; clear the new page table content
19227                                  <1>       ; EAX = Physical (base) address of the new page table
19228 000058DC E813F3FFFF            <1>       call   clear_page ; Clear page content
19229                                  <1>       ;
19230 000058E1 0C07                  <1>       or     al, PDE_A_PRESENT + PDE_A_WRITE + PDE_A_USER
19231                                  <1>              ; set bit 0, bit 1 and bit 2 to 1
19232                                  <1>              ; (present, writable, user)
19233 000058E3 8946FC                <1>       mov    [esi-4], eax
19234                                  <1> a_u_pd_2:
19235 000058E6 662500F0              <1>       and    ax, PDE_A_CLEAR ; 0F000h ; clear lower 12 (attribute) bits
19236                                  <1>       ; EAX = PHYSICAL (flat) ADDRESS OF THE PAGE TABLE
19237 000058EA 8B3C24                <1>       mov    edi, [esp] ; *
19238                                  <1>       ; EDI = PTE index (of page directory)
19239                                  <1>       ;and   edi, PTE_MASK ; PTE entry index in the page table
19240                                  <1>       ; EBX = next PDE address
19241 000058ED 89FE                  <1>       mov    esi, edi ; PTE index in page table (0-1023)
19242 000058EF C1E702                <1>       shl    edi, 2 ; convert PTE index to dword offset
19243 000058F2 01C7                  <1>       add    edi, eax ; now, edi points to requested PTE
19244                                  <1> a_u_pt_0:
19245                                  <1>       ; 02/05/2017
19246 000058F4 8B07                  <1>       mov    eax, [edi]
19247                                  <1>       ;
19248 000058F6 A801                  <1>       test   al, PTE_A_PRESENT ; bit 0, present flag (must be 1)
19249 000058F8 7445                  <1>       jz     short a_u_pt_1
19250                                  <1>       ;
19251 000058FA A802                  <1>       test   al, PTE_A_WRITE   ; bit 1, writable (r/w) flag
19252                                  <1>                                ; (must be 1)
19253 000058FC 7550                  <1>       jnz    short a_u_pt_3
19254                                  <1>       ; Read only -duplicated- page (belongs to a parent or a child)
19255 000058FE 66A90002              <1>         test    ax, PTE_DUPLICATED ; Was this page duplicated
19256                                  <1>                                ; as child's page ?
19257 00005902 7455                  <1>       jz     short a_u_pt_4      ; Clear PTE but don't deallocate the page!
19258                                  <1>       ;
```

```
19259                                  <1>       ; check the parent's PTE value is read only & same page or not..
19260                                  <1>       ; EDX = page directory entry index (0-1023)
19261 00005904 52                      <1>       push   edx ; **
19262 00005905 53                      <1>       push   ebx ; ***
19263                                  <1>       ; ESI = page table entry index (0-1023)
19264                                  <1>       ;push  esi ; **** ; 20/05/2017
19265 00005906 8B1D[BC030300]          <1>       mov    ebx, [u.ppgdir] ; page directory of the parent process
19266 0000590C 66C1E202                <1>       shl    dx, 2 ; *4
19267 00005910 01D3                    <1>       add    ebx, edx ; PTE address,0 (for the parent)
19268 00005912 8B13                    <1>       mov    edx, [ebx] ; page table address
19269 00005914 F6C201                  <1>       test   dl, PDE_A_PRESENT ; present (valid) or not ?
19270 00005917 7433                    <1>       jz     short a_u_pt_2      ; parent process does not use this page
19271 00005919 6681E200F0              <1>       and    dx, PDE_A_CLEAR ; 0F000h ; Clear attribute bits
19272 0000591E 66C1E602                <1>       shl    si, 2 ; *4
19273                                  <1>       ; ESI = page table entry offset (0-4092)
19274 00005922 01D6                    <1>       add    esi, edx     ; PTE address (for the parent)
19275 00005924 8B1E                    <1>       mov    ebx, [esi]
19276 00005926 F6C301                  <1>       test   bl, PTE_A_PRESENT ; present or not ?
19277 00005929 7421                    <1>       jz     short a_u_pt_2      ; parent process does not use this page
19278 0000592B 662500F0                <1>       and    ax, PTE_A_CLEAR ; 0F000h ; Clear attribute bits
19279 0000592F 6681E300F0              <1>       and    bx, PTE_A_CLEAR ; 0F000h ; Clear attribute bits
19280 00005934 39D8                    <1>       cmp    eax, ebx     ; parent's and child's pages are same ?
19281 00005936 7514                    <1>       jne    short a_u_pt_2      ; not same page
19282                                  <1>                           ; deallocate the child's page
19283 00005938 800E02                  <1>         or      byte [esi], PTE_A_WRITE ; convert to writable page (parent)
19284                                  <1>       ;pop   esi ; **** ; 20/05/2017
19285 0000593B 5B                      <1>       pop    ebx ; ***
19286 0000593C 5A                      <1>       pop    edx ; **
19287 0000593D EB1A                    <1>       jmp    short a_u_pt_4
19288                                  <1> a_u_pt_1:
19289 0000593F 09C0                    <1>       or     eax, eax     ; swapped page ?
19290 00005941 7416                    <1>       jz     short a_u_pt_4      ; no
19291                                  <1>                           ; yes
19292 00005943 D1E8                    <1>       shr    eax, 1
19293 00005945 E8B5F8FFFF              <1>       call   unlink_swap_block ; Deallocate swapped page block
19294                                  <1>                                ; on the swap disk (or in file)
19295 0000594A EB0D                    <1>       jmp    short a_u_pt_4
19296                                  <1> a_u_pt_2:
19297                                  <1>       ;pop   esi ; **** ; 20/05/2017
19298 0000594C 5B                      <1>       pop    ebx ; ***
19299 0000594D 5A                      <1>       pop    edx ; **
19300                                  <1> a_u_pt_3:
19301 0000594E 66A90004                <1>       test   ax, PTE_SHARED      ; shared or direct memory access indicator
19302 00005952 7505                    <1>       jnz    short a_u_pt_4      ; AVL bit 1 = 1, do not deallocate this page!
19303                                  <1>       ;
19304                                  <1>       ;and   ax, PTE_A_CLEAR ; 0F000h ; clear lower 12 (attribute) bits
19305 00005954 E8FFF3FFFF              <1>       call   deallocate_page ; set the mem allocation bit of this page
19306                                  <1>       ;
19307                                  <1> a_u_pt_4:
19308 00005959 89E8                    <1>       mov    eax, ebp ; physical address
19309 0000595B 0C07                    <1>       or     al, PTE_A_PRESENT + PTE_A_WRITE + PTE_A_USER ; 04/03/2017
19310 0000595D AB                      <1>       stosd
19311 0000595E 5E                      <1>       pop    esi ; * ; 20/05/2017
19312 0000595F 49                      <1>       dec    ecx ; remain page count
19313 00005960 7417                    <1>       jz     short a_u_pd_5
19314 00005962 81C500100000            <1>       add    ebp, PAGE_SIZE
19315 00005968 46                      <1>       inc    esi ; next PTE (index)
19316                                  <1>       ; 20/05/2017
19317                                  <1>       ;cmp   esi, PAGE_SIZE/4 ; 1024
19318                                  <1>       ;jb    short a_u_pt_0
19319 00005969 6681E6FF03              <1>       and    si, PTE_MASK ; 3FFh (0 to 1023)
19320 0000596E 56                      <1>       push   esi ; *
19321 0000596F 7583                    <1>       jnz    short a_u_pt_0 ; > 0 (<1024)
19322                                  <1> a_u_pd_3:
19323 00005971 42                      <1>       inc    edx
19324                                  <1> ;     cmp    edx, 1024
19325                                  <1> ;     jnb    short a_u_pd_4 ; 02/05/2017 (error!, ecx > 0)
19326 00005972 89DE                    <1>       mov    esi, ebx ; the next PDE address
19327 00005974 E953FFFFFF              <1>       jmp    a_u_pd_0
19328                                  <1> a_u_pd_4:
19329                                  <1>       ; 02/05/2017
19330                                  <1> ;     stc
19331                                  <1> a_u_pd_5:
19332                                  <1>       ; 20/05/2017
19333                                  <1>       ;pop   edi ; *
19334 00005979 C3                      <1>       retn
19335                                  <1>
19336                                  <1>
19337                                  <1> ; /// End Of MEMORY MANAGEMENT FUNCTIONS ///
19338                                  <1>
19339                                  <1> ;; Data:
19340                                  <1>
19341                                  <1> ; 09/03/2015
19342                                  <1> ;swpq_count: dw 0 ; count of pages on the swap que
19343                                  <1> ;swp_drv:    dd 0 ; logical drive description table address of the swap drive/disk
19344                                  <1> ;swpd_size:  dd 0 ; size of swap drive/disk (volume) in sectors (512 bytes).

19345                                  <1> ;swpd_free:  dd 0 ; free page blocks (4096 bytes) on swap disk/drive (logical)
19346                                  <1> ;swpd_next:  dd 0 ; next free page block
19347                                  <1> ;swpd_last:  dd 0 ; last swap page block
19348                                      %include 'timer.s'   ; 17/01/2015
19349                                  <1> ; ****************************************************************************
19350                                  <1> ; TRDOS386.ASM (TRDOS 386 Kernel) - v2.0.0 - timer.s
19351                                  <1> ; ----------------------------------------------------------------------------
19352                                  <1> ; Last Update: 15/01/2017
19353                                  <1> ; ----------------------------------------------------------------------------
19354                                  <1> ; Beginning: 17/01/2016
19355                                  <1> ; ----------------------------------------------------------------------------
19356                                  <1> ; Assembler: NASM version 2.11 (trdos386.s)
19357                                  <1> ; ----------------------------------------------------------------------------
19358                                  <1> ; Turkish Rational DOS
19359                                  <1> ; Operating System Project v2.0 by ERDOGAN TAN (Beginning: 04/01/2016)
19360                                  <1> ;
```

```
19361                           <1> ; Derived from 'Retro UNIX 386 Kernel - v0.2.1.0' source code by Erdogan Tan
19362                           <1> ;
19363                           <1> ; Derived from 'IBM PC-AT' BIOS source code (1985)
19364                           <1> ; ****************************************************************************
19365                           <1>
19366                           <1> ; TRDOS 386  (TRDOS v2.0) Kernel - TIMER & REAL TIME CLOCK (BIOS) FUNCTIONS
19367                           <1>
19368                           <1> ; IBM PC-AT BIOS Source Code ('BIOS2.ASM')
19369                           <1> ; TITLE BIOS2 ---- 06/10/85 BIOS INTERRUPT ROUTINES
19370                           <1>
19371                           <1> ;
19372                           <1> ; ///////// TIMER (& REAL TIME CLOCK) FUNCTIONS //////////////
19373                           <1>
19374                           <1> int1Ah:
19375                           <1>       ; 29/01/2016
19376                           <1>       ; 17/01/2016 (TRDOS 386 = TRDOS v2.0)
19377 0000597A 9C               <1>       pushfd
19378 0000597B 0E               <1>       push   cs
19379 0000597C E801000000       <1>       call   TIME_OF_DAY_1
19380 00005981 C3               <1>       retn
19381                           <1>
19382                           <1> ;--- INT  1A H -- (TIME OF DAY) ------------------------------------------------
19383                           <1> ;       THIS BIOS ROUTINE ALLOWS THE CLOCKS TO BE SET OR READ                 :
19384                           <1> ;                                                                             :
19385                           <1> ; PARAMETERS:                                                                 :
19386                           <1> ;     (AH) = 00H  READ THE CURRENT SETTING AND RETURN WITH,                   :
19387                           <1> ;                       (CX) = HIGH PORTION OF COUNT                        :
19388                           <1> ;                       (DX) = LOW PORTION OF COUNT                         :
19389                           <1> ;                       (AL) = 0 TIMER HAS NOT PASSED 24 HOURS SINCE LAST READ :
19390                           <1> ;                              1 IF ON ANOTHER DAY. (RESET TO ZERO AFTER READ) :
19391                           <1> ;                                                                             :
19392                           <1> ;     (AH) = 01H  SET THE CURRENT CLOCK USING,                                :
19393                           <1> ;                 (CX) = HIGH PORTION OF COUNT                              :
19394                           <1> ;                 (DX) = LOW PORTION OF COUNT.                              :
19395                           <1> ;                                                                           :
19396                           <1> ;               NOTE: COUNTS OCCUR AT THE RATE OF 1193180/65536 COUNTS/SECOND :
19397                           <1> ;                           (OR ABOUT 18.2 PER SECOND -- SEE EQUATES)       :
19398                           <1> ;                                                                           :
19399                           <1> ;     (AH) = 02H  READ THE REAL TIME CLOCK AND RETURN WITH,                   :
19400                           <1> ;                       (CH) = HOURS IN BCD (00-23)                         :
19401                           <1> ;                       (CL) = MINUTES IN BCD (00-59)                       :
19402                           <1> ;                       (DH) = SECONDS IN BCD (00-59)                       :
19403                           <1> ;                       (DL) = DAYLIGHT SAVINGS ENABLE (00-01)              :
19404                           <1> ;                                                                           :
19405                           <1> ;     (AH) = 03H  SET THE REAL TIME CLOCK USING,                              :
19406                           <1> ;                     (CH) = HOURS IN BCD (00-23)                           :
19407                           <1> ;                     (CL) = MINUTES IN BCD (00-59)                         :
19408                           <1> ;                     (DH) = SECONDS IN BCD (00-59)                         :
19409                           <1> ;                     (DL) = 01 IF DAYLIGHT SAVINGS ENABLE OPTION, ELSE 00. :
19410                           <1> ;                                                                           :
19411                           <1> ;              NOTE: (DL) = 00 IF DAYLIGHT SAVINGS TIME ENABLE IS NOT ENABLED. :
19412                           <1> ;                    (DL) = 01 ENABLES TWO SPECIAL UPDATES THE LAST SUNDAY IN  :
19413                           <1> ;                 APRIL   (1:59:59 --> 3:00:00 AM) AND THE LAST SUNDAY IN     :
19414                           <1> ;                    OCTOBER (1:59:59 --> 1:00:00 AM) THE FIRST TIME.         :
19415                           <1> ;                                                                           :
19416                           <1> ;     (AH) = 04H  READ THE DATE FROM THE REAL TIME CLOCK AND RETURN WITH,     :
19417                           <1> ;                       (CH) = CENTURY IN BCD (19 OR 20)                    :
19418                           <1> ;                       (CL) = YEAR IN BCD (00-99)                          :
19419                           <1> ;                       (DH) = MONTH IN BCD (01-12)                         :
19420                           <1> ;                       (DL) = DAY IN BCD (01-31).                          :
19421                           <1> ;                                                                           :
19422                           <1> ;     (AH) = 05H  SET THE DATE INTO THE REAL TIME CLOCK USING,                :
19423                           <1> ;                     (CH) = CENTURY IN BCD (19 OR 20)                      :
19424                           <1> ;                     (CL) = YEAR IN BCD (00-99)                            :
19425                           <1> ;                     (DH) = MONTH IN BCD (01-12)                           :
19426                           <1> ;                     (DL) = DAY IN BCD (01-31).                            :
19427                           <1> ;                                                                           :
19428                           <1> ;     (AH) = 06H  SET THE ALARM TO INTERRUPT AT SPECIFIED TIME,               :
19429                           <1> ;                     (CH) = HOURS IN BCD (00-23 (OR FFH))                  :
19430                           <1> ;                     (CL) = MINUTES IN BCD (00-59 (OR FFH))                :
19431                           <1> ;                     (DH) = SECONDS IN BCD (00-59 (OR FFH))                :
19432                           <1> ;                                                                           :
19433                           <1> ;     (AH) = 07H  RESET THE ALARM INTERRUPT FUNCTION.                       :
19434                           <1> ;                                                                           :
19435                           <1> ; NOTES: FOR ALL RETURNS CY= 0 FOR SUCCESSFUL OPERATION.                    :
19436                           <1> ;        FOR (AH)= 2, 4, 6 - CARRY FLAG SET IF REAL TIME CLOCK NOT OPERATING. :
19437                           <1> ;        FOR (AH)= 6 - CARRY FLAG SET IF ALARM ALREADY ENABLED.             :
19438                           <1> ;        FOR THE ALARM FUNCTION (AH = 6) THE USER MUST SUPPLY A ROUTINE AND   :
19439                           <1> ;         INTERCEPT THE CORRECT ADDRESS IN THE VECTOR TABLE FOR INTERRUPT 4AH.      :
19440                           <1> ;         USE 0FFH FOR ANY "DO NOT CARE" POSITION FOR INTERVAL INTERRUPTS.    :
19441                           <1> ;        INTERRUPTS ARE DISABLED DURING DATA MODIFICATION.                    :
19442                           <1> ;        AH & AL ARE RETURNED MODIFIED AND NOT DEFINED EXCEPT WHERE INDICATED.      :
19443                           <1> ;--------------------------------------------------------------------------------
19444                           <1>
19445                           <1> ; 15/01/2017
19446                           <1> ; 14/01/2017
19447                           <1> ; 07/01/2017
19448                           <1> ; 02/01/2017
19449                           <1> ; 29/05/2016
19450                           <1> ; 29/01/2016
19451                           <1> ; 17/01/2016 (TRDOS 386 = TRDOS v2.0)
19452                           <1>
19453                           <1> ; 29/05/2016
19454                           <1> ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
19455                           <1> int35h:  ; Date/Time functions
19456                           <1>
19457                           <1> TIME_OF_DAY_1:
19458                           <1>       ;sti                    ; INTERRUPTS BACK ON
19459                           <1>       ; 29/05/2016
19460 00005982 80642408FE       <1>       and    byte [esp+8], 11111110b   ; clear carry bit of eflags register
19461                           <1>       ;
19462 00005987 80FC08           <1>       cmp    ah, (RTC_TBE-RTC_TB)/4    ; CHECK IF COMMAND IN VALID RANGE (0-7)
19463 0000598A F5               <1>       cmc                      ; COMPLEMENT CARRY FOR ERROR EXIT
```

```
                                <1>       ; (*) jc short TIME_9           ; EXIT WITH CARRY = 1 IF NOT VALID
19465 0000598B 721A             <1>       jc    short _TIME_9 ; 29/05/2016
19466                           <1>
19467 0000598D 1E               <1>       push  ds
19468 0000598E 56               <1>       push  esi
19469 0000598F 66BE1000         <1>       mov   si, KDATA         ; kernel data segment
19470 00005993 8EDE             <1>       mov   ds, si
19471                           <1>
19472                           <1>       ;;15/01/2017
19473                           <1>       ; 14/01/2017
19474                           <1>       ; 02/01/2017
19475                           <1>       ;;mov  byte [intflg], 35h  ; date & time interrupt
19476                           <1>       ;sti
19477                           <1>       ;
19478 00005995 C0E402           <1>       shl   ah, 2               ; convert function to dword offset
19479 00005998 0FB6F4           <1>       movzx esi, ah                   ; PLACE INTO ADDRESSING REGISTER
19480                           <1>       ;cli                      ; NO INTERRUPTS DURING TIME FUNCTIONS
19481 0000599B FF96[AD590000]   <1>       call  [esi+RTC_TB]        ; VECTOR TO FUNCTION REQUESTED WITH CY=0
19482                           <1>                                 ; RETURN WITH CARRY FLAG SET FOR RESULT
19483                           <1>       ;sti                      ; INTERRUPTS BACK ON
19484 000059A1 B400             <1>       mov   ah, 0               ; CLEAR (AH) TO ZERO
19485 000059A3 5E               <1>       pop   esi                 ; RECOVER USERS REGISTER
19486 000059A4 1F               <1>       pop   ds                  ; RECOVER USERS SEGMENT SELECTOR
19487                           <1>
19488                           <1>       ;;15/01/2017
19489                           <1>       ; 02/01/2017
19490                           <1>       ;;mov  byte [ss:intflg], 0 ; 07/01/2017
19491                           <1>
19492                           <1> ;TIME_9:
19493                           <1>                                 ; RETURN WITH CY= 0 IF NO ERROR
19494                           <1>       ; (*) 29/05/2016
19495                           <1>       ; (*) retf 4 ; skip eflags on stack
19496 000059A5 7305             <1>       jnc   short _TIME_10
19497                           <1> _TIME_9:
19498                           <1>       ; 29/05/2016 -set carry flag on stack-
19499                           <1>       ; [esp] = EIP
19500                           <1>       ; [esp+4] = CS
19501                           <1>       ; [esp+8] = E-FLAGS
19502 000059A7 804C240801      <1>       or    byte [esp+8], 1     ; set carry bit of eflags register
19503                           <1>       ; [esp+12] = ESP (user)
19504                           <1>       ; [esp+16] = SS (User)
19505                           <1> _TIME_10:
19506 000059AC CF               <1>       iretd
19507                           <1>
19508                           <1>       ; (*) 29/05/2016 - 'ref 4' intruction causes to stack fault
19509                           <1>       ; (OUTER-PRIVILEGE-LEVEL)
19510                           <1>       ; INTEL 80386 PROGRAMMER'S REFERENCE MANUAL 1986
19511                           <1>       ; // RETF instruction:
19512                           <1>       ;
19513                           <1>       ; IF OperandMode=32 THEN
19514                           <1>       ;    Load CS:EIP from stack;
19515                           <1>       ;    Set CS RPL to CPL;
19516                           <1>       ;    Increment eSP by 8 plus the immediate offset if it exists;
19517                           <1>       ;    Load SS:eSP from stack;
19518                           <1>       ; ELSE (* OperandMode=16 *)
19519                           <1>       ;    Load CS:IP from stack;
19520                           <1>       ;    Set CS RPL to CPL;
19521                           <1>       ;    Increment eSP by 4 plus the immediate offset if it exists;
19522                           <1>       ;    Load SS:eSP from stack;
19523                           <1>       ; FI;
19524                           <1>       ;
19525                           <1>       ; //
19526                           <1>                                 ; ROUTINE VECTOR TABLE (AH)=
19527                           <1> RTC_TB:
19528 000059AD [CD590000]       <1>       dd    RTC_00              ; 0 = READ CURRENT CLOCK COUNT
19529 000059B1 [E0590000]       <1>       dd    RTC_10              ; 1 = SET CLOCK COUNT
19530 000059B5 [EE590000]       <1>       dd    RTC_20              ; 2 = READ THE REAL TIME CLOCK TIME
19531 000059B9 [1D5A0000]       <1>       dd    RTC_30              ; 3 = SET REAL TIME CLOCK TIME
19532 000059BD [5F5A0000]       <1>       dd    RTC_40              ; 4 = READ THE REAL TIME CLOCK DATE
19533 000059C1 [8C5A0000]       <1>       dd    RTC_50              ; 5 = SET REAL TIME CLOCK DATE
19534 000059C5 [D95A0000]       <1>       dd    RTC_60              ; 6 = SET THE REAL TIME CLOCK ALARM
19535 000059C9 [2C5B0000]       <1>       dd    RTC_70              ; 7 = RESET ALARM
19536                           <1>
19537                           <1> RTC_TBE     equ   $
19538                           <1>
19539                           <1> RTC_00:                           ; READ TIME COUNT
19540 000059CD A0[A4520100]     <1>       mov   al, [TIMER_OFL]           ; GET THE OVERFLOW FLAG
19541 000059D2 C605[A4520100]00 <1>       mov   byte [TIMER_OFL], 0 ; AND THEN RESET THE OVERFLOW FLAG
19542 000059D9 8B0D[A0520100]   <1>         mov   ecx, [TIMER_LH]         ; GET COUNT OF TIME
19543 000059DF C3               <1>       retn
19544                           <1>
19545                           <1> RTC_10:                           ; SET TIME COUNT
19546 000059E0 890D[A0520100]   <1>         mov   [TIMER_LH], ecx         ; SET TIME COUNT
19547 000059E6 C605[A4520100]00 <1>       mov   byte [TIMER_OFL], 0 ; RESET OVERFLOW FLAG
19548 000059ED C3               <1>       retn                      ; RETURN WITH NO CARRY
19549                           <1>
19550                           <1> RTC_20:                           ; GET RTC TIME
19551 000059EE E8EB010000       <1>       call  UPD_IPR                   ; CHECK FOR UPDATE IN PROCESS
19552 000059F3 7227             <1>       jc    short RTC_29        ; EXIT IF ERROR (CY= 1)
19553                           <1>
19554 000059F5 B000             <1>       mov   al, CMOS_SECONDS    ; SET ADDRESS OF SECONDS
19555 000059F7 E8FD010000       <1>       call  CMOS_READ           ; GET SECONDS
19556 000059FC 88C6             <1>       mov   dh, al              ; SAVE
19557 000059FE B00B             <1>       mov   al, CMOS_REG_B            ; ADDRESS ALARM REGISTER
19558 00005A00 E8F4010000       <1>       call  CMOS_READ           ; READ CURRENT VALUE OF DSE BIT
19559 00005A05 2401             <1>       and   al, 00000001b       ; MASK FOR VALID DSE BIT
19560 00005A07 88C2             <1>       mov   dl, al              ; SET [DL] TO ZERO FOR NO DSE BIT
19561 00005A09 B002             <1>       mov   al, CMOS_MINUTES    ; SET ADDRESS OF MINUTES
19562 00005A0B E8E9010000       <1>       call  CMOS_READ           ; GET MINUTES
19563 00005A10 88C1             <1>       mov   cl, al              ; SAVE
19564 00005A12 B004             <1>         mov   al, CMOS_HOURS          ; SET ADDRESS OF HOURS
19565 00005A14 E8E0010000       <1>       call  CMOS_READ           ; GET HOURS
19566 00005A19 88C5             <1>       mov   ch, al              ; SAVE
```

```
19567 00005A1B F8                <1>       clc                     ; SET CY= 0
19568                            <1> RTC_29:
19569 00005A1C C3                <1>       retn                    ; RETURN WITH RESULT IN CARRY FLAG
19570                            <1>
19571                            <1> RTC_30:                       ; SET RTC TIME
19572 00005A1D E8BC010000        <1>       call  UPD_IPR                 ; CHECK FOR UPDATE IN PROCESS
19573 00005A22 7305              <1>       jnc   short RTC_35      ; GO AROUND IF CLOCK OPERATING
19574 00005A24 E817010000        <1>       call  RTC_STA                 ; ELSE TRY INITIALIZING CLOCK
19575                            <1> RTC_35:
19576 00005A29 88F4              <1>       mov   ah, dh            ; GET TIME BYTE - SECONDS
19577 00005A2B B000              <1>       mov   al, CMOS_SECONDS  ; ADDRESS SECONDS
19578 00005A2D E8E0010000        <1>       call  CMOS_WRITE        ; UPDATE SECONDS
19579 00005A32 88CC              <1>       mov   ah, cl            ; GET TIME BYTE - MINUTES
19580 00005A34 B002              <1>       mov   al, CMOS_MINUTES  ; ADDRESS MINUTES
19581 00005A36 E8D7010000        <1>       call  CMOS_WRITE        ; UPDATE MINUTES
19582 00005A3B 88EC              <1>       mov   ah, ch            ; GET TIME BYTE - HOURS
19583 00005A3D B004              <1>       mov   al, CMOS_HOURS          ; ADDRESS HOURS
19584 00005A3F E8CE010000        <1>       call  CMOS_WRITE        ; UPDATE ADDRESS
19585                            <1>       ;mov  al, CMOS_REG_B          ; ADDRESS ALARM REGISTER
19586                            <1>       ;mov  ah, al
19587 00005A44 66B80B0B          <1>       mov   ax, CMOS_REG_B * 257
19588 00005A48 E8AC010000        <1>       call  CMOS_READ         ; READ CURRENT TIME
19589 00005A4D 2462              <1>       and   al, 01100010b     ; MASK FOR VALID BIT POSITIONS
19590 00005A4F 0C02              <1>       or    al, 00000010b     ; TURN ON 24 HOUR MODE
19591 00005A51 80E201            <1>       and   dl, 00000001b     ; USE ONLY THE DSE BIT
19592 00005A54 08D0              <1>       or    al, dl            ; GET DAY LIGHT SAVINGS TIME BIT (OSE)
19593 00005A56 86E0              <1>       xchg  ah, al            ; PLACE IN WORK REGISTER AND GET ADDRESS
19594 00005A58 E8B5010000        <1>       call  CMOS_WRITE        ; SET NEW ALARM SITS
19595 00005A5D F8                <1>       clc                     ; SET CY= 0
19596 00005A5E C3                <1>       retn                    ; RETURN WITH CY= 0
19597                            <1>
19598                            <1> RTC_40:                       ; GET RTC DATE
19599 00005A5F E87A010000        <1>       call  UPD_IPR                 ; CHECK FOR UPDATE IN PROCESS
19600 00005A64 7225              <1>       jc    short RTC_49      ; EXIT IF ERROR (CY= 1)
19601                            <1>
19602 00005A66 B007              <1>       mov   al, CMOS_DAY_MONTH  ; ADDRESS DAY OF MONTH
19603 00005A68 E88C010000        <1>       call  CMOS_READ         ; READ DAY OF MONTH
19604 00005A6D 88C2              <1>       mov   dl, al            ; SAVE
19605 00005A6F B008              <1>       mov   al, CMOS_MONTH          ; ADDRESS MONTH
19606 00005A71 E883010000        <1>       call  CMOS_READ         ; READ MONTH
19607 00005A76 88C6              <1>       mov   dh, al            ; SAVE
19608 00005A78 B009              <1>       mov   al, CMOS_YEAR     ; ADDRESS YEAR
19609 00005A7A E87A010000        <1>       call  CMOS_READ         ; READ YEAR
19610 00005A7F 88C1              <1>       mov   cl, al            ; SAVE
19611 00005A81 B032              <1>       mov   al, CMOS_CENTURY  ; ADDRESS CENTURY LOCATION
19612 00005A83 E871010000        <1>       call  CMOS_READ         ; GET CENTURY BYTE
19613 00005A88 88C5              <1>       mov   ch, al            ; SAVE
19614 00005A8A F8                <1>       clc                     ; SET CY=0
19615                            <1> RTC_49:
19616 00005A8B C3                <1>       retn                    ; RETURN WITH RESULTS IN CARRY FLAG
19617                            <1>
19618                            <1> RTC_50:                       ; SET RTC DATE
19619 00005A8C E84D010000        <1>       call  UPD_IPR                 ; CHECK FOR UPDATE IN PROCESS
19620 00005A91 7305              <1>       jnc   short RTC_55      ; GO AROUND IF NO ERROR
19621 00005A93 E8A8000000        <1>       call  RTC_STA                 ; ELSE INITIALIZE CLOCK
19622                            <1> RTC_55:
19623 00005A98 66B80600          <1>       mov   ax, CMOS_DAY_WEEK   ; ADDRESS OF DAY OF WEEK BYTE
19624 00005A9C E871010000        <1>       call  CMOS_WRITE        ; LOAD ZEROS TO DAY OF WEEK
19625 00005AA1 88D4              <1>       mov   ah, dl            ; GET DAY OF MONTH BYTE
19626 00005AA3 B007              <1>       mov   al, CMOS_DAY_MONTH  ; ADDRESS DAY OF MONTH BYTE
19627 00005AA5 E868010000        <1>       call  CMOS_WRITE        ; WRITE OF DAY OF MONTH REGISTER
19628 00005AAA 88F4              <1>       mov   ah, dh            ; GET MONTH
19629 00005AAC B008              <1>       mov   al, CMOS_MONTH          ; ADDRESS MONTH BYTE
19630 00005AAE E85F010000        <1>       call  CMOS_WRITE        ; WRITE MONTH REGISTER
19631 00005AB3 88CC              <1>       mov   ah, cl            ; GET YEAR BYTE
19632 00005AB5 B009              <1>       mov   al, CMOS_YEAR     ; ADDRESS YEAR REGISTER
19633 00005AB7 E856010000        <1>       call  CMOS_WRITE        ; WRITE YEAR REGISTER
19634 00005ABC 88EC              <1>       mov   ah, ch            ; GET CENTURY BYTE
19635 00005ABE B032              <1>       mov   al, CMOS_CENTURY  ; ADDRESS CENTURY BYTE
19636 00005AC0 E84D010000        <1>       call  CMOS_WRITE        ; WRITE CENTURY LOCATION
19637                            <1>       ;mov  al, CMOS_REG_B          ; ADDRESS ALARM REGISTER
19638                            <1>       ;mov  ah, al
19639 00005AC5 66B80B0B          <1>       mov   ax, CMOS_REG_B * 257
19640 00005AC9 E82B010000        <1>       call  CMOS_READ           ; READ WIRRENT SETTINGS
19641 00005ACE 247F              <1>       and   al, 07Fh            ; CLEAR 'SET BIT'
19642 00005AD0 86E0              <1>       xchg  ah, al              ; MOVE TO WORK REGISTER
19643 00005AD2 E83B010000        <1>       call  CMOS_WRITE          ; AND START CLOCK UPDATING
19644 00005AD7 F8                <1>       clc                       ; SET CY= 0
19645 00005AD8 C3                <1>       retn                      ; RETURN CY=0
19646                            <1>
19647                            <1> RTC_60:                         ; SET RTC ALARM
19648 00005AD9 B00B              <1>       mov   al, CMOS_REG_B            ; ADDRESS ALARM
19649 00005ADB E819010000        <1>       call  CMOS_READ           ; READ ALARM REGISTER
19650 00005AE0 A820              <1>       test  al, 20h                   ; CHECK FOR ALARM ALREADY ENABLED
19651 00005AE2 F9                <1>       stc                       ; SET CARRY IN CASE OF ERROR
19652 00005AE3 7542              <1>       jnz   short RTC_69        ; ERROR EXIT IF ALARM SET
19653 00005AE5 E8F4000000        <1>       call  UPD_IPR                   ; CHECK FOR UPDATE IN PROCESS
19654 00005AEA 7305              <1>       jnc   short RTC_65        ; SKIP INITIALIZATION IF NO ERROR
19655 00005AEC E84F000000        <1>       call  RTC_STA                   ; ELSE INITIALIZE CLOCK
19656                            <1> RTC_65:
19657 00005AF1 88F4              <1>       mov   ah, dh              ; GET SECONDS BYTE
19658 00005AF3 B001              <1>       mov   al, CMOS_SEC_ALARM  ; ADDRESS THE SECONDS ALARM REGISTER
19659 00005AF5 E818010000        <1>       call  CMOS_WRITE          ; INSERT SECONDS
19660 00005AFA 88CC              <1>       mov   ah, cl              ; GET MINUTES PARAMETER
19661 00005AFC B003              <1>       mov   al, CMOS_MIN_ALARM  ; ADDRESS MINUTES ALARM REGISTER
19662 00005AFE E80F010000        <1>       call  CMOS_WRITE          ; INSERT MINUTES
19663 00005B03 88EC              <1>       mov   ah, ch              ; GET HOURS PARAMETER
19664 00005B05 B005              <1>       mov   al, CMOS_HR_ALARM   ; ADDRESS HOUR ALARM REGISTER
19665 00005B07 E806010000        <1>       call  CMOS_WRITE          ; INSERT HOURS
19666 00005B0C E4A1              <1>       in    al, INTB01          ; READ SECOND INTERRUPT MASK REGISTER
19667 00005B0E 24FE              <1>       and   al, 0FEh            ; ENABLE ALARM TIMER BIT (CY= 0)
19668 00005B10 E6A1              <1>       out   INTB01, al          ; WRITE UPDATED MASK
19669                            <1>       ;mov  al, CMOS_REG_B            ; ADDRESS ALARM REGISTER
```

```
19670                                  <1>       ;mov   ah, al
19671 00005B12 66B80B0B                <1>       mov    ax, CMOS_REG_B * 257
19672 00005B16 E8DE000000              <1>       call   CMOS_READ           ; READ CURRENT ALARM REGISTER
19673 00005B1B 247F                    <1>       and    al, 07Fh            ; ENSURE SET BIT TURNED OFF
19674 00005B1D 0C20                    <1>       or     al, 20h                  ; TURN ON ALARM ENABLE
19675 00005B1F 86E0                    <1>       xchg   ah, al              ; MOVE MASK TO OUTPUT REGISTER
19676 00005B21 E8EC000000              <1>       call   CMOS_WRITE          ; WRITE NEW ALARM MASK
19677 00005B26 F8                      <1>       clc                        ; SET CY= 0
19678                                  <1> RTC_69:
19679 00005B27 66B80000                <1>       mov    ax, 0               ; CLEAR AX REGISTER
19680 00005B2B C3                      <1>       retn                       ; RETURN WITH RESULTS IN CARRY FLAC
19681                                  <1>
19682                                  <1> RTC_70:                           ; RESET ALARM
19683                                  <1>       ;mov   al, CMOS_REG_B            ; ADDRESS ALARM REGISTER
19684                                  <1>       ;mov   ah, al
19685 00005B2C 66B80B0B                <1>       mov    ax, CMOS_REG_B * 257      ; ADDRESS ALARM REGISTER (TO BOTH AH,AL)
19686 00005B30 E8C4000000              <1>       call   CMOS_READ           ; READ ALARM REGISTER
19687 00005B35 2457                    <1>       and    al, 57h                  ; TURN OFF ALARM ENABLE
19688 00005B37 86E0                    <1>       xchg   ah, al              ; SAVE DATA AND RECOVER ADDRESS
19689 00005B39 E8D4000000              <1>       call   CMOS_WRITE          ; RESTORE NEW VALUE
19690 00005B3E F8                      <1>       clc                        ; SET CY= 0
19691 00005B3F C3                      <1>       retn                       ; RETURN WITH NO CARRY
19692                                  <1>
19693                                  <1> RTC_STA:                  ; INITIALIZE REAL TIME CLOCK
19694                                  <1>       ;mov   al, CMOS_REG_A            ; ADDRESS REGISTER A AND LOAD DATA MASK
19695                                  <1>       ;mov   ah, 26h
19696 00005B40 66B80A26                <1>       mov    ax, (26h*100h)+CMOS_REG_A
19697 00005B44 E8C9000000              <1>       call   CMOS_WRITE          ; INITIALIZE STATUS REGISTER A
19698                                  <1>       ;mov   al, CMOS_REG_B            ; SET "SET BIT" FOR CLOCK INITIALIZATION
19699                                  <1>       ;mov   ah, 82h
19700 00005B49 66B80B82                <1>       mov    ax, (82h*100h)+CMOS_REG_B
19701 00005B4D E8C0000000              <1>       call   CMOS_WRITE          ; AND 24 HOUR MODE TO REGISTER B
19702 00005B52 B00C                    <1>       mov    al, CMOS_REG_C            ; ADDRESS REGISTER C
19703 00005B54 E8A0000000              <1>       call   CMOS_READ           ; READ REGISTER C TO INITIALIZE
19704 00005B59 B00D                    <1>       mov    al, CMOS_REG_D            ; ADDRESS REGISTER D
19705 00005B5B E899000000              <1>       call   CMOS_READ           ; READ REGISTER D TO INITIALIZE
19706 00005B60 C3                      <1>       retn
19707                                  <1>
19708                                  <1> ; 17/01/2016 (TRDOS 386 = TRDOS v2.0)
19709                                  <1>
19710                                  <1> ;--- HARDWARE INT  70 H -- ( IRQ LEVEL  8) ------------------------------------
19711                                  <1> ; ALARM INTERRUPT HANDLER (RTC)                                          :
19712                                  <1> ;       THIS ROUTINE HANDLES THE PERIODIC AND ALARM INTERRUPTS FROM THE CMOS  :
19713                                  <1> ;       TIMER. INPUT FREQUENCY IS 1.024 KHZ OR APPROXIMATELY 1024 INTERRUPTS  :
19714                                  <1> ;       EVERY SECOND FOR THE PERIODIC INTERRUPT. FOR THE ALARM FUNCTION,      :
19715                                  <1> ;       THE INTERRUPT WILL OCCUR AT THE DESIGNATED TIME.                      :
19716                                  <1> ;                                                                             :
19717                                  <1> ;       INTERRUPTS ARE ENABLED WHEN THE EVENT OR ALARM FUNCTION IS ACTIVATED. :
19718                                  <1> ;       FOR THE EVENT INTERRUPT, THE HANDLER WILL DECREMENT THE WAIT COUNTER  :
19719                                  <1> ;       AND WHEN IT EXPIRES WILL SET THE DESIGNATED LOCATION TO 80H. FOR      :
19720                                  <1> ;       THE ALARM INTERRUPT. THE USER MUST PROVIDE A ROUTINE TO INTERCEPT     :
19721                                  <1> ;       THE CORRECT ADDRESS FROM THE VECTOR TABLE INVOKED BY INTERRUPT 4AH    :
19722                                  <1> ;       PRIOR TO SETTING THE REAL TIME CLOCK ALARM (INT 1AH, AH= 06H).        :
19723                                  <1> ;-------------------------------------------------------------------------------
19724                                  <1>
19725                                  <1> RTC_A_INT: ; 07/01/2017
19726                                  <1> ;RTC_INT:                         ; ALARM INTERRUPT
19727 00005B61 1E                      <1>       push   ds                  ; LEAVE INTERRUPTS DISABLED
19728 00005B62 50                      <1>       push   eax                 ; SAVE REGISTERS
19729 00005B63 57                      <1>       push   edi
19730                                  <1> RTC_I_1:                          ; CHECK FOR SECOND INTERRUPT
19731 00005B64 66B88C8B                <1>       mov    ax, 256*(CMOS_REG_B+NMI)+CMOS_REG_C+NMI ; ALARM AND STATUS
19732 00005B68 E670                    <1>       out    CMOS_PORT, al       ; WRITE ALARM FLAG MASK ADDRESS
19733 00005B6A 90                      <1>       nop                        ; I/O DELAY
19734 00005B6B EB00                    <1>       jmp    short $+2
19735 00005B6D E471                    <1>       in     al, CMOS_DATA       ; READ AND RESET INTERRUPT REQUEST FLAGS
19736 00005B6F A860                    <1>       test   al, 01100000b       ; CHECK FOR EITHER INTERRUPT PENDING
19737 00005B71 745D                    <1>       jz     short  RTC_I_9            ; EXIT IF NOT A VALID RTC INTERRUPT
19738                                  <1>
19739 00005B73 86E0                    <1>       xchg   ah, al              ; SAVE FLAGS AND GET ENABLE ADDRESS
19740 00005B75 E670                    <1>       out    CMOS_PORT, al       ; WRITE ALARM ENABLE MASK ADDRESS
19741 00005B77 90                      <1>       nop                        ; I/O DELAY
19742 00005B78 EB00                    <1>       jmp    short $+2
19743 00005B7A E471                    <1>       in     al, CMOS_DATA       ; READ CURRENT ALARM ENABLE MASK
19744 00005B7C 20E0                    <1>       and    al, ah              ; ALLOW ONLY SOURCES THAT ARE ENABLED
19745 00005B7E A840                    <1>       test   al, 01000000b       ; CHECK FOR PERIODIC INTERRUPT
19746 00005B80 743B                    <1>       jz     short RTC_I_5       ; SKIP IF NOT A PERIODIC INTERRUPT
19747                                  <1>
19748                                  <1> ;-----       DECREMENT WAIT COUNT BY INTERRUPT INTERVAL
19749                                  <1>
19750 00005B82 66BF1000                <1>       mov    di, KDATA           ; kernel data segment
19751 00005B86 8EDF                    <1>       mov    ds, di
19752                                  <1>
19753 00005B88 812D[98520100]D003-     <1>       sub    dword [RTC_LH], 976 ; DECREMENT COUNT BY 1/1024
19754 00005B90 0000                    <1>
19755 00005B92 7329                    <1>       jnc    short RTC_I_5       ; SKIP TILL 32 BIT WORD LESS THAN ZERO
19756                                  <1>
19757                                  <1> ;-----       TURN OFF PERIODIC INTERRUPT ENABLE
19758                                  <1>
19759 00005B94 6650                    <1>       push   ax                  ; SAVE INTERRUPT FLAG MASK
19760 00005B96 66B88B8B                <1>       mov    ax, 257*(CMOS_REG_B+NMI) ; INTERRUPT ENABLE REGISTER
19761 00005B9A E670                    <1>       out    CMOS_PORT, al       ; WRITE ADDRESS TO CMOS CLOCK
19762 00005B9C 90                      <1>       nop                        ; I/O DELAY
19763 00005B9D EB00                    <1>       jmp    short $+2
19764 00005B9F E471                    <1>       in     al, CMOS_DATA       ; READ CURRENT ENABLES
19765 00005BA1 24BF                    <1>       and    al, 0BFh            ; TURN OFF PIE
19766 00005BA3 86C4                    <1>       xchg   al, ah              ; GET CMOS ADDRESS AND SAVE VALUE
19767 00005BA5 E670                    <1>       out    CMOS_PORT, al       ; ADDRESS REGISTER B
19768 00005BA7 86C4                    <1>       xchg   al, ah              ; GET NEW INTERRUPT ENABLE MASK
19769 00005BA9 E671                    <1>       out    CMOS_DATA, al       ; SET MASK IN INTERRUPT ENABLE REGISTER
19770 00005BAB C605[9C520100]00        <1>       mov    byte [RTC_WAIT_FLAG], 0   ; SET FUNCTION ACTIVE FLAG OFF
19771 00005BB2 8B3D[9D520100]          <1>       mov    edi, [USER_FLAG]    ; SET UP (DS:DI) TO POINT TO USER FLAG
19772 00005BB8 C60780                  <1>       mov    byte [edi], 80h           ; TURN ON USERS FLAG
```

```
19773 00005BBB 6658              <1>       pop    ax                ; GET INTERRUPT SOURCE BACK
19774                            <1> RTC_I_5:
19775 00005BBD A820              <1>       test   al, 00100000b     ; TEST FOR ALARM INTERRUPT
19776 00005BBF 740D              <1>       jz     short RTC_I_7     ; SKIP USER INTERRUPT CALL IF NOT ALARM
19777                            <1>
19778 00005BC1 B00D              <1>       mov    al, CMOS_REG_D           ; POINT TO DEFAULT READ ONLY REGISTER
19779 00005BC3 E670              <1>       out    CMOS_PORT, al     ; ENABLE NMI AND CMOS ADDRESS TO DEFAULT
19780 00005BC5 FB                <1>       sti                      ; INTERRUPTS BACK ON NOW
19781 00005BC6 52                <1>       push   edx
19782 00005BC7 E809980000        <1>       call   INT4Ah            ; TRANSFER TO USER ROUTINE
19783 00005BCC 5A                <1>       pop    edx
19784 00005BCD FA                <1>       cli                      ; BLOCK INTERRUPT FOR RETRY
19785                            <1> RTC_I_7:                      ; RESTART ROUTINE TO HANDLE DELAYED
19786 00005BCE EB94              <1>       jmp    short RTC_I_1     ;  ENTRY AND SECOND EVENT BEFORE DONE
19787                            <1>
19788                            <1> RTC_I_9:                      ; EXIT - NO PENDING INTERRUPTS
19789 00005BD0 B00D              <1>       mov    al, CMOS_REG_D           ; POINT TO DEFAULT READ ONLY REGISTER
19790 00005BD2 E670              <1>       out    CMOS_PORT, al     ; ENABLE NMI AND CMOS ADDRESS TO DEFAULT
19791 00005BD4 B020              <1>       mov    al, EOI                  ; END OF INTERRUPT MASK TO 8259 - 2
19792 00005BD6 E6A0              <1>       out    INTB00, al        ; TO 8259 - 2
19793 00005BD8 E620              <1>       out    INTA00,      al          ; TO 8259 - 1
19794 00005BDA 5F                <1>       pop    edi               ; RESTORE REGISTERS
19795 00005BDB 58                <1>       pop    eax
19796 00005BDC 1F                <1>       pop    ds
19797 00005BDD CF                <1>       iretd                    ; END OF INTERRUPT
19798                            <1>
19799                            <1>
19800                            <1>       ; 29/05/2016 - TRDOS 386 (TRDOS v2.0)
19801                            <1>       ; 22/08/2014 (Retro UNIX 386 v1)
19802                            <1>       ; IBM PC/AT BIOS source code ----- 10/06/85 (bios2.asm)
19803                            <1> UPD_IPR:                      ; WAIT TILL UPDATE NOT IN PROGRESS
19804 00005BDE 51                <1>       push   ecx
19805                            <1>
19806                            <1>       ; 29/05/2016
19807 00005BDF B968110000        <1>       mov    ecx, ((1984+244)*4)/2     ; AWARD BIOS 1999, ATIME.ASM
19808                            <1>                                ; 'WAITCPU_CK_UD_STAT'
19809                            <1>                                ; (244Us + 1984Us)
19810                            <1>                                ; (assume each read takes
19811                            <1>                                ;  2 microseconds).
19812                            <1>       ;mov   ecx, 65535
19813                            <1>              ;mov cx, 800       ; SET TIMEOUT LOOP COUNT (= 800)
19814                            <1> UPD_10:
19815 00005BE4 B00A              <1>       mov    al, CMOS_REG_A           ; ADDRESS STATUS REGISTER A
19816 00005BE6 FA                <1>       cli                      ; NO TIMER INTERRUPTS DURING UPDATES
19817 00005BE7 E80D000000        <1>       call   CMOS_READ         ; READ UPDATE IN PROCESS FLAG
19818 00005BEC A880              <1>       test   al, 80h                  ; IF UIP BIT IS ON ( CANNOT READ TIME )
19819 00005BEE 7406              <1>       jz     short UPD_90       ; EXIT WITH CY= 0 IF CAN READ CLOCK NOW
19820 00005BF0 FB                <1>       sti                      ; ALLOW INTERRUPTS WHILE WAITING
19821 00005BF1 E2F1              <1>       loop   UPD_10            ; LOOP TILL READY OR TIMEOUT
19822 00005BF3 31C0              <1>       xor    eax, eax          ; CLEAR RESULTS IF ERROR
19823                            <1>              ; xor ax, ax
19824 00005BF5 F9                <1>       stc                      ; SET CARRY FOR ERROR
19825                            <1> UPD_90:
19826 00005BF6 59                <1>       pop    ecx               ; RESTORE CALLERS REGISTER
19827 00005BF7 FA                <1>       cli                      ; INTERRUPTS OFF DURING SET
19828 00005BF8 C3                <1>       retn                     ; RETURN WITH CY FLAG SET
19829                            <1>
19830                            <1>
19831                            <1>       ; 29/05/2016 - TRDOS 386 (TRDOS v2.0)
19832                            <1>       ; 22/08/2014 (Retro UNIX 386 v1)
19833                            <1>       ; IBM PC/AT BIOS source code ----- 10/06/85 (test4.asm)
19834                            <1>
19835                            <1> ;--- CMOS_READ ------------------------------------------------------------
19836                            <1> ;           READ BYTE FROM CMOS_SYSTEM CLOCK CONFIGURATION TABLE       :
19837                            <1> ;                                                                      :
19838                            <1> ; INPUT: (AL)=     CMOS_TABLE ADDRESS TO BE READ                         :
19839                            <1> ;           BIT    7 = 0 FOR NMI ENABLED AND 1 FOR NMI DISABLED ON EXIT    :
19840                            <1> ;           BITS 6-0 = ADDRESS OF TABLE LOCATION TO READ               :
19841                            <1> ;                                                                      :
19842                            <1> ; OUTPUT: (AL)     VALUE AT LOCATION (AL) MOVED INTO (AL). IF BIT 7 OF (AL) WAS   :
19843                            <1> ;           ON THEN NMI LEFT DISABLED, DURING THE CMOS READ BOTH NMI AND   :
19844                            <1> ;           NORMAL INTERRUPTS ARE DISABLED TO PROTECT CMOS DATA INTEGRITY. :
19845                            <1> ;           THE CMOS ADDRESS REGISTER IS POINTED TO A DEFAULT VALUE AND    :
19846                            <1> ;           THE INTERRUPT FLAG RESTORED TO THE ENTRY STATE ON RETURN.      :
19847                            <1> ;           ONLY THE (AL) REGISTER AND THE NMI STATE IS CHANGED.       :
19848                            <1> ;--------------------------------------------------------------------------
19849                            <1>
19850                            <1> CMOS_READ:
19851 00005BF9 9C                <1>       pushf                    ; SAVE INTERRUPT ENABLE STATUS AND FLAGS
19852 00005BFA D0C0              <1>       rol    al, 1             ; MOVE NMI BIT TO LOW POSITION
19853 00005BFC F9                <1>       stc                      ; FORCE NMI BIT ON IN CARRY FLAG
19854 00005BFD D0D8              <1>       rcr    al, 1             ; HIGH BIT ON TO DISABLE NMI - OLD IN CY
19855 00005BFF FA                <1>       cli                      ; DISABLE INTERRUPTS
19856 00005C00 E670              <1>       out    CMOS_PORT, al     ; ADDRESS LOCATION AND DISABLE NMI
19857                            <1>       ; 29/05/2016
19858                            <1>       ;nop                     ; I/O DELAY
19859 00005C02 E6EB              <1>       out    0ebh,al      ; NEWIODELAY ; AWARD BIOS 1999, ATIME.ASM
19860                            <1>       ;
19861 00005C04 E471              <1>       in     al, CMOS_DATA     ; READ THE REQUESTED CMOS LOCATION
19862 00005C06 6650              <1>       push   ax                ; SAVE (AH) REGISTER VALUE AND CMOS BYTE
19863                            <1>       ; 15/03/2015 ; IBM PC/XT Model 286 BIOS source code
19864                            <1>                    ; ----- 10/06/85 (test4.asm)
19865 00005C08 B01E              <1>       mov    al, CMOS_SHUT_DOWN*2      ; GET ADDRESS OF DEFAULT LOCATION
19866                            <1>       ;mov   al, CMOS_REG_D*2  ; GET ADDRESS OF DEFAULT LOCATION
19867 00005C0A D0D8              <1>       rcr    al, 1             ; PUT ORIGINAL NMI MASK BIT INTO ADDRESS
19868 00005C0C E670              <1>       out    CMOS_PORT, al     ; SET DEFAULT TO READ ONLY REGISTER
19869 00005C0E 6658              <1>       pop    ax                ; RESTORE (AH) AND (AL), CMOS BYTE
19870 00005C10 9D                <1>       popf
19871 00005C11 C3                <1>       retn                     ; RETURN WITH FLAGS RESTORED
19872                            <1>
19873                            <1> ; 17/01/2016 (TRDOS 386 = TRDOS v2.0)
19874                            <1>
19875                            <1> ;--- CMOS_WRITE -----------------------------------------------------------
```

```
                        <1> ;     WRITE BYTE TO CMOS SYSTEM CLOCK CONFIGURATION TABLE              :
                        <1> ;                                                                      :
                        <1> ; INPUT: (AL)=     CMOS TABLE ADDRESS TO BE WRITTEN TO                      :
                        <1> ;           BIT    7 = 0 FOR NMI ENABLED AND 1 FOR NMI DISABLED ON EXIT    :
                        <1> ;           BITS 6-0 = ADDRESS OF TABLE LOCATION TO WRITE                  :
                        <1> ;      (AH)= NEW VALUE TO BE PLACED IN THE ADDRESSED TABLE LOCATION          :
                        <1> ;                                                                          :
                        <1> ; OUTPUT:    VALUE IN (AH) PLACED IN LOCATION (AL) WITH NMI LEFT DISABLED   :
                        <1> ;           IF BIT 7 OF (AL) IS ON, DURING THE CMOS UPDATE BOTH NMI AND    :
                        <1> ;           NORMAL INTERRUPTS ARE DISABLED TO PROTECT CMOS DATA INTEGRITY.  :
                        <1> ;           THE CMOS ADDRESS REGISTER IS POINTED TO A DEFAULT VALUE AND     :
                        <1> ;           THE INTERRUPT FLAG RESTORED TO THE ENTRY STATE ON RETURN.       :
                        <1> ;           ONLY THE CMOS LOCATION AND THE NMI STATE IS CHANGED.           :
                        <1> ;-------------------------------------------------------------------------------
                        <1>
                        <1> CMOS_WRITE:                     ; WRITE (AH) TO LOCATION (AL)
00005C12 9C             <1>     pushf                       ; SAVE INTERRUPT ENABLE STATUS AND FLAGS
00005C13 6650           <1>     push   ax                   ; SAVE WORK REGISTER VALUES
00005C15 D0C0           <1>     rol    al, 1                ; MOVE NMI BIT TO LOW POSITION
00005C17 F9             <1>     stc                         ; FORCE NMI BIT ON IN CARRY FLAG
00005C18 D0D8           <1>     rcr    al, 1                ; HIGH BIT ON TO DISABLE NMI - OLD IN CY
00005C1A FA             <1>     cli                         ; DISABLE INTERRUPTS
00005C1B E670           <1>     out    CMOS_PORT, al        ; ADDRESS LOCATION AND DISABLE NMI
00005C1D 88E0           <1>     mov    al, ah               ; GET THE DATA BYTE TO WRITE
00005C1F E671           <1>     out    CMOS_DATA, al        ; PLACE IN REQUESTED CMOS LOCATION
00005C21 B01E           <1>     mov    al, CMOS_SHUT_DOWN*2      ; GET ADDRESS OF DEFAULT LOCATION
                        <1>     ;mov   al, CMOS_REG_D*2     ; GET ADDRESS OF DEFAULT LOCATION
00005C23 D0D8           <1>     rcr    al, 1                ; PUT ORIGINAL NMI MASK BIT INTO ADDRESS
00005C25 E670           <1>     out    CMOS_PORT, al        ; SET DEFAULT TO READ ONLY REGISTER
00005C27 90             <1>     nop                         ; I/O DELAY
00005C28 E471           <1>     in     al, CMOS_DATA        ; OPEN STANDBY LATCH
00005C2A 6658           <1>     pop    ax                   ; RESTORE WORK REGISTERS
00005C2C 9D             <1>     popf
00005C2D C3             <1>     retn
                        <1>
                        <1> ; /// End Of TIMER FUNCTIONS ///

00005C2E 90<rept>           Align 16

                            gdt:  ; Global Descriptor Table
                                  ; (30/07/2015, conforming cs)
                                  ; (26/03/2015)
                                  ; (24/03/2015, tss)
                                  ; (19/03/2015)
                                  ; (29/12/2013)
                                  ;
00005C30 0000000000000000         dw 0, 0, 0, 0       ; NULL descriptor
                                  ; 18/08/2014
                                                ; 8h kernel code segment, base = 00000000h
                                  ;dw 0FFFFh, 0, 9E00h, 00CFh      ; KCODE  ; 30/12/2016
00005C38 FFFF0000009ACF00         dw 0FFFFh, 0, 9A00h, 00CFh; KCODE
                                                ; 10h kernel data segment, base = 00000000h
00005C40 FFFF00000092CF00         dw 0FFFFh, 0, 9200h, 00CFh; KDATA
                                                ; 1Bh user code segment, base address = 400000h ; CORE
                                  ;dw 0FBFFh, 0, 0FE40h, 00CFh     ; UCODE  ; 30/12/2016
00005C48 FFFB000040FACF00         dw 0FBFFh, 0, 0FA40h, 00CFh      ; UCODE
                                                ; 23h user data segment, base address = 400000h ; CORE
00005C50 FFFB000040F2CF00         dw 0FBFFh, 0, 0F240h, 00CFh      ; UDATA
                                                ; Task State Segment
00005C58 6700                     dw 0067h ; Limit = 103 ; (104-1, tss size = 104 byte,
                                                ;  no IO permission in ring 3)
                                  gdt_tss0:
00005C5A 0000                     dw 0  ; TSS base address, bits 0-15
                                  gdt_tss1:
00005C5C 00                       db 0  ; TSS base address, bits 16-23
                                                ; 49h
00005C5D E9                       db 11101001b ; E9h => P=1/DPL=11/0/1/0/B/1 --> B = Task is busy (1)
00005C5E 00                       db 0 ; G/0/0/AVL/LIMIT=0000 ; (Limit bits 16-19 = 0000) (G=0, 1 byte)
                                  gdt_tss2:
00005C5F 00                       db 0  ; TSS base address, bits 24-31

                                  gdt_end:
                                        ;; 9Eh = 1001 1110b (GDT byte 5) P=1/DPL=00/1/TYPE=1110,
                                                                ;; Type= 1 (code)/C=1/R=1/A=0
                                              ; P= Present, DPL=0=ring 0,  1= user (0= system)
                                              ; 1= Code C= Conforming, R= Readable, A = Accessed

                                        ;; 9Ah = 1001 1010b (GDT byte 5) P=1/DPL=00/1/TYPE=1010,
                                                                ;; Type= 1 (code)/C=0/R=1/A=0
                                              ; P= Present, DPL=0=ring 0,  1= user (0= system)
                                              ; 1= Code C= non-Conforming, R= Readable, A = Accessed

                                        ;; 92h = 1001 0010b (GDT byte 5) P=1/DPL=00/1/TYPE=1010,
                                                                ;; Type= 0 (data)/E=0/W=1/A=0
                                              ; P= Present, DPL=0=ring 0,  1= user (0= system)
                                              ; 0= Data E= Expansion direction (1= down, 0= up)
                                              ; W= Writeable, A= Accessed

                                        ;; FEh = 1111 1110b (GDT byte 5) P=1/DPL=11/1/TYPE=1110,
                                                                ;; Type= 1 (code)/C=1/R=1/A=0
                                              ; P= Present, DPL=3=ring 3,  1= user (0= system)
                                              ; 1= Code C= Conforming, R= Readable, A = Accessed

                                        ;; FAh = 1111 1010b (GDT byte 5) P=1/DPL=11/1/TYPE=1010,
                                                                ;; Type= 1 (code)/C=0/R=1/A=0
                                              ; P= Present, DPL=3=ring 3,  1= user (0= system)
                                              ; 1= Code C= non-Conforming, R= Readable, A = Accessed

                                        ;; F2h = 1111 0010b (GDT byte 5) P=1/DPL=11/1/TYPE=0010,
                                                                ;; Type= 0 (data)/E=0/W=1/A=0
                                              ; P= Present, DPL=3=ring 3,  1= user (0= system)
                                              ; 0= Data E= Expansion direction (1= down, 0= up)
```

```
19979                                   ;; CFh = 1100 1111b (GDT byte 6) G=1/B=1/0/AVL=0, Limit=1111b (3)
19980
19981                                         ;; Limit = FFFFFh (=> FFFFFh+1= 100000h) // bits 0-15, 48-51 //
19982                                         ;       = 100000h * 1000h (G=1) = 4GB
19983                                         ;; Limit = FFBFFh (=> FFBFFh+1= FFC00h) // bits 0-15, 48-51 //
19984                                         ;       = FFC00h * 1000h (G=1) = 4GB - 4MB
19985                                         ; G= Granularity (1= 4KB), B= Big (32 bit),
19986                                         ; AVL= Available to programmers
19987
19988                                   gdtd:
19989 00005C60 2F00                             dw gdt_end - gdt - 1    ; Limit (size)
19990 00005C62 [305C0000]                       dd gdt                  ; Address of the GDT
19991
19992                                         ; 20/08/2014
19993                                   idtd:
19994 00005C66 7F02                             dw idt_end - idt - 1    ; Limit (size)
19995 00005C68 [384F0100]                       dd idt                  ; Address of the IDT
19996
19997                                   ; 20/02/2017
19998                                   ;;; 11/03/2015
19999                                   %include 'diskdata.s'     ; DISK (BIOS) DATA (initialized)
20000                                 <1> ; ********************************************************************************
20001                                 <1> ; TRDOS386.ASM (TRDOS 386 Kernel) - v2.0.0 - diskdata.s
20002                                 <1> ; --------------------------------------------------------------------------------
20003                                 <1> ; Last Update: 24/01/2016
20004                                 <1> ; --------------------------------------------------------------------------------
20005                                 <1> ; Beginning: 24/01/2016
20006                                 <1> ; --------------------------------------------------------------------------------
20007                                 <1> ; Assembler: NASM version 2.11 (trdos386.s)
20008                                 <1> ; --------------------------------------------------------------------------------
20009                                 <1> ; Turkish Rational DOS
20010                                 <1> ; Operating System Project v2.0 by ERDOGAN TAN (Beginning: 04/01/2016)
20011                                 <1> ;
20012                                 <1> ; Derived from 'Retro UNIX 386 Kernel - v0.2.1.0' source code by Erdogan Tan
20013                                 <1> ; diskdata.inc (11/03/2015)
20014                                 <1> ;
20015                                 <1> ; Derived from 'IBM PC-XT-286' BIOS source code (1986)
20016                                 <1> ; ********************************************************************************
20017                                 <1>
20018                                 <1> ; Retro UNIX 386 v1 Kernel - DISKDATA.INC
20019                                 <1> ; Last Modification: 11/03/2015
20020                                 <1> ;     (Initialized Disk Parameters Data section for 'DISKIO.INC')
20021                                 <1> ;
20022                                 <1>
20023                                 <1> ;---------------------------------------
20024                                 <1> ;     80286 INTERRUPT LOCATIONS :
20025                                 <1> ;     REFERENCED BY POST & BIOS :
20026                                 <1> ;---------------------------------------
20027                                 <1>
20028 00005C6C [CF5C0000]           <1> DISK_POINTER:      dd     MD_TBL6             ; Pointer to Diskette Parameter Table
20029                                 <1>
20030                                 <1> ; IBM PC-XT Model 286 source code ORGS.ASM (06/10/85) - 14/12/2014
20031                                 <1> ;--------------------------------------------------------------
20032                                 <1> ; DISK_BASE                                          :
20033                                 <1> ;     THIS IS THE SET OF PARAMETERS REQUIRED FOR          :
20034                                 <1> ;     DISKETTE OPERATION. THEY ARE POINTED AT BY THE             :
20035                                 <1> ;     DATA VARIABLE @DISK_POINTER. TO MODIFY THE PARAMETERS,     :
20036                                 <1> ;     BUILD ANOTHER PARAMETER BLOCK AND POINT AT IT       :
20037                                 <1> ;--------------------------------------------------------------
20038                                 <1>
20039                                 <1> ;DISK_BASE:
20040                                 <1> ;     DB     11011111B    ; SRT=D, HD UNLOAD=0F - 1ST SPECIFY BYTE
20041                                 <1> ;     DB     2            ; HD LOAD=1, MODE=DMA - 2ND SPECIFY BYTE
20042                                 <1> ;     DB     MOTOR_WAIT   ; WAIT TIME AFTER OPERATION TILL MOTOR OFF
20043                                 <1> ;     DB     2            ; 512 BYTES/SECTOR
20044                                 <1> ;     ;DB    15           ; EOT (LAST SECTOR ON TRACK)
20045                                 <1> ;     db     18           ; (EOT for 1.44MB diskette)
20046                                 <1> ;     DB     01BH         ; GAP LENGTH
20047                                 <1> ;     DB     0FFH         ; DTL
20048                                 <1> ;     ;DB    054H         ; GAP LENGTH FOR FORMAT
20049                                 <1> ;     db     06ch         ; (for 1.44MB dsikette)
20050                                 <1> ;     DB     0F6H         ; FILL BYTE FOR FORMAT
20051                                 <1> ;     DB     15           ; HEAD SETTLE TIME (MILLISECONDS)
20052                                 <1> ;     DB     8            ; MOTOR START TIME (1/8 SECONDS)
20053                                 <1>
20054                                 <1> ;---------------------------------------
20055                                 <1> ;     ROM BIOS DATA AREAS       :
20056                                 <1> ;---------------------------------------
20057                                 <1>
20058                                 <1> ;DATA       SEGMENT AT 40H            ; ADDRESS= 0040:0000
20059                                 <1>
20060                                 <1> ;@EQUIP_FLAG DW    ?            ; INSTALLED HARDWARE FLAGS
20061                                 <1>
20062                                 <1> ;---------------------------------------
20063                                 <1> ;     DISKETTE DATA AREAS       :
20064                                 <1> ;---------------------------------------
20065                                 <1>
20066                                 <1> ;@SEEK_STATUS      DB     ?            ; DRIVE RECALIBRATION STATUS
20067                                 <1> ;                                  ; BIT 3-0 = DRIVE 3-0 RECALIBRATION
20068                                 <1> ;                                  ; BEFORE NEXT SEEK IF BIT IS = 0
20069                                 <1> ;@MOTOR_STATUS     DB     ?            ; MOTOR STATUS
20070                                 <1> ;                                  ; BIT 3-0 = DRIVE 3-0 CURRENTLY RUNNING
20071                                 <1> ;                                  ; BIT 7 = CURRENT OPERATION IS A WRITE
20072                                 <1> ;@MOTOR_COUNT      DB     ?            ; TIME OUT COUNTER FOR MOTOR(S) TURN OFF
20073                                 <1> ;@DSKETTE_STATUS DB       ?            ; RETURN CODE STATUS BYTE
20074                                 <1> ;                                  ; CMD_BLOCK  IN STACK FOR DISK OPERATION
20075                                 <1> ;@NEC_STATUS DB    7 DUP(?)     ; STATUS BYTES FROM DISKETTE OPERATION
20076                                 <1>
20077                                 <1> ;---------------------------------------
20078                                 <1> ;     POST AND BIOS WORK DATA AREA     :
20079                                 <1> ;---------------------------------------
20080                                 <1>
20081                                 <1> ;@INTR_FLAG  DB    ?            ; FLAG INDICATING AN INTERRUPT HAPPENED
```

```
20082                                  <1>
20083                                  <1> ;----------------------------------------
20084                                  <1> ;     TIMER DATA AREA          :
20085                                  <1> ;----------------------------------------
20086                                  <1>
20087                                  <1> ; 17/12/2014  (IRQ 0 - INT 08H)
20088                                  <1> ;TIMER_LOW   equ    46Ch         ; Timer ticks (counter)  @ 40h:006Ch
20089                                  <1> ;TIMER_HIGH  equ    46Eh         ; (18.2 timer ticks per second)
20090                                  <1> ;TIMER_OFL   equ    470h         ; Timer - 24 hours flag  @ 40h:0070h
20091                                  <1>
20092                                  <1> ;----------------------------------------
20093                                  <1> ;     ADDITIONAL MEDIA DATA           :
20094                                  <1> ;----------------------------------------
20095                                  <1>
20096                                  <1> ;@LASTRATE   DB     ?            ; LAST DISKETTE DATA RATE SELECTED
20097                                  <1> ;@DSK_STATE  DB     ?            ; DRIVE 0 MEDIA STATE
20098                                  <1> ;            DB     ?            ; DRIVE 1 MEDIA STATE
20099                                  <1> ;            DB     ?            ; DRIVE 0 OPERATION START STATE
20100                                  <1> ;            DB     ?            ; DRIVE 1 OPERATION START STATE
20101                                  <1> ;@DSK_TRK    DB     ?            ; DRIVE 0 PRESENT CYLINDER
20102                                  <1> ;            DB     ?            ; DRIVE 1 PRESENT CYLINDER
20103                                  <1>
20104                                  <1> ;DATA        ENDS                ; END OF BIOS DATA SEGMENT
20105                                  <1>
20106                                  <1> ;--------------------------------------------------------
20107                                  <1> ;     DRIVE TYPE TABLE                      :
20108                                  <1> ;--------------------------------------------------------
20109                                  <1>              ; 16/02/2015 (unix386.s, 32 bit modifications)
20110                                  <1> DR_TYPE:
20111 00005C70 01                      <1>              DB     01           ;DRIVE TYPE, MEDIA TABLE
20112                                  <1>                  ;DW      MD_TBL1
20113 00005C71 [8E5C0000]              <1>              dd     MD_TBL1
20114 00005C75 82                      <1>              DB     02+BIT7ON
20115                                  <1>              ;DW      MD_TBL2
20116 00005C76 [9B5C0000]              <1>                 dd      MD_TBL2
20117 00005C7A 02                      <1> DR_DEFAULT:  DB     02
20118                                  <1>                 ;DW      MD_TBL3
20119 00005C7B [A85C0000]              <1>              dd      MD_TBL3
20120 00005C7F 03                      <1>              DB     03
20121                                  <1>                 ;DW      MD_TBL4
20122 00005C80 [B55C0000]              <1>              dd      MD_TBL4
20123 00005C84 84                      <1>              DB     04+BIT7ON
20124                                  <1>                 ;DW      MD_TBL5
20125 00005C85 [C25C0000]              <1>              dd      MD_TBL5
20126 00005C89 04                      <1>              DB     04
20127                                  <1>                 ;DW      MD_TBL6
20128 00005C8A [CF5C0000]              <1>              dd      MD_TBL6
20129                                  <1> DR_TYPE_E       equ $                   ; END OF TABLE
20130                                  <1> ;DR_CNT           EQU    (DR_TYPE_E-DR_TYPE)/3
20131                                  <1> DR_CNT            equ    (DR_TYPE_E-DR_TYPE)/5
20132                                  <1> ;--------------------------------------------------------
20133                                  <1> ;     MEDIA/DRIVE PARAMETER TABLES                  :
20134                                  <1> ;--------------------------------------------------------
20135                                  <1> ;--------------------------------------------------------
20136                                  <1> ;     360 KB MEDIA IN 360 KB DRIVE                  :
20137                                  <1> ;--------------------------------------------------------
20138                                  <1> MD_TBL1:
20139 00005C8E DF                      <1>      DB     11011111B    ; SRT=D, HD UNLOAD=0F - 1ST SPECIFY BYTE
20140 00005C8F 02                      <1>      DB     2            ; HD LOAD=1, MODE=DMA - 2ND SPECIFY BYTE
20141 00005C90 25                      <1>      DB     MOTOR_WAIT   ; WAIT TIME AFTER OPERATION TILL MOTOR OFF
20142 00005C91 02                      <1>      DB     2            ; 512 BYTES/SECTOR
20143 00005C92 09                      <1>      DB     09           ; EOT (LAST SECTOR ON TRACK)
20144 00005C93 2A                      <1>      DB     02AH         ; GAP LENGTH
20145 00005C94 FF                      <1>      DB     0FFH         ; DTL
20146 00005C95 50                      <1>      DB     050H         ; GAP LENGTH FOR FORMAT
20147 00005C96 F6                      <1>      DB     0F6H         ; FILL BYTE FOR FORMAT
20148 00005C97 0F                      <1>      DB     15           ; HEAD SETTLE TIME (MILLISECONDS)
20149 00005C98 08                      <1>      DB     8            ; MOTOR START TIME (1/8 SECONDS)
20150 00005C99 27                      <1>      DB     39           ; MAX. TRACK NUMBER
20151 00005C9A 80                      <1>      DB     RATE_250     ; DATA TRANSFER RATE
20152                                  <1> ;--------------------------------------------------------
20153                                  <1> ;     360 KB MEDIA IN 1.2 MB DRIVE                  :
20154                                  <1> ;--------------------------------------------------------
20155                                  <1> MD_TBL2:
20156 00005C9B DF                      <1>      DB     11011111B    ; SRT=D, HD UNLOAD=0F - 1ST SPECIFY BYTE
20157 00005C9C 02                      <1>      DB     2            ; HD LOAD=1, MODE=DMA - 2ND SPECIFY BYTE
20158 00005C9D 25                      <1>      DB     MOTOR_WAIT   ; WAIT TIME AFTER OPERATION TILL MOTOR OFF
20159 00005C9E 02                      <1>      DB     2            ; 512 BYTES/SECTOR
20160 00005C9F 09                      <1>      DB     09           ; EOT (LAST SECTOR ON TRACK)
20161 00005CA0 2A                      <1>      DB     02AH         ; GAP LENGTH
20162 00005CA1 FF                      <1>      DB     0FFH         ; DTL
20163 00005CA2 50                      <1>      DB     050H         ; GAP LENGTH FOR FORMAT
20164 00005CA3 F6                      <1>      DB     0F6H         ; FILL BYTE FOR FORMAT
20165 00005CA4 0F                      <1>      DB     15           ; HEAD SETTLE TIME (MILLISECONDS)
20166 00005CA5 08                      <1>      DB     8            ; MOTOR START TIME (1/8 SECONDS)
20167 00005CA6 27                      <1>      DB     39           ; MAX. TRACK NUMBER
20168 00005CA7 40                      <1>      DB     RATE_300     ; DATA TRANSFER RATE
20169                                  <1> ;--------------------------------------------------------
20170                                  <1> ;     1.2 MB MEDIA IN 1.2 MB DRIVE                  :
20171                                  <1> ;--------------------------------------------------------
20172                                  <1> MD_TBL3:
20173 00005CA8 DF                      <1>      DB     11011111B    ; SRT=D, HD UNLOAD=0F - 1ST SPECIFY BYTE
20174 00005CA9 02                      <1>      DB     2            ; HD LOAD=1, MODE=DMA - 2ND SPECIFY BYTE
20175 00005CAA 25                      <1>      DB     MOTOR_WAIT   ; WAIT TIME AFTER OPERATION TILL MOTOR OFF
20176 00005CAB 02                      <1>      DB     2            ; 512 BYTES/SECTOR
20177 00005CAC 0F                      <1>      DB     15           ; EOT (LAST SECTOR ON TRACK)
20178 00005CAD 1B                      <1>      DB     01BH         ; GAP LENGTH
20179 00005CAE FF                      <1>      DB     0FFH         ; DTL
20180 00005CAF 54                      <1>      DB     054H         ; GAP LENGTH FOR FORMAT
20181 00005CB0 F6                      <1>      DB     0F6H         ; FILL BYTE FOR FORMAT
20182 00005CB1 0F                      <1>      DB     15           ; HEAD SETTLE TIME (MILLISECONDS)
20183 00005CB2 08                      <1>      DB     8            ; MOTOR START TIME (1/8 SECONDS)
20184 00005CB3 4F                      <1>      DB     79           ; MAX. TRACK NUMBER
```

```
20185 00005CB4 00                <1>       DB     RATE_500     ; DATA TRANSFER RATE
20186                            <1> ;-------------------------------------------------------
20187                            <1> ;     720 KB MEDIA IN 720 KB DRIVE                  :
20188                            <1> ;-------------------------------------------------------
20189                            <1> MD_TBL4:
20190 00005CB5 DF                <1>       DB     11011111B    ; SRT=D, HD UNLOAD=0F - 1ST SPECIFY BYTE
20191 00005CB6 02                <1>       DB     2            ; HD LOAD=1, MODE=DMA - 2ND SPECIFY BYTE
20192 00005CB7 25                <1>       DB     MOTOR_WAIT   ; WAIT TIME AFTER OPERATION TILL MOTOR OFF
20193 00005CB8 02                <1>       DB     2            ; 512 BYTES/SECTOR
20194 00005CB9 09                <1>       DB     09           ; EOT (LAST SECTOR ON TRACK)
20195 00005CBA 2A                <1>       DB     02AH         ; GAP LENGTH
20196 00005CBB FF                <1>       DB     0FFH         ; DTL
20197 00005CBC 50                <1>       DB     050H         ; GAP LENGTH FOR FORMAT
20198 00005CBD F6                <1>       DB     0F6H         ; FILL BYTE FOR FORMAT
20199 00005CBE 0F                <1>       DB     15           ; HEAD SETTLE TIME (MILLISECONDS)
20200 00005CBF 08                <1>       DB     8            ; MOTOR START TIME (1/8 SECONDS)
20201 00005CC0 4F                <1>       DB     79           ; MAX. TRACK NUMBER
20202 00005CC1 80                <1>       DB     RATE_250     ; DATA TRANSFER RATE
20203                            <1> ;-------------------------------------------------------
20204                            <1> ;     720 KB MEDIA IN 1.44 MB DRIVE                 :
20205                            <1> ;-------------------------------------------------------
20206                            <1> MD_TBL5:
20207 00005CC2 DF                <1>       DB     11011111B    ; SRT=D, HD UNLOAD=0F - 1ST SPECIFY BYTE
20208 00005CC3 02                <1>       DB     2            ; HD LOAD=1, MODE=DMA - 2ND SPECIFY BYTE
20209 00005CC4 25                <1>       DB     MOTOR_WAIT   ; WAIT TIME AFTER OPERATION TILL MOTOR OFF
20210 00005CC5 02                <1>       DB     2            ; 512 BYTES/SECTOR
20211 00005CC6 09                <1>       DB     09           ; EOT (LAST SECTOR ON TRACK)
20212 00005CC7 2A                <1>       DB     02AH         ; GAP LENGTH
20213 00005CC8 FF                <1>       DB     0FFH         ; DTL
20214 00005CC9 50                <1>       DB     050H         ; GAP LENGTH FOR FORMAT
20215 00005CCA F6                <1>       DB     0F6H         ; FILL BYTE FOR FORMAT
20216 00005CCB 0F                <1>       DB     15           ; HEAD SETTLE TIME (MILLISECONDS)
20217 00005CCC 08                <1>       DB     8            ; MOTOR START TIME (1/8 SECONDS)
20218 00005CCD 4F                <1>       DB     79           ; MAX. TRACK NUMBER
20219 00005CCE 80                <1>       DB     RATE_250     ; DATA TRANSFER RATE
20220                            <1> ;-------------------------------------------------------
20221                            <1> ;     1.44 MB MEDIA IN 1.44 MB DRIVE                :
20222                            <1> ;-------------------------------------------------------
20223                            <1> MD_TBL6:
20224 00005CCF AF                <1>       DB     10101111B    ; SRT=A, HD UNLOAD=0F - 1ST SPECIFY BYTE
20225 00005CD0 02                <1>       DB     2            ; HD LOAD=1, MODE=DMA - 2ND SPECIFY BYTE
20226 00005CD1 25                <1>       DB     MOTOR_WAIT   ; WAIT TIME AFTER OPERATION TILL MOTOR OFF
20227 00005CD2 02                <1>       DB     2            ; 512 BYTES/SECTOR
20228 00005CD3 12                <1>       DB     18           ; EOT (LAST SECTOR ON TRACK)
20229 00005CD4 1B                <1>       DB     01BH         ; GAP LENGTH
20230 00005CD5 FF                <1>       DB     0FFH         ; DTL
20231 00005CD6 6C                <1>       DB     06CH         ; GAP LENGTH FOR FORMAT
20232 00005CD7 F6                <1>       DB     0F6H         ; FILL BYTE FOR FORMAT
20233 00005CD8 0F                <1>       DB     15           ; HEAD SETTLE TIME (MILLISECONDS)
20234 00005CD9 08                <1>       DB     8            ; MOTOR START TIME (1/8 SECONDS)
20235 00005CDA 4F                <1>       DB     79           ; MAX. TRACK NUMBER
20236 00005CDB 00                <1>       DB     RATE_500     ; DATA TRANSFER RATE
20237                            <1>
20238                            <1>
20239                            <1> ; << diskette.inc >>
20240                            <1> ; ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
20241                            <1> ;
20242                            <1> ;---------------------------------------
20243                            <1> ;     ROM BIOS DATA AREAS       :
20244                            <1> ;---------------------------------------
20245                            <1>
20246                            <1> ;DATA       SEGMENT AT 40H            ; ADDRESS= 0040:0000
20247                            <1>
20248                            <1> ;---------------------------------------
20249                            <1> ;     FIXED DISK DATA AREAS           :
20250                            <1> ;---------------------------------------
20251                            <1>
20252                            <1> ;DISK_STATUS1:     DB     0           ; FIXED DISK STATUS
20253                            <1> ;HF_NUM:           DB     0           ; COUNT OF FIXED DISK DRIVES
20254                            <1> ;CONTROL_BYTE:     DB     0           ; HEAD CONTROL BYTE
20255                            <1> ;@PORT_OFF   DB    ?           ;  RESERVED (PORT OFFSET)
20256                            <1>
20257                            <1> ;---------------------------------------
20258                            <1> ;     ADDITIONAL MEDIA DATA           :
20259                            <1> ;---------------------------------------
20260                            <1>
20261                            <1> ;@LASTRATE   DB     ?           ; LAST DISKETTE DATA RATE SELECTED
20262                            <1> ;HF_STATUS   DB     0           ; STATUS REGISTER
20263                            <1> ;HF_ERROR    DB     0           ; ERROR REGISTER
20264                            <1> ;HF_INT_FLAG DB     0           ; FIXED DISK INTERRUPT FLAG
20265                            <1> ;HF_CNTRL    DB     0           ; COMBO FIXED DISK/DISKETTE CARD BIT 0=1
20266                            <1> ;@DSK_STATE  DB     ?           ; DRIVE 0 MEDIA STATE
20267                            <1> ;            DB     ?           ; DRIVE 1 MEDIA STATE
20268                            <1> ;            DB     ?           ; DRIVE 0 OPERATION START STATE
20269                            <1> ;            DB     ?           ; DRIVE 1 OPERATION START STATE
20270                            <1> ;@DSK_TRK    DB     ?           ; DRIVE 0 PRESENT CYLINDER
20271                            <1> ;            DB     ?           ; DRIVE 1 PRESENT CYLINDER
20272                            <1>
20273                            <1> ;DATA       ENDS                ; END OF BIOS DATA SEGMENT
20274                            <1> ;
20275                            <1> ; ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
20276                            <1>
20277                            <1> ERR_TBL:
20278 00005CDC E0                <1>       db     NO_ERR
20279 00005CDD 024001BB          <1>       db     BAD_ADDR_MARK,BAD_SEEK,BAD_CMD,UNDEF_ERR
20280 00005CE1 04BB100A          <1>       db     RECORD_NOT_FND,UNDEF_ERR,BAD_ECC,BAD_SECTOR
20281                            <1>
20282                            <1> ; 17/12/2014 (mov ax, [cfd])
20283                            <1> ; 11/12/2014
20284 00005CE5 00                <1> cfd:        db 0               ; current floppy drive (for GET_PARM)
20285                            <1> ; 17/12/2014                   ; instead of 'DISK_POINTER'
20286 00005CE6 01                <1> pfd:        db 1               ; previous floppy drive (for GET_PARM)
20287                            <1>                                ; (initial value of 'pfd
```

```
20288                                   <1>                                     ; must be different then 'cfd' value
20289                                   <1>                                     ; to force updating/initializing
20290                                   <1>                                     ; current drive parameters)
20291 00005CE7 90                       <1> align 2
20292                                   <1>
20293 00005CE8 F001                     <1> HF_PORT:    dw      1F0h    ; Default = 1F0h
20294                                   <1>                             ; (170h)
20295 00005CEA F603                     <1> HF_REG_PORT: dw     3F6h    ; HF_PORT + 206h
20296                                   <1>
20297                                   <1> ; 05/01/2015
20298 00005CEC 00                       <1> hf_m_s:         db      0     ; (0 = Master, 1 = Slave)
20299                                   <1>
20300                                   <1> ; ****************************************************************************
20301
20302 00005CED 90                        Align 2
20303
20304                                    ; 04/11/2014 (Retro UNIX 386 v1)
20305 00005CEE 0000                     mem_1m_1k:   dw 0  ; Number of contiguous KB between
20306                                                      ; 1 and 16 MB, max. 3C00h = 15 MB.
20307 00005CF0 0000                     mem_16m_64k: dw 0  ; Number of contiguous 64 KB blocks
20308                                                ; between 16 MB and 4 GB.
20309
20310                                    ; 12/11/2014 (Retro UNIX 386 v1)
20311 00005CF2 00                       boot_drv:    db 0 ; boot drive number (physical)
20312                                    ; 24/11/2014
20313 00005CF3 00                       drv:       db 0
20314 00005CF4 00                       last_drv:    db 0 ; last hdd
20315 00005CF5 00                       hdc:         db 0  ; number of hard disk drives
20316                                                    ; (present/detected)
20317
20318                                    ; 24/11/2014 (Retro UNIX 386 v1)
20319                                    ; Physical drive type & flags
20320 00005CF6 00                       fd0_type:    db 0  ; floppy drive type
20321 00005CF7 00                       fd1_type:    db 0  ; 4 = 1.44 Mb, 80 track, 3.5" (18 spt)
20322                                                      ; 6 = 2.88 Mb, 80 track, 3.5" (36 spt)
20323                                                      ; 3 = 720 Kb, 80 track, 3.5" (9 spt)
20324                                                      ; 2 = 1.2 Mb, 80 track, 5.25" (15 spt)
20325                                                      ; 1 = 360 Kb, 40 track, 5.25" (9 spt)
20326 00005CF8 00                       hd0_type:    db 0  ; EDD status for hd0 (bit 7 = present flag)
20327 00005CF9 00                       hd1_type:    db 0  ; EDD status for hd1 (bit 7 = present flag)
20328 00005CFA 00                       hd2_type:    db 0  ; EDD status for hd2 (bit 7 = present flag)
20329 00005CFB 00                       hd3_type:    db 0  ; EDD status for hd3 (bit 7 = present flag)
20330                                                      ; bit 0 - Fixed disk access subset supported
20331                                                      ; bit 1 - Drive locking and ejecting
20332                                                      ; bit 2 - Enhanced disk drive support
20333                                                      ; bit 3 = Reserved (64 bit EDD support)
20334                                                      ; (If bit 0 is '1' Retro UNIX 386 v1
20335                                                      ; will interpret it as 'LBA ready'!)
20336
20337                                    ; 11/03/2015 - 10/07/2015
20338 00005CFC 000000000000000000-      drv.cylinders: dw 0,0,0,0,0,0,0
20339 00005D05 0000000000
20340 00005D0A 000000000000000000-      drv.heads:     dw 0,0,0,0,0,0,0
20341 00005D13 0000000000
20342 00005D18 000000000000000000-      drv.spt:       dw 0,0,0,0,0,0,0
20343 00005D21 0000000000
20344 00005D26 000000000000000000-      drv.size:      dd 0,0,0,0,0,0,0
20345 00005D2F 000000000000000000-
20346 00005D38 000000000000000000-
20347 00005D41 00
20348 00005D42 00000000000000           drv.status:    db 0,0,0,0,0,0,0
20349 00005D49 00000000000000           drv.error:     db 0,0,0,0,0,0,0
20350
20351                                    Align 2
20352
20353                                    ;;; 11/03/2015
20354                                    %include 'kybdata.s'     ; KEYBOARD (BIOS) DATA
20355                                <1> ; ****************************************************************************
20356                                <1> ; TRDOS386.ASM (TRDOS 386 Kernel) - v2.0.0 - kybdata.s
20357                                <1> ; ----------------------------------------------------------------------------
20358                                <1> ; Last Update: 17/01/2016
20359                                <1> ; ----------------------------------------------------------------------------
20360                                <1> ; Beginning: 17/01/2016
20361                                <1> ; ----------------------------------------------------------------------------
20362                                <1> ; Assembler: NASM version 2.11 (trdos386.s)
20363                                <1> ; ----------------------------------------------------------------------------
20364                                <1> ; Turkish Rational DOS
20365                                <1> ; Operating System Project v2.0 by ERDOGAN TAN (Beginning: 04/01/2016)
20366                                <1> ;
20367                                <1> ; Derived from 'Retro UNIX 386 Kernel - v0.2.1.0' source code by Erdogan Tan
20368                                <1> ; kybdata.inc (11/03/2015)
20369                                <1> ;
20370                                <1> ; Derived from 'IBM PC-XT-286' BIOS source code (1986)
20371                                <1> ; ****************************************************************************
20372                                <1>
20373                                <1> ; Retro UNIX 386 v1 Kernel - KYBDATA.INC
20374                                <1> ; Last Modification: 11/03/2015
20375                                <1> ;            (Data Section for 'KEYBOARD.INC')
20376                                <1> ;
20377                                <1> ; ///////// KEYBOARD DATA ///////////////
20378                                <1>
20379                                <1> ; 05/12/2014
20380                                <1> ; 04/12/2014 (derived from pc-xt-286 bios source code -1986-)
20381                                <1> ; 03/06/86  KEYBOARD BIOS
20382                                <1>
20383                                <1> ;--------------------------------------------------------------------------------
20384                                <1> ;     KEY IDENTIFICATION SCAN TABLES
20385                                <1> ;--------------------------------------------------------------------------------
20386                                <1>
20387                                <1> ;-----       TABLES FOR ALT CASE ------------
20388                                <1> ;-----       ALT-INPUT-TABLE
20389 00005D50 524F50514B          <1> K30:  db     82,79,80,81,75
20390 00005D55 4C4D474849          <1>       db     76,77,71,72,73            ; 10 NUMBER ON KEYPAD
```

```
20391                                   <1> ;-----      SUPER-SHIFT-TABLE
20392 00005D5A 101112131415             <1>         db  16,17,18,19,20,21   ; A-Z TYPEWRITER CHARS
20393 00005D60 161718191E1F             <1>         db  22,23,24,25,30,31
20394 00005D66 202122232425             <1>         db  32,33,34,35,36,37
20395 00005D6C 262C2D2E2F30             <1>         db  38,44,45,46,47,48
20396 00005D72 3132                     <1>         db  49,50
20397                                   <1>
20398                                   <1> ;-----      TABLE OF SHIFT KEYS AND MASK VALUES
20399                                   <1> ;-----      KEY_TABLE
20400 00005D74 52                       <1> _K6:    db      INS_KEY                 ; INSERT KEY
20401 00005D75 3A4546381D               <1>         db  CAPS_KEY,NUM_KEY,SCROLL_KEY,ALT_KEY,CTL_KEY
20402 00005D7A 2A36                     <1>           db    LEFT_KEY,RIGHT_KEY
20403                                   <1> _K6L    equ     $-_K6
20404                                   <1>
20405                                   <1> ;-----      MASK_TABLE
20406 00005D7C 80                       <1> _K7:    db      INS_SHIFT               ; INSERT MODE SHIFT
20407 00005D7D 4020100804               <1>         db  CAPS_SHIFT,NUM_SHIFT,SCROLL_SHIFT,ALT_SHIFT,CTL_SHIFT
20408 00005D82 0201                     <1>         db  LEFT_SHIFT,RIGHT_SHIFT
20409                                   <1>
20410                                   <1> ;-----      TABLES FOR CTRL CASE            ;---- CHARACTERS ------
20411 00005D84 1BFF00FFFFFF             <1> _K8:  db    27,-1,0,-1,-1,-1    ; Esc, 1, 2, 3, 4, 5
20412 00005D8A 1EFFFFFFFF1F             <1>       db    30,-1,-1,-1,-1,31   ; 6, 7, 8, 9, 0, -
20413 00005D90 FF7FFF111705             <1>       db    -1,127,-1,17,23,5   ; =, Bksp, Tab, Q, W, E
20414 00005D96 12141915090F             <1>       db    18,20,25,21,9,15    ; R, T, Y, U, I, O
20415 00005D9C 101B1D0AFF01             <1>       db    16,27,29,10,-1,1    ; P, [, ], Enter, Ctrl, A
20416 00005DA2 13040607080A             <1>       db    19,4,6,7,8,10       ; S, D, F, G, H, J
20417 00005DA8 0B0CFFFFFFFF             <1>       db    11,12,-1,-1,-1,-1   ; K, L, :, ', `, LShift
20418 00005DAE 1C1A18031602             <1>       db    28,26,24,3,22,2           ; Bkslash, Z, X, C, V, B
20419 00005DB4 0E0DFFFFFFFF             <1>       db    14,13,-1,-1,-1,-1   ; N, M, ,, ., /, RShift
20420 00005DBA 96FF20FF                 <1>       db    150,-1,' ',-1       ; *, ALT, Spc, CL
20421                                   <1>       ;                         ;----- FUNCTIONS ------
20422 00005DBE 5E5F60616263             <1>       db    94,95,96,97,98,99   ; F1 - F6
20423 00005DC4 64656667FFFF             <1>       db    100,101,102,103,-1,-1     ; F7 - F10, NL, SL
20424 00005DCA 778D848E738F             <1>       db    119,141,132,142,115,143   ; Home, Up, PgUp, -, Left, Pad5
20425 00005DD0 749075917692             <1>       db    116,144,117,145,118,146 ; Right, +, End, Down, PgDn, Ins
20426 00005DD6 93FFFFFF898A             <1>       db    147,-1,-1,-1,137,138      ; Del, SysReq, Undef, WT, F11, F12
20427                                   <1>
20428                                   <1> ;-----      TABLES FOR LOWER CASE ----------
20429 00005DDC 1B3132333435363738-      <1> K10:  db    27,'1234567890-=',8,9
20430 00005DE5 39302D3D0809             <1>
20431 00005DEB 71776572747975696F-      <1>       db    'qwertyuiop[]',13,-1,'asdfghjkl;',39
20432 00005DF4 705B5D0DFF61736466-      <1>
20433 00005DFD 67686A6B6C3B27           <1>
20434 00005E04 60FF5C7A786376626E-      <1>       db    96,-1,92,'zxcvbnm,./',-1,'*',-1,' ',-1
20435 00005E0D 6D2C2E2FFF2AFF20FF       <1>
20436                                   <1> ;-----      LC TABLE SCAN
20437 00005E16 3B3C3D3E3F               <1>       db    59,60,61,62,63            ; BASE STATE OF F1 - F10
20438 00005E1B 4041424344               <1>       db    64,65,66,67,68
20439 00005E20 FFFF                     <1>       db    -1,-1               ; NL, SL
20440                                   <1>
20441                                   <1> ;-----      KEYPAD TABLE
20442 00005E22 474849FF4BFF             <1> K15:  db    71,72,73,-1,75,-1   ; BASE STATE OF KEYPAD KEYS
20443 00005E28 4DFF4F50515253           <1>       db    77,-1,79,80,81,82,83
20444 00005E2F FFFF5C8586               <1>       db    -1,-1,92,133,134    ; SysRq, Undef, WT, F11, F12
20445                                   <1>
20446                                   <1> ;-----      TABLES FOR UPPER CASE ----------
20447 00005E34 1B21402324255E262A-      <1> K11:  db    27,'!@#$%',94,'&*()_+',8,0
20448 00005E3D 28295F2B0800             <1>
20449 00005E43 51574552545955494F-      <1>       db    'QWERTYUIOP{}',13,-1,'ASDFGHJKL:"'
20450 00005E4C 507B7D0DFF41534446-      <1>
20451 00005E55 47484A4B4C3A22           <1>
20452 00005E5C 7EFF7C5A584356424E-      <1>       db    126,-1,'|ZXCVBNM<>?',-1,'*',-1,' ',-1
20453 00005E65 4D3C3E3FFF2AFF20FF       <1>
20454                                   <1> ;-----      UC TABLE SCAN
20455 00005E6E 5455565758               <1> K12:  db    84,85,86,87,88            ; SHIFTED STATE OF F1 - F10
20456 00005E73 595A5B5C5D               <1>       db    89,90,91,92,93
20457 00005E78 FFFF                     <1>       db    -1,-1               ; NL, SL
20458                                   <1>
20459                                   <1> ;-----      NUM STATE TABLE
20460 00005E7A 3738392D3435362B31-      <1> K14:  db    '789-456+1230.'           ; NUMLOCK STATE OF KEYPAD KEYS
20461 00005E83 3233302E                 <1>
20462                                   <1>       ;
20463 00005E87 FFFF7C8788               <1>       db    -1,-1,124,135,136   ; SysRq, Undef, WT, F11, F12
20464                                   <1>
20465                                   <1> ; 26/08/2014
20466                                   <1> ; Retro UNIX 8086 v1 - UNIX.ASM (03/03/2014)
20467                                   <1> ; Derived from IBM "pc-at"
20468                                   <1> ; rombios source code (06/10/1985)
20469                                   <1> ; 'dseg.inc'
20470                                   <1>
20471                                   <1> ;--------------------------------------;
20472                                   <1> ;     SYSTEM DATA AREA          ;
20473                                   <1> ;--------------------------------------
20474 00005E8C 00                       <1> BIOS_BREAK   db    0            ; BIT 7=1 IF BREAK KEY HAS BEEN PRESSED
20475                                   <1>
20476                                   <1> ;--------------------------------------
20477                                   <1> ;     KEYBOARD DATA AREAS       ;
20478                                   <1> ;--------------------------------------
20479                                   <1>
20480 00005E8D 00                       <1> KB_FLAG           db    0           ; KEYBOARD SHIFT STATE AND STATUS FLAGS
20481 00005E8E 00                       <1> KB_FLAG_1    db    0            ; SECOND BYTE OF KEYBOARD STATUS
20482 00005E8F 00                       <1> KB_FLAG_2    db    0            ; KEYBOARD LED FLAGS
20483 00005E90 00                       <1> KB_FLAG_3    db    0            ; KEYBOARD MODE STATE AND TYPE FLAGS
20484 00005E91 00                       <1> ALT_INPUT    db    0            ; STORAGE FOR ALTERNATE KEY PAD ENTRY
20485 00005E92 [A25E0000]               <1> BUFFER_START dd    KB_BUFFER    ; OFFSET OF KEYBOARD BUFFER START
20486 00005E96 [C25E0000]               <1> BUFFER_END   dd    KB_BUFFER + 32      ; OFFSET OF END OF BUFFER
20487 00005E9A [A25E0000]               <1> BUFFER_HEAD  dd    KB_BUFFER    ; POINTER TO HEAD OF KEYBOARD BUFFER
20488 00005E9E [A25E0000]               <1> BUFFER_TAIL  dd    KB_BUFFER    ; POINTER TO TAIL OF KEYBOARD BUFFER
20489                                   <1> ; ------    HEAD = TAIL  INDICATES THAT THE BUFFER IS EMPTY
20490 00005EA2 0000<rept>               <1> KB_BUFFER    times  16 dw 0            ; ROOM FOR 16 SCAN CODE ENTRIES
20491                                   <1>
20492                                   <1> ; /// End Of KEYBOARD DATA ///
20493                                       %include 'vidata.s'       ; VIDEO (BIOS) DATA
```

```
20494                                  <1> ; ****************************************************************************
20495                                  <1> ; TRDOS386.ASM (TRDOS 386 Kernel) - v2.0.0 - vidata.s
20496                                  <1> ; ----------------------------------------------------------------------------
20497                                  <1> ; Last Update: 31/07/2016
20498                                  <1> ; ----------------------------------------------------------------------------
20499                                  <1> ; Beginning: 16/01/2016
20500                                  <1> ; ----------------------------------------------------------------------------
20501                                  <1> ; Assembler: NASM version 2.11 (trdos386.s)
20502                                  <1> ; ----------------------------------------------------------------------------
20503                                  <1> ; Turkish Rational DOS
20504                                  <1> ; Operating System Project v2.0 by ERDOGAN TAN (Beginning: 04/01/2016)
20505                                  <1> ;
20506                                  <1> ; Derived from 'Retro UNIX 386 Kernel - v0.2.1.0' source code by Erdogan Tan
20507                                  <1> ; vidata.inc (11/03/2015)
20508                                  <1> ;
20509                                  <1> ; Derived from 'IBM PC-AT' BIOS source code (1985)
20510                                  <1> ; ****************************************************************************
20511                                  <1>
20512                                  <1> ; Retro UNIX 386 v1 Kernel - VIDATA.S
20513                                  <1> ; Last Modification: 11/03/2015
20514                                  <1> ;               (Data section for 'VIDEO.INC')
20515                                  <1> ;
20516                                  <1> ; //////// VIDEO DATA ///////////////
20517                                  <1>
20518                                  <1> ;---------------------------------------
20519                                  <1> ;     VIDEO DISPLAY DATA AREA         ;
20520                                  <1> ;---------------------------------------
20521 00005EC2 03                      <1> CRT_MODE:    db     3      ; CURRENT DISPLAY MODE (TYPE)
20522 00005EC3 29                      <1> CRT_MODE_SET:      db     29h    ; CURRENT SETTING OF THE 3X8 REGISTER
20523                                  <1>                            ; (29h default setting for video mode 3)
20524                                  <1>                            ; Mode Select register Bits
20525                                  <1>                            ;   BIT 0 - 80x25 (1), 40x25 (0)
20526                                  <1>                            ;   BIT 1 - ALPHA (0), 320x200 GRAPHICS (1)
20527                                  <1>                            ;   BIT 2 - COLOR (0), BW (1)
20528                                  <1>                            ;   BIT 3 - Video Sig. ENABLE (1), DISABLE (0)
20529                                  <1>                            ;   BIT 4 - 640x200 B&W Graphics Mode (1)
20530                                  <1>                            ;   BIT 5 - ALPHA mode BLINKING (1)
20531                                  <1>                            ;   BIT 6, 7 - Not Used
20532                                  <1>
20533                                  <1> ; Mode 0 - 2Ch = 101100b  ; 40x25 text, 16 gray colors
20534                                  <1> ; Mode 1 - 28h = 101000b  ; 40x25 text, 16 fore colors, 8 back colors
20535                                  <1> ; Mode 2 - 2Dh = 101101b  ; 80x25 text, 16 gray colors
20536                                  <1> ; Mode 3 - 29h = 101001b  ; 80x25 text, 16 fore color, 8 back color
20537                                  <1> ; Mode 4 - 2Ah = 101010b  ; 320x200 graphics, 4 colors
20538                                  <1> ; Mode 5 - 2Eh = 101110b  ; 320x200 graphics, 4 gray colors
20539                                  <1> ; Mode 6 - 1Eh = 011110b  ; 640x200 graphics, 2 colors
20540                                  <1> ; Mode 7 - 29h = 101001b  ; 80x25 text, black & white colors
20541                                  <1> ; Mode & 37h = Video signal OFF
20542                                  <1>
20543                                  <1> ; 24/06/2016
20544 00005EC4 50                      <1> CRT_COLS:    db     80     ; Number of columns
20545                                  <1>
20546                                  <1> ; 01/07/2016
20547 00005EC5 00                      <1> CRT_PALETTE: db     0      ; Current palette setting
20548                                  <1>
20549                                  <1> ; 03/07/2016
20550 00005EC6 10                      <1> CHAR_HEIGHT: db     16     ; Default character height
20551 00005EC7 60                      <1> VGA_VIDEO_CTL:     db     60h    ; ROM BIOS DATA AREA Offset 87h
20552 00005EC8 F9                      <1> VGA_SWITCHES:      db     0F9h   ; Feature Bit Switches (the basic screen)
20553 00005EC9 51                      <1> VGA_MODESET_CTL: db       051h   ; Basic mode set options (VGA video flags)
20554                                  <1>                            ; ROM BIOS DATA AREA Offset 89h
20555                                  <1>                            ; Bit 7, 4 : Mode
20556                                  <1>                            ;       01 : 400-line mode
20557                                  <1>                            ; Bit 6      : Display switch enabled =  1
20558                                  <1>                            ; Bit 5      : Reserved  = 0
20559                                  <1>                            ; Bit 3      : Default palette loading
20560                                  <1>                            ;       disabled = 0
20561                                  <1>                            ; Bit 2 : Color monitor = 0
20562                                  <1>                            ; Bit 1 = Gray scale summing
20563                                  <1>                            ;       disabled = 0
20564                                  <1>                            ; Bit 0 = VGA active = 1
20565 00005ECA 19                      <1> VGA_ROWS:    db     25
20566                                  <1>
20567                                  <1> ; 16/01/2016
20568                                  <1> chr_attrib:  ; Character color/attributes for viode pages (0 to 7)
20569 00005ECB 0707070707070707       <1>       db     07h, 07h, 07h, 07h, 07h, 07h, 07h, 07h
20570                                  <1> ; 30/01/2016
20571                                  <1> vmode:
20572 00005ED3 0303030303030303       <1>       db     3,3,3,3,3,3,3,3 ; video modes for pseudo screens
20573                                  <1>
20574                                  <1> CURSOR_MODE: ; cursor start (ch) = 14, cursor end (cl) = 15
20575 00005EDB 0F0E                    <1>       db     15, 14 ; 07/07/2016 - TRDOS 386 (TRDOS v2.0)
20576                                  <1>
20577                                  <1> ;align 4
20578                                  <1> ;VGA_BASE: ; 26/07/2016
20579                                  <1> ;     dd      0B8000h  ; (Mode < 0Dh) or 0A0000h (mode >= 0Dh)
20580                                  <1>
20581 00005EDD 90                      <1> align 2
20582                                  <1>
20583                                  <1> vga_modes:
20584                                  <1>       ; 25/07/2016
20585                                  <1>       ; 09/07/2016
20586                                  <1>       ; 03/07/2016
20587                                  <1>       ; valid (implemented) video modes (>7, extension to IBM PC CGA modes)
20588 00005EDE 0302010007040506       <1>       db     03h, 02h, 01h, 00h, 07h, 04h, 05h, 06h
20589                                  <1> vga_g_modes: ; 31/07/2016
20590 00005EE6 13F0126A0D0E1011        <1>       db     13h, 0F0h, 12h, 6Ah, 0Dh, 0Eh, 10h, 11h
20591                                  <1> vga_mode_count equ $ - vga_modes
20592                                  <1> vga_g_mode_count equ $ - vga_g_modes
20593                                  <1>
20594                                  <1> vga_mode_tbl_ptr:
20595                                  <1>       ; 25/07/2016
20596 00005EEE [4E5F0000]             <1>       dd     vga_mode_03h
```

```
20597 00005EF2 [4E5F0000]          <1>       dd     vga_mode_03h ; mode 02h -> mode 03h
20598 00005EF6 [8E5F0000]          <1>       dd     vga_mode_01h
20599 00005EFA [8E5F0000]          <1>       dd     vga_mode_01h ; mode 00h -> mode 01h
20600                              <1>       ;dd    vga_mode_07h
20601 00005EFE [4E5F0000]          <1>       dd     vga_mode_03h ; mode 07h -> mode 03h
20602 00005F02 [CE5F0000]          <1>       dd     vga_mode_04h
20603 00005F06 [CE5F0000]          <1>       dd     vga_mode_04h ; mode 05h -> mode 04h
20604 00005F0A [0E600000]          <1>       dd     vga_mode_06h
20605 00005F0E [4E600000]          <1>       dd     vga_mode_13h
20606 00005F12 [8E600000]          <1>       dd     vga_mode_F0h
20607 00005F16 [CE600000]          <1>       dd     vga_mode_12h
20608 00005F1A [0E610000]          <1>       dd     vga_mode_6Ah
20609 00005F1E [4E610000]          <1>       dd     vga_mode_0Dh
20610 00005F22 [8E610000]          <1>       dd     vga_mode_0Eh
20611 00005F26 [CE610000]          <1>       dd     vga_mode_10h
20612 00005F2A [0E620000]          <1>       dd     vga_mode_11h
20613                              <1>
20614                              <1> vga_memmodel:
20615                              <1>       ; 25/07/2016
20616                              <1>       ; 07/07/2016
20617                              <1>       CTEXT  equ 0
20618                              <1>       ;MTEXT equ 1
20619                              <1>       MTEXT  equ 0 ; mode 07h -> mode 03h
20620                              <1>       CGA    equ 2
20621                              <1>       LINEAR8 equ 5
20622                              <1>       PLANAR4      equ 4
20623                              <1>       PLANAR1      equ 3
20624 00005F2E 0000000000020202   <1>       db     CTEXT, CTEXT, CTEXT, CTEXT, MTEXT, CGA, CGA, CGA
20625                              <1> vga_g_memmodel: ; 31/07/2016
20626 00005F36 0504040404040403   <1>       db     LINEAR8, PLANAR4, PLANAR4, PLANAR4, PLANAR4, PLANAR4, PLANAR4, PLANAR1
20627                              <1> ;vga_pixbits:
20628                              <1> ;      ; 25/07/2016
20629                              <1> ;      ; 08/07/2016
20630                              <1> ;      db     4, 4, 4, 4, 4, 2, 2, 1, 8, 4, 4, 4, 4, 4, 4, 1
20631                              <1> vga_dac_s:
20632 00005F3E 020202020001010103- <1>       db     2, 2, 2, 2, 0, 1, 1, 1, 3, 3, 2, 2, 1, 1, 2, 2
20633 00005F47 03020201010202      <1>
20634                              <1>
20635                              <1> vga_params:
20636                              <1>       ; 25/07/2016
20637                              <1>       ; 19/07/2016
20638                              <1>       ; 03/07/2016
20639                              <1>       ; derived from 'Plex86/Bochs VGABios' source code
20640                              <1>       ; vgabios-0.7a (2011)
20641                              <1>       ; by the LGPL VGABios Developers Team (2001-2008)
20642                              <1>       ; 'vgatables.h'
20643                              <1>       ; Oracle VirtualBox 5.0.24 VGABios Source Code
20644                              <1>       ; ('vgabios.c', 'vgatables.h', 'vgafonts.h', 'vgarom.asm')
20645                              <1>       ;
20646                              <1> vga_mode_03h:  ; mode 03h, 80*25 text, CGA colors
20647 00005F4E 5018100010         <1>       db     80, 24, 16, 00h, 10h ; tw, th-1, ch, slength (5)
20648 00005F53 00030002            <1>       db     00h, 03h, 00h, 02h ; sequ regs (4)
20649 00005F57 67                  <1>       db     67h    ; misc reg (1)
20650 00005F58 5F4F50825581BF1F    <1>       db     5Fh, 4Fh, 50h, 82h, 55h, 81h, 0BFh, 1Fh
20651 00005F60 004F                <1>       db     00h, 4Fh
20652                              <1> vga_p_cm_pos equ $ - vga_mode_03h
20653 00005F62 0D0E00000000        <1>       db     0Dh, 0Eh, 00h, 00h, 00h, 00h
20654 00005F68 9C8E8F281F96B9A3    <1>       db     9Ch, 8Eh, 8Fh, 28h, 1Fh, 96h, 0B9h, 0A3h
20655 00005F70 FF                  <1>       db     0FFh   ; crtc_regs (25)
20656 00005F71 0001020304051407    <1>       db     00h, 01h, 02h, 03h, 04h, 05h, 14h, 07h
20657 00005F79 38393A3B3C3D3E3F    <1>       db     38h, 39h, 3Ah, 3Bh, 3Ch, 3Dh, 3Eh, 3Fh
20658 00005F81 0C000F08            <1>       db     0Ch, 00h, 0Fh, 08h  ; actl regs (20)
20659 00005F85 0000000000100E0FFF  <1>       db     00h, 00h, 00h, 00h, 00h, 10h, 0Eh, 0Fh, 0FFh ; grdc regs (9)
20660                              <1> vga_mode_01h:      ; mode 01h, 40*25 text, CGA colors
20661 00005F8E 2818100008          <1>       db     40, 24, 16, 00h, 08h ; tw, th-1, ch, slength
20662 00005F93 08030002            <1>       db     08h, 03h, 00h, 02h  ; sequ regs
20663 00005F97 67                  <1>       db     67h    ; misc reg
20664 00005F98 2D2728902BA0BF1F    <1>       db     2Dh, 27h, 28h, 90h, 2Bh, 0A0h, 0BFh, 1Fh
20665 00005FA0 004F0D0E00000000    <1>       db     00h, 4Fh, 0Dh, 0Eh, 00h, 00h, 00h, 00h
20666 00005FA8 9C8E8F141F96B9A3    <1>       db     9Ch, 8Eh, 8Fh, 14h, 1Fh, 96h, 0B9h, 0A3h
20667 00005FB0 FF                  <1>       db     0FFh   ; crtc_regs
20668 00005FB1 0001020304051407    <1>       db     00h, 01h, 02h, 03h, 04h, 05h, 14h, 07h
20669 00005FB9 38393A3B3C3D3E3F    <1>       db     38h, 39h, 3Ah, 3Bh, 3Ch, 3Dh, 3Eh, 3Fh
20670 00005FC1 0C000F08            <1>       db     0Ch, 00h, 0Fh, 08h  ; actl regs
20671 00005FC5 0000000000100E0FFF  <1>       db     00h, 00h, 00h, 00h, 00h, 10h, 0Eh, 0Fh, 0FFh ; grdc regs
20672                              <1> ;vga_mode_07h:     ; mode 07h, 80*25 text, mono color
20673                              <1> ;      db     80, 24, 16, 00h, 10h  ; tw, th-1, ch, slength
20674                              <1> ;      db     00h, 03h, 00h, 02h ; sequ regs
20675                              <1> ;      db     66h    ; misc reg
20676                              <1> ;      db     5Fh, 4Fh, 50h, 82h, 55h, 81h, 0BFh, 1Fh
20677                              <1> ;      db     00h, 4Fh, 0Dh, 0Eh, 00h, 00h, 00h, 00h
20678                              <1> ;      db     9Ch, 8Eh, 8Fh, 28h, 0Fh, 96h, 0B9h, 0A3h
20679                              <1> ;      db     0FFh   ; crtc regs
20680                              <1> ;      db     00h, 08h, 08h, 08h, 08h, 08h, 08h, 08h
20681                              <1> ;      db     10h, 18h, 18h, 18h, 18h, 18h, 18h, 18h
20682                              <1> ;      db     0Eh, 00h, 0Fh, 08h  ; actl regs
20683                              <1> ;      db     00h, 00h, 00h, 00h, 00h, 10h, 0Ah, 0Fh, 0FFh ; grdc regs
20684                              <1> vga_mode_04h:      ; 320*200 graphics, 4 colors, CGA
20685 00005FCE 2818080008          <1>       db     40, 24, 8, 00h, 08h   ; tw, th-1, ch, slength
20686 00005FD3 09030002            <1>       db     09h, 03h, 00h, 02h ; sequ regs
20687 00005FD7 63                  <1>       db     63h    ; misc reg
20688 00005FD8 2D2728902B80BF1F    <1>       db     2Dh, 27h, 28h, 90h, 2Bh, 80h, 0BFh, 1Fh
20689 00005FE0 00C1000000000000    <1>       db     00h, 0C1h, 00h, 00h, 00h, 00h, 00h, 00h
20690 00005FE8 9C8E8F140096B9A2    <1>       db     9Ch, 8Eh, 8Fh, 14h, 00h, 96h, 0B9h, 0A2h
20691 00005FF0 FF                  <1>       db     0FFh   ; crtc_regs
20692 00005FF1 0013151702040607    <1>       db     00h, 13h, 15h, 17h, 02h, 04h, 06h, 07h
20693 00005FF9 1011121314151617    <1>       db     10h, 11h, 12h, 13h, 14h, 15h, 16h, 17h
20694 00006001 01000300            <1>       db     01h, 00h, 03h, 00h ; actl regs
20695 00006005 0000000000300F0FFF  <1>       db     00h, 00h, 00h, 00h, 00h, 30h, 0Fh, 0Fh, 0FFh ; grdc regs
20696                              <1> vga_mode_06h:      ; 640*200 graphics, 2 colors, CGA
20697 0000600E 5018080010          <1>       db     80, 24, 8, 00h, 10h   ;   tw, th-1, ch, slength
20698 00006013 01010006            <1>       db     01h, 01h, 00h, 06h ; sequ regs
20699 00006017 63                  <1>       db     63h    ; misc reg
```

```
20700 00006018 5F4F50825480BF1F    <1>       db     5Fh, 4Fh, 50h, 82h, 54h, 80h, 0BFh, 1Fh
20701 00006020 00C1000000000000    <1>       db     00h, 0C1h, 00h, 00h, 00h, 00h, 00h, 00h
20702 00006028 9C8E8F280096B9C2    <1>       db     9Ch, 8Eh, 8Fh, 28h, 00h, 96h, 0B9h, 0C2h
20703 00006030 FF                  <1>       db     0FFh   ; crtc regs
20704 00006031 0017171717171717    <1>       db     00h, 17h, 17h, 17h, 17h, 17h, 17h, 17h
20705 00006039 1717171717171717    <1>       db     17h, 17h, 17h, 17h, 17h, 17h, 17h, 17h
20706 00006041 01000100            <1>       db     01h, 00h, 01, 00h  ; actl regs
20707 00006045 0000000000000D0FFF  <1>       db     00h, 00h, 00h, 00h, 00h, 00h, 0Dh, 0Fh, 0FFh ; grdc regs
20708                              <1> vga_mode_13h:  ; mode 13h, 300*200, 256 colors, linear
20709 0000604E 2818080000          <1>       db     40, 24, 8, 0, 0     ; tw, th-1, ch, slength (5)
20710 00006053 010F000E            <1>       db     01h, 0Fh, 00h, 0Eh ; sequ regs (4)
20711 00006057 63                  <1>       db     63h    ; misc reg (1)
20712 00006058 5F4F50825480BF1F    <1>       db     5Fh, 4Fh, 50h, 82h, 54h, 80h, 0BFh, 1Fh
20713 00006060 0041000000000000    <1>       db     00h, 41h, 00h, 00h, 00h, 00h, 00h, 00h
20714 00006068 9C8E8F284096B9A3    <1>       db     9Ch, 8Eh, 8Fh, 28h, 40h, 96h, 0B9h, 0A3h
20715 00006070 FF                  <1>       db     0FFh   ; crtc regs (25)
20716 00006071 0001020304050607    <1>       db     00h, 01h, 02h, 03h, 04h, 05h, 06h, 07h
20717 00006079 08090A0B0C0D0E0F    <1>       db     08h, 09h, 0Ah, 0Bh, 0Ch, 0Dh, 0Eh, 0Fh
20718 00006081 41000F00            <1>       db     41h, 00h, 0Fh, 00h  ; actl regs (20)
20719 00006085 000000000040050FFF  <1>       db     00h, 00h, 00h, 00h, 00h, 40h, 05h, 0Fh, 0FFh ; grdc regs (9)
20720                              <1> vga_mode_setl equ $ - vga_mode_13h  ; = 64
20721                              <1> vga_mode_F0h:  ; mode X ; 320*240, 256 colors, planar
20722 0000608E 2818080000          <1>       db     40, 24, 8, 0, 0     ; tw, th-1, ch, slength
20723 00006093 010F0006            <1>       db     01h, 0Fh, 00h, 06h ; sequ regs
20724 00006097 E3                  <1>       db     0E3h   ; misc reg
20725 00006098 5F4F508254800D3E    <1>       db     5Fh, 4Fh, 50h, 82h, 54h, 80h, 0Dh, 3Eh
20726 000060A0 0041000000000000    <1>       db     00h, 41h, 00h, 00h, 00h, 00h, 00h, 00h
20727 000060A8 EAACDF2800E706E3    <1>       db     0EAh, 0ACh, 0DFh, 28h, 00h, 0E7h, 06h, 0E3h
20728 000060B0 FF                  <1>       db     0FFh   ; crtc regs (25)
20729 000060B1 0001020304050607    <1>       db     00h, 01h, 02h, 03h, 04h, 05h, 06h, 07h
20730 000060B9 08090A0B0C0D0E0F    <1>       db     08h, 09h, 0Ah, 0Bh, 0Ch, 0Dh, 0Eh, 0Fh
20731 000060C1 41000F00            <1>       db     41h, 00h, 0Fh, 00h  ; actl regs
20732 000060C5 000000000040050FFF  <1>       db     00h, 00h, 00h, 00h, 00h, 40h, 05h, 0Fh, 0FFh ; grdc regs
20733                              <1> vga_mode_12h:  ; mode 12h, 640*480, 16 colors, planar
20734 000060CE 501D100000          <1>       db     80, 29, 16, 0, 0 ; tw, th-1, ch, slength
20735 000060D3 010F0006            <1>       db     01h, 0Fh, 00h, 06h ; sequ regs
20736 000060D7 E3                  <1>       db     0E3h   ; misc reg
20737 000060D8 5F4F508254800B3E    <1>       db     5Fh, 4Fh, 50h, 82h, 54h, 80h, 0Bh, 3Eh
20738 000060E0 0040000000000000    <1>       db     00h, 40h, 00h, 00h, 00h, 00h, 00h, 00h
20739 000060E8 EA8CDF2800E704E3    <1>       db     0EAh, 8Ch, 0DFh, 28h, 00h, 0E7h, 04h, 0E3h
20740 000060F0 FF                  <1>       db     0FFh   ; crtc regs
20741 000060F1 0001020304051407    <1>       db     00h, 01h, 02h, 03h, 04h, 05h, 14h, 07h
20742 000060F9 38393A3B3C3D3E3F    <1>       db     38h, 39h, 3Ah, 3Bh, 3Ch, 3Dh, 3Eh, 3Fh
20743 00006101 01000F00            <1>       db     01h, 00h, 0Fh, 00h  ; actl regs
20744 00006105 000000000000050FFF  <1>       db     00h, 00h, 00h, 00h, 00h, 00h, 05h, 0Fh, 0FFh ; grdc regs
20745                              <1> vga_mode_6Ah:  ; mode 6Ah, 800*600, 16 colors, planar
20746 0000610E 6424100000          <1>       db     100, 36, 16, 0, 0 ; tw, th-1, ch, slength
20747 00006113 010F0006            <1>       db     01h, 0Fh, 00h, 06h ; sequ regs
20748 00006117 E3                  <1>       db     0E3h   ; misc reg
20749 00006118 7F6363836B1B72F0    <1>       db     7Fh, 63h, 63h, 83h, 6Bh, 1Bh, 72h, 0F0h
20750 00006120 0060000000000000    <1>       db     00h, 60h, 00h, 00h, 00h, 00h, 00h, 00h
20751 00006128 598D5732005773E3    <1>       db     59h, 8Dh, 57h, 32h, 00h, 57h, 73h, 0E3h
20752 00006130 FF                  <1>       db     0FFh   ; crtc regs
20753 00006131 0001020304051407    <1>       db     00h, 01h, 02h, 03h, 04h, 05h, 14h, 07h
20754 00006139 38393A3B3C3D3E3F    <1>       db     38h, 39h, 3Ah, 3Bh, 3Ch, 3Dh, 3Eh, 3Fh
20755 00006141 01000F00            <1>       db     01h, 00h, 0Fh, 00h  ; actl regs
20756 00006145 000000000000050FFF  <1>       db     00h, 00h, 00h, 00h, 00h, 00h, 05h, 0Fh, 0FFh ; grdc regs
20757                              <1> vga_mode_0Dh:  ; mode 0Dh, 320*200, 16 colors, planar
20758 0000614E 2818080020          <1>       db     40, 24, 8, 0, 20h ; tw, th-1, ch, slength
20759 00006153 090F0006            <1>       db     09h, 0Fh, 00h, 06h ; sequ regs
20760 00006157 63                  <1>       db     63h    ; misc reg
20761 00006158 2D2728902B80BF1F    <1>       db     2Dh, 27h, 28h, 90h, 2Bh, 80h, 0BFh, 1Fh
20762 00006160 00C0000000000000    <1>       db     00h, 0C0h, 00h, 00h, 00h, 00h, 00h, 00h
20763 00006168 9C8E8F140096B9E3    <1>       db     9Ch, 8Eh, 8Fh, 14h, 00h, 96h, 0B9h, 0E3h
20764 00006170 FF                  <1>       db     0FFh   ; crtc regs
20765 00006171 0001020304050607    <1>       db     00h, 01h, 02h, 03h, 04h, 05h, 06h, 07h
20766 00006179 1011121314151617    <1>       db     10h, 11h, 12h, 13h, 14h, 15h, 16h, 17h
20767 00006181 01000F00            <1>       db     01h, 00h, 0Fh, 00h  ; actl regs
20768 00006185 000000000000050FFF  <1>       db     00h, 00h, 00h, 00h, 00h, 00h, 05h, 0Fh, 0FFh ; grdc regs
20769                              <1> vga_mode_0Eh:  ; mode 0Eh, 640*200, 16 colors, planar
20770 0000618E 5018080040          <1>       db     80, 24, 8, 0, 40h ; tw, th-1, ch, slength
20771 00006193 010F0006            <1>       db     01h, 0Fh, 00h, 06h ; sequ regs
20772 00006197 63                  <1>       db     63h    ; misc reg
20773 00006198 5F4F50825480BF1F    <1>       db     5Fh, 4Fh, 50h, 82h, 54h, 80h, 0BFh, 1Fh
20774 000061A0 00C0000000000000    <1>       db     00h, 0C0h, 00h, 00h, 00h, 00h, 00h, 00h
20775 000061A8 9C8E8F280096B9E3    <1>       db     9Ch, 8Eh, 8Fh, 28h, 00h, 96h, 0B9h, 0E3h
20776 000061B0 FF                  <1>       db     0FFh   ; crtc regs
20777 000061B1 0001020304050607    <1>       db     00h, 01h, 02h, 03h, 04h, 05h, 06h, 07h
20778 000061B9 1011121314151617    <1>       db     10h, 11h, 12h, 13h, 14h, 15h, 16h, 17h
20779 000061C1 01000F00            <1>       db     01h, 00h, 0Fh, 00h  ; actl regs
20780 000061C5 000000000000050FFF  <1>       db     00h, 00h, 00h, 00h, 00h, 00h, 05h, 0Fh, 0FFh ; grdc regs
20781                              <1> vga_mode_10h: ; mode 10h, 640*350, 16 colors, planar
20782 000061CE 50180E0080          <1>       db     80, 24, 14, 0, 80h ; tw, th-1, ch, slength
20783 000061D3 010F0006            <1>       db     01h, 0Fh, 00h, 06h ; sequ regs
20784 000061D7 A3                  <1>       db     0A3h   ; misc reg
20785 000061D8 5F4F50825480BF1F    <1>       db     5Fh, 4Fh, 50h, 82h, 54h, 80h, 0BFh, 1Fh
20786 000061E0 0040000000000000    <1>       db     00h, 40h, 00h, 00h, 00h, 00h, 00h, 00h
20787 000061E8 83855D280F63BAE3    <1>       db     83h, 85h, 5Dh, 28h, 0Fh, 63h, 0BAh, 0E3h
20788 000061F0 FF                  <1>       db     0FFh   ; crtc regs
20789 000061F1 0001020304051407    <1>       db     00h, 01h, 02h, 03h, 04h, 05h, 14h, 07h
20790 000061F9 38393A3B3C3D3E3F    <1>       db     38h, 39h, 3Ah, 3Bh, 3Ch, 3Dh, 3Eh, 3Fh
20791 00006201 01000F00            <1>       db     01h, 00h, 0Fh, 00h  ; actl regs
20792 00006205 000000000000050FFF  <1>       db     00h, 00h, 00h, 00h, 00h, 00h, 05h, 0Fh, 0FFh ; grdc regs
20793                              <1> vga_mode_11h: ; mode 11h, 640*480, mono color, planar
20794 0000620E 501D100000          <1>       db     80, 29, 16, 0, 0   ; tw, th-1, ch, slength
20795 00006213 010F0006            <1>       db     01h, 0Fh, 00h, 06h ; sequ regs
20796 00006217 E3                  <1>       db     0E3h   ; misc reg
20797 00006218 5F4F508254800B3E    <1>       db     5Fh, 4Fh, 50h, 82h, 54h, 80h, 0Bh, 3Eh
20798 00006220 0040000000000000    <1>       db     00h, 40h, 00h, 00h, 00h, 00h, 00h, 00h
20799 00006228 EA8CDF2800E704E3    <1>       db     0EAh, 8Ch, 0DFh, 28h, 00h, 0E7h, 04h, 0E3h
20800 00006230 FF                  <1>       db     0FFh   ; crtc regs
20801 00006231 003F003F003F003F    <1>       db     00h, 3Fh, 00h, 3Fh, 00h, 3Fh, 00h, 3Fh
20802 00006239 003F003F003F003F    <1>       db     00h, 3Fh, 00h, 3Fh, 00h, 3Fh, 00h, 3Fh
```

```
20803 00006241 01000F00            <1>       db     01h, 00h, 0Fh, 00h  ; actl regs
20804 00006245 000000000000050FFF  <1>       db     00h, 00h, 00h, 00h, 00h, 00h, 05h, 0Fh, 0FFh ; grdc regs
20805                              <1> end_of_vga_params:
20806                              <1>
20807                              <1> ; /// End Of VIDEO DATA ///
20808                                  ;%include 'diskdata.s'    ; DISK (BIOS) DATA (initialized)
20809                                  ;;;
20810
20811                                  Align 2
20812
20813                                  %include 'sysdefs.s' ; 24/01/2015
20814                              <1> ; ****************************************************************************
20815                              <1> ; TRDOS386.ASM (TRDOS 386 Kernel - v2.0.0) - SYSTEM DEFINITIONS : sysdefs.s
20816                              <1> ; ----------------------------------------------------------------------------
20817                              <1> ; Last Update: 28/08/2017
20818                              <1> ; ----------------------------------------------------------------------------
20819                              <1> ; Beginning: 24/01/2016
20820                              <1> ; ----------------------------------------------------------------------------
20821                              <1> ; Assembler: NASM version 2.11 (trdos386.s)
20822                              <1> ; ----------------------------------------------------------------------------
20823                              <1> ; Derived from 'Retro UNIX 386 Kernel - v0.2.1.0' source code by Erdogan Tan
20824                              <1> ; sysdefs.inc (14/11/2015)
20825                              <1> ; ****************************************************************************
20826                              <1>
20827                              <1> ; Retro UNIX 386 v1 Kernel - SYSDEFS.INC
20828                              <1> ; Last Modification: 14/11/2015
20829                              <1> ;
20830                              <1> ; //////// RETRO UNIX 386 V1 SYSTEM DEFINITIONS //////////////
20831                              <1> ; (Modified from
20832                              <1> ;     Retro UNIX 8086 v1 system definitions in 'UNIX.ASM', 01/09/2014)
20833                              <1> ; ((UNIX.ASM (RETRO UNIX 8086 V1 Kernel), 11/03/2013 - 01/09/2014))
20834                              <1> ;     UNIX.ASM (MASM 6.11) --> SYSDEFS.INC (NASM 2.11)
20835                              <1> ; ----------------------------------------------------------------------------
20836                              <1> ;
20837                              <1> ; Derived from UNIX Operating System (v1.0 for PDP-11)
20838                              <1> ; (Original) Source Code by Ken Thompson (1971-1972)
20839                              <1> ; <Bell Laboratories (17/3/1972)>
20840                              <1> ; <Preliminary Release of UNIX Implementation Document>
20841                              <1> ;
20842                              <1> ; ****************************************************************************
20843                              <1>
20844                              <1> nproc       equ    16  ; number of processes
20845                              <1> nfiles      equ    50
20846                              <1> ntty  equ     8   ; 8+1 -> 8 (10/05/2013)
20847                              <1> nbuf  equ    4   ; 6 ;; 21/08/2015 - 'namei' buffer problem when nbuf > 4
20848                              <1>              ; NOTE: If fd0 super block buffer addres is beyond of the 1st
20849                              <1>              ; 32K, DMA r/w routine or someting else causes a jump to
20850                              <1>              ; kernel panic routine (in 'alloc' routine, in u5.s)
20851                              <1>              ; because of invalid buffer content (r/w error).
20852                              <1>              ; When all buffers are set before the end of the 1st 32k,
20853                              <1>              ; there is no problem!? (14/11/2015)
20854                              <1>
20855                              <1> ;csgmnt     equ    2000h  ; 26/05/2013 (segment of process 1)
20856                              <1> ;core equ    0          ; 19/04/2013
20857                              <1> ;ecore      equ    32768 - 64  ; 04/06/2013 (24/05/2013)
20858                              <1>       ; (if total size of argument list and arguments is 128 bytes)
20859                              <1>       ; maximum executable file size = 32768-(64+40+128-6) = 32530 bytes
20860                              <1>       ; maximum stack size = 40 bytes (+6 bytes for 'IRET' at 32570)
20861                              <1>       ; initial value of user's stack pointer = 32768-64-128-2 = 32574
20862                              <1>       ;     (sp=32768-args_space-2 at the beginning of execution)
20863                              <1>       ; argument list offset = 32768-64-128 = 32576 (if it is 128 bytes)
20864                              <1>       ; 'u' structure offset (for the '/core' dump file) = 32704
20865                              <1>       ; '/core' dump file size = 32768 bytes
20866                              <1>
20867                              <1> ; 08/03/2014
20868                              <1> ;sdsegmnt equ      6C0h  ; 256*16 bytes (swap data segment size for 16 processes)
20869                              <1> ; 19/04/2013 Retro UNIX 8086 v1 feaure only !
20870                              <1> ;;sdsegmnt equ     740h  ; swap data segment (for user structures and registers)
20871                              <1>
20872                              <1> ; 30/08/2013
20873                              <1> time_count equ 4 ; 10 --> 4 01/02/2014
20874                              <1>
20875                              <1> ; 05/02/2014
20876                              <1> ; process status
20877                              <1> ;SFREE      equ 0
20878                              <1> ;SRUN equ 1
20879                              <1> ;SWAIT      equ 2
20880                              <1> ;SZOMB      equ 3
20881                              <1> ;SSLEEP     equ 4 ; Retro UNIX 8086 V1 extension (for sleep and wakeup)
20882                              <1>
20883                              <1> ; 09/03/2015
20884                              <1> userdata equ 80000h ; user structure data address for current user ; temporary
20885                              <1> swap_queue equ 90000h - 2000h ; swap queue address ; temporary
20886                              <1> swap_alloc_table equ 0D0000h  ;  swap allocation table address ; temporary
20887                              <1>
20888                              <1> ; 17/09/2015
20889                              <1> ESPACE equ 48 ; [u.usp] (at 'sysent') - [u.sp] value for error return
20890                              <1>
20891                              <1> ; 19/02/2017
20892                              <1> ; 15/10/2016
20893                              <1> ; 20/05/2016
20894                              <1> ; 19/05/2016
20895                              <1> ; 18/05/2016
20896                              <1> ; 29/04/2016
20897                              <1> ; TRDOS 386 (TRDOS v2.0) system calls - temporary List
20898                              <1> ; 14/07/2013 - 21/09/2015 (Retro UNIX 8086 & 386 system calls)
20899                              <1> ; UNIX v1 system calls
20900                              <1> ;_rele      equ 0
20901                              <1> _ver  equ 0 ; Get TRDOS version (v2.0)
20902                              <1> _exit       equ 1
20903                              <1> _fork       equ 2
20904                              <1> _read       equ 3
20905                              <1> _write      equ 4
```

```
20906                                   <1> _open equ 5
20907                                   <1> _close      equ 6
20908                                   <1> _wait       equ 7
20909                                   <1> _creat      equ 8
20910                                   <1> _link       equ 9
20911                                   <1> _unlink     equ 10
20912                                   <1> _exec equ 11
20913                                   <1> _chdir      equ 12
20914                                   <1> _time       equ 13
20915                                   <1> _mkdir      equ 14
20916                                   <1> _chmod      equ 15
20917                                   <1> _chown      equ 16
20918                                   <1> _break      equ 17
20919                                   <1> _stat equ 18
20920                                   <1> _seek equ 19
20921                                   <1> _tell       equ 20
20922                                   <1> _mount      equ 21
20923                                   <1> _umount     equ 22
20924                                   <1> _setuid     equ 23
20925                                   <1> _getuid     equ 24
20926                                   <1> _stime      equ 25
20927                                   <1> _quit equ 26
20928                                   <1> _intr equ 27
20929                                   <1> _fstat      equ 28
20930                                   <1> _emt  equ 29
20931                                   <1> _mdate      equ 30
20932                                   <1> ;_stty      equ 31
20933                                   <1> _video  equ 31 ; TRDOS 386 Video Functions (16/05/2016)
20934                                   <1> ;_gtty      equ 32
20935                                   <1> _audio      equ 32 ; TRDOS 386 Video Functions (16/05/2016)
20936                                   <1> ;_ilgins equ 33
20937                                   <1> _timer      equ 33 ; TRDOS 386 Timer Functions (18/05/2016)
20938                                   <1> _sleep      equ 34 ; Retro UNIX 8086 v1 feature only !
20939                                   <1> _msg  equ 35 ; Retro UNIX 386 v1 feature only !
20940                                   <1> _geterr     equ 36 ; Retro UNIX 386 v1 feature only !
20941                                   <1> _fpsave equ 37 ; TRDOS 386 FPU state option (28/02/2017)
20942                                   <1> _pri  equ 38 ; change priority - TRDOS 386 (20/05/2016)
20943                                   <1> _rele equ 39 ; TRDOS 386 (19/05/2016)
20944                                   <1> _fff  equ 40 ; Find First File - TRDOS 386 (15/10/2016)
20945                                   <1> _fnf  equ 41 ; Find Next File - TRDOS 386 (15/10/2016)
20946                                   <1> _alloc      equ 42 ; Allocate memory - TRDOS 386 (19/02/2017)
20947                                   <1>             ; TRDOS 386 (19/02/2017) DMA buff fuctions
20948                                   <1> _dalloc equ 43 ; Deallocate mem - TRDOS 386 (19/02/2017)
20949                                   <1> _calbac equ 44 ; Set IRQ callback - TRDOS 386 (20/02/2017)
20950                                   <1> _dma  equ 45 ; DMA service - TRDOS 386 (20/08/2017)
20951                                   <1>
20952                                   <1> %macro sys 1-4
20953                                   <1>     ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
20954                                   <1>     ; 03/09/2015
20955                                   <1>     ; 13/04/2015
20956                                   <1>     ; Retro UNIX 386 v1 system call.
20957                                   <1>     %if %0 >= 2
20958                                   <1>         mov ebx, %2
20959                                   <1>         %if %0 >= 3
20960                                   <1>             mov ecx, %3
20961                                   <1>             %if %0 = 4
20962                                   <1>                mov edx, %4
20963                                   <1>             %endif
20964                                   <1>         %endif
20965                                   <1>     %endif
20966                                   <1>     mov eax, %1
20967                                   <1>     ;int 30h
20968                                   <1>     int 40h ; TRDOS 386 (TRDOS v2.0)
20969                                   <1> %endmacro
20970                                   <1>
20971                                   <1> ; TRDOS 386 system calls, interrupt number
20972                                   <1> ; 25/12/2016
20973                                   <1> SYSCALL_INT_NUM   equ '40' ; '40h'
20974                                   <1>
20975                                   <1> ; 13/05/2015 - ERROR CODES
20976                                   <1> ERR_FILE_NOT_OPEN  equ 10 ; 'file not open !' error
20977                                   <1> ERR_FILE_ACCESS    equ 11 ; 'permission denied !' error
20978                                   <1> ; 14/05/2015
20979                                   <1> ERR_DIR_ACCESS     equ 11 ; 'permission denied !' error
20980                                   <1> ERR_FILE_NOT_FOUND equ 12 ; 'file not found !' error
20981                                   <1> ERR_TOO_MANY_FILES equ 13 ; 'too many open files !' error
20982                                   <1> ERR_DIR_EXISTS     equ 14 ; 'directory already exists !' error
20983                                   <1> ; 16/05/2015
20984                                   <1> ERR_DRV_NOT_RDY    equ 15 ; 'drive not ready !' error
20985                                   <1> ; 18/05/2015
20986                                   <1> ERR_DEV_NOT_RDY    equ 15 ; 'device not ready !' error
20987                                   <1> ERR_DEV_ACCESS     equ 11 ; 'permission denied !' error
20988                                   <1> ERR_DEV_NOT_OPEN   equ 10 ; 'device not open !' error
20989                                   <1> ; 07/06/2015
20990                                   <1> ERR_FILE_EOF    equ 16 ; 'end of file !' error
20991                                   <1> ERR_DEV_VOL_SIZE   equ 16 ; 'out of volume !' error
20992                                   <1> ; 09/06/2015
20993                                   <1> ERR_DRV_READ    equ 17 ; 'disk read error !'
20994                                   <1> ERR_DRV_WRITE         equ 18 ; 'disk write error !'
20995                                   <1> ; 16/06/2015
20996                                   <1> ERR_NOT_DIR     equ 19 ; 'not a (valid) directory !' error
20997                                   <1> ERR_FILE_SIZE         equ 20 ; 'file size error !'
20998                                   <1> ; 22/06/2015
20999                                   <1> ERR_NOT_SUPERUSER  equ 11 ; 'permission denied !' error
21000                                   <1> ERR_NOT_OWNER      equ 11 ; 'permission denied !' error
21001                                   <1> ERR_NOT_FILE       equ 11 ; 'permission denied !' error
21002                                   <1> ; 23/06/2015
21003                                   <1> ERR_FILE_EXISTS    equ 14 ; 'file already exists !' error
21004                                   <1> ERR_DRV_NOT_SAME   equ 21 ; 'not same drive !' error
21005                                   <1> ERR_DIR_NOT_FOUND  equ 12 ; 'directory not found !' error
21006                                   <1> ERR_NOT_EXECUTABLE equ 22 ; 'not executable file !' error
21007                                   <1> ; 27/06/2015
21008                                   <1> ERR_INV_PARAMETER  equ 23 ; 'invalid parameter !' error
```

```
21009                                  <1> ERR_INV_DEV_NAME   equ 24 ; 'invalid device name !' error
21010                                  <1> ; 29/06/2015
21011                                  <1> ERR_TIME_OUT    equ 25 ; 'time out !' error
21012                                  <1> ERR_DEV_NOT_RESP   equ 25 ; 'device not responding !' error
21013                                  <1> ; 10/10/2016
21014                                  <1> ERR_INV_FILE_NAME  equ 26 ; 'invalid file name !' error
21015                                  <1> ERR_INV_FLAGS         equ 23 ; 'invalid flags !' error
21016                                  <1> ; For code compatibility with previous version of TRDOS (2011)
21017                                  <1> ; (Temporary error codes for current TRDOS 386 -2016- version)
21018                                  <1> ERR_NO_MORE_FILES  equ 12 ; 'no more files !' error
21019                                  <1> ERR_PATH_NOT_FOUND equ  3 ; 'path not found !' error
21020                                  <1>                         ; 'dir not found !' ; TRDOS 8086
21021                                  <1> ERR_NOT_FOUND:         equ  2 ; 'file not found !' ; TRDOS 8086
21022                                  <1> ERR_DISK_SPACE         equ 39 ; 'out of volume !' TRDOS 8086
21023                                  <1>                        ; 'insufficient disk space !' ; 27h
21024                                  <1> ERR_DISK_WRITE         equ 30 ; 'disk write protected !' ; 16/10/2016
21025                                  <1> ERR_ACCESS_DENIED  equ  5 ; 'access denied !' ; TRDOS 8086
21026                                  <1> ; 28/02/2017
21027                                  <1> ERR_PERM_DENIED        equ 11 ; 'permission denied !' error
21028                                  <1> ; 18/05/2016
21029                                  <1> ERR_MISC        equ 27 ; miscellaneous/other errors
21030                                  <1> ; 15/10/2016
21031                                  <1> ; TRDOS 8086 -> TRDOS 386 (0Bh -> 28)
21032                                  <1> ERR_INV_FORMAT         equ 28 ; 'invalid format !' error
21033                                  <1> ; TRDOS 8086 -> TRDOS 386 (0Dh -> 29)
21034                                  <1> ERR_INV_DATA    equ 29 ; 'invalid data !' error
21035                                  <1> ; TRDOS 8086 -> TRDOS 386 (0Eh -> 20)
21036                                  <1> ERR_ZERO_LENGTH        equ 20  ; 'zero length !' error
21037                                  <1> ; TRDOS 8086 -> TRDOS 386 (15h -> 17, 1Dh -> 18, 1Eh -> 17)
21038                                  <1> ERR_DRV_NR_READ        equ 17 ; 'drive not ready or read error !'
21039                                  <1> ERR_DRV_NR_WRITE   equ 18 ; 'drive not ready or write error !'
21040                                  <1> ; 15/10/2016
21041                                  <1> ERR_INV_PATH_NAME  equ 19 ; 'bad path name !' error
21042                                  <1> ERR_BAD_CMD_ARG        equ  1 ; 'bad command argument !' ; TRDOS 8086
21043                                  <1> ERR_INV_FNUMBER        equ  1 ; 'invalid function number !' ; TRDOS 8086
21044                                  <1> ERR_BIG_FILE    equ  8 ; 'big file & out of memory ! ; TRDOS 8086
21045                                  <1> ERR_BIG_DATA    equ  8 ; 'big data & out of memory ! ; TRDOS 8086
21046                                  <1> ERR_CLUSTER     equ 35 ; 'cluster not available !' ; TRDOS 8086
21047                                  <1> ERR_OUT_OF_MEMORY  equ  4 ; 'out of memory !'
21048                                  <1>                         ; 'insufficient memory !'
21049                                  <1> ERR_P_VIOLATION        equ 6 ; 'protection violation !'
21050                                  <1> ERR_PAGE_FAULT         equ 224 ;'page fault !' ;0E0h
21051                                  <1> ERR_SWP_DISK_READ            equ 40
21052                                  <1> ERR_SWP_DISK_NOT_PRESENT   equ 41
21053                                  <1> ERR_SWP_SECTOR_NOT_PRESENT equ 42
21054                                  <1> ERR_SWP_NO_FREE_SPACE      equ 43
21055                                  <1> ERR_SWP_DISK_WRITE         equ 44
21056                                  <1> ERR_SWP_NO_PAGE_TO_SWAP    equ 45
21057                                  <1> ; 10/04/2017
21058                                  <1> ERR_BUFFER     equ 46  ; 'buffer error !'
21059                                  <1> ; 28/08/2017 (20/08/2017)
21060                                  <1> ERR_DMA               equ -1  ; DMA buffer (allocation/misc.) error!
21061                                  <1>
21062                                  <1> ; 26/08/2015
21063                                  <1> ; 24/07/2015
21064                                  <1> ; 24/06/2015
21065                                  <1> MAX_ARG_LEN    equ 256 ; max. length of sys exec arguments
21066                                  <1> ; 01/07/2015
21067                                  <1> MAX_MSG_LEN    equ 255 ; max. msg length for 'sysmsg'
21068                                  <1> ;
21069                                  <1> ; 06/10/2016
21070                                  <1> OPENFILES      equ 10  ; max. number of open files (system)
21071                                  <1> ; 07/10/2016
21072                                  <1> ;NUMOFDEVICES         equ 20  ; max. num of available devices (sys)
21073                                  <1>
21074                                      %include 'trdosk0.s' ; 04/01/2016
21075                                  <1> ; ****************************************************************************
21076                                  <1> ; TRDOS386.ASM (TRDOS 386 Kernel - v2.0.0) - DEFINITIONS : trdosk0.s
21077                                  <1> ; ----------------------------------------------------------------------------
21078                                  <1> ; Last Update: 29/02/2016
21079                                  <1> ; ----------------------------------------------------------------------------
21080                                  <1> ; Beginning: 04/01/2016
21081                                  <1> ; ----------------------------------------------------------------------------
21082                                  <1> ; Assembler: NASM version 2.11 (trdos386.s)
21083                                  <1> ; ----------------------------------------------------------------------------
21084                                  <1> ; Derived from TRDOS Operating System v1.0 (8086) source code by Erdogan Tan
21085                                  <1> ; TRDOS2.ASM (09/11/2011)
21086                                  <1> ; ****************************************************************************
21087                                  <1> ; TRDOS2.ASM (c) 2004-2011 Erdogan TAN [ 17/01/2004 ] Last Update: 09/11/2011
21088                                  <1> ;
21089                                  <1> ; Masterboot / Partition Table at Beginning+1BEh
21090                                  <1> ptBootable       equ 0
21091                                  <1> ptBeginHead      equ 1
21092                                  <1> ptBeginSector    equ 2
21093                                  <1> ptBeginCylinder  equ 3
21094                                  <1> ptFileSystemID   equ 4
21095                                  <1> ptEndHead        equ 5
21096                                  <1> ptEndSector      equ 6
21097                                  <1> ptEndCylinder    equ 7
21098                                  <1> ptStartSector    equ 8
21099                                  <1> ptSectors        equ 12
21100                                  <1>
21101                                  <1> ; Boot Sector Parameters at 7C00h
21102                                  <1> DataArea1     equ -4
21103                                  <1> DataArea2     equ -2
21104                                  <1> BootStart     equ 0h
21105                                  <1> OemName       equ 03h
21106                                  <1> BytesPerSec   equ 0Bh
21107                                  <1> SecPerClust   equ 0Dh
21108                                  <1> ResSectors    equ 0Eh
21109                                  <1> FATs          equ 10h
21110                                  <1> RootDirEnts   equ 11h
21111                                  <1> Sectors       equ 13h
```

```
21112                                   <1> Media         equ 15h
21113                                   <1> FATSecs       equ 16h
21114                                   <1> SecPerTrack   equ 18h
21115                                   <1> Heads         equ 1Ah
21116                                   <1> Hidden1       equ 1Ch
21117                                   <1> Hidden2       equ 1Eh
21118                                   <1> HugeSec1      equ 20h
21119                                   <1> HugeSec2      equ 22h
21120                                   <1> DriveNumber   equ 24h
21121                                   <1> Reserved1     equ 25h
21122                                   <1> bootsignature equ 26h
21123                                   <1> VolumeID      equ 27h
21124                                   <1> VolumeLabel   equ 2Bh
21125                                   <1> FileSysType   equ 36h
21126                                   <1> Reserved2     equ 3Eh                       ; Starting cluster of P2000
21127                                   <1>
21128                                   <1> ; FAT32 BPB Structure
21129                                   <1> FAT32_FAT_Size equ 36
21130                                   <1> FAT32_RootFClust equ 44
21131                                   <1> FAT32_FSInfoSec equ 48
21132                                   <1> FAT32_DrvNum equ 64
21133                                   <1> FAT32_BootSig equ 66
21134                                   <1> FAT32_VolID equ 67
21135                                   <1> FAT32_VolLab equ 71
21136                                   <1> FAT32_FilSysType equ 82
21137                                   <1>
21138                                   <1> ; BIOS Disk Parameters
21139                                   <1> DPDiskNumber  equ 0h
21140                                   <1> DPDType       equ 1h
21141                                   <1> DPReturn      equ 2h
21142                                   <1> DPHeads       equ 3h
21143                                   <1> DPCylinders   equ 4h
21144                                   <1> DPSecPerTrack equ 6h
21145                                   <1> DPDisks       equ 7h
21146                                   <1> DPTableOff    equ 8h
21147                                   <1> DPTableSeg    equ 0Ah
21148                                   <1> DPNumOfSecs   equ 0Ch
21149                                   <1>
21150                                   <1> ; BIOS INT 13h Extensions (LBA extensions)
21151                                   <1> ; Just After DP Data (DPDiskNumber+)
21152                                   <1> DAP_PacketSize equ 10h  ; If extensions present, this byte will be >=10h
21153                                   <1> DAP_Reserved1 equ 11h   ; Reserved Byte
21154                                   <1> DAP_NumOfBlocks equ 12h ; Value of this byte must be 0 to 127
21155                                   <1> DAP_Reserved2 equ 13h   ; Reserved Byte
21156                                   <1> DAP_Destination equ 14h ; Address of Transfer Buffer as SEGMENT:OFFSET
21157                                   <1> DAP_LBA_Address equ 18h ; LBA=(C1*H0+H1)*S0+S1-1
21158                                   <1>                         ; C1= Selected Cylinder Number
21159                                   <1>                         ; H0= Number Of Heads (Maximum Head Number + 1)
21160                                   <1>                         ; H1= Selected Head Number
21161                                   <1>                         ; S0= Maximum Sector Number
21162                                   <1>                         ; S1= Selected Sector Number
21163                                   <1>                         ; QUAD WORD
21164                                   <1> ; DAP_Flat_Destination equ 20h ; 64 bit address, if value in 4h is FFFF:FFFFh
21165                                   <1>                               ; QUAD WORD (Also, value in 0h must be 18h)
21166                                   <1>                               ; TR-DOS will not use 64 bit Flat Address
21167                                   <1>
21168                                   <1> ; INT 13h Function 48h "Get Enhanced Disk Drive Parameters"
21169                                   <1> ; Just After DP Data (DPDiskNumber+)
21170                                   <1> GetDParams_48h equ 20h ; Word. Data Length, must be 26 (1Ah) for short data.
21171                                   <1> GDP_48h_InfoFlag equ 22h ; Word
21172                                   <1> ; Bit 1 = 1 -> The geometry returned in bytes 4-15 is valid.
21173                                   <1> GDP_48h_NumOfPCyls equ 24h ; Double Word. Number physical cylinders.
21174                                   <1> GDP_48h_NumOfPHeads equ 28h ; Double Word. Number of physical heads.
21175                                   <1> GDP_48h_NumOfPSpT equ 2Ch ; Double word. Num of physical sectors per track.
21176                                   <1> GDP_48h_LBA_Sectors equ 30h ; 8 bytes. Number of physical/LBA sectors.
21177                                   <1> GDP_48h_BytesPerSec equ 38h ; Word. Number of bytes in a sector.
21178                                   <1>
21179                                   <1> ; TR-DOS Standalone Program Extensions to the DiskParams Block
21180                                   <1> ; Just After DP Data (DPDiskNumber+)
21181                                   <1> TRDP_CurrentSector equ 3Ah  ; DX:AX (LBA)
21182                                   <1> TRDP_SectorCount equ 3Eh    ; CX (or Counter)
21183                                   <1>
21184                                   <1>
21185                                   <1> ; DOS Logical Disks
21186                                   <1> LD_Name equ 0
21187                                   <1> LD_DiskType equ 1
21188                                   <1> LD_PhyDrvNo equ 2
21189                                   <1> LD_FATType equ 3
21190                                   <1> LD_FSType equ 4
21191                                   <1> LD_LBAYes equ 5
21192                                   <1> LD_BPB equ 6
21193                                   <1> LD_FATBegin equ 96
21194                                   <1> LD_ROOTBegin equ 100
21195                                   <1> LD_DATABegin equ 104
21196                                   <1> LD_StartSector equ 108
21197                                   <1> LD_TotalSectors equ 112
21198                                   <1> LD_FreeSectors equ 116
21199                                   <1> LD_Clusters equ 120
21200                                   <1> LD_PartitionEntry equ 124
21201                                   <1> LD_DParamEntry equ 125
21202                                   <1> LD_MediaChanged equ 126
21203                                   <1> LD_CDirLevel equ 127
21204                                   <1> LD_CurrentDirectory equ 128
21205                                   <1>
21206                                   <1> ; Singlix FS Extensions to DOS Logical Disks
21207                                   <1> ; 03/01/2010 (LD_BPB compatibility for CHS r/w)
21208                                   <1>
21209                                   <1> LD_FS_Name equ 0
21210                                   <1> LD_FS_DiskType equ 1
21211                                   <1> LD_FS_PhyDrvNo equ 2
21212                                   <1> LD_FS_FATType equ 3
21213                                   <1> LD_FS_FSType equ 4
21214                                   <1> LD_FS_LBAYes equ 5
```

```
21215                                   <1> LD_FS_BPB equ 6
21216                                   <1> LD_FS_MediaAttrib equ 6
21217                                   <1> LD_FS_VersionMajor equ 7
21218                                   <1> LD_FS_RootDirD equ 8
21219                                   <1> LD_FS_MATLocation equ 12
21220                                   <1> LD_FS_Reserved1 equ 16 ;1 reserved byte
21221                                   <1> LD_FS_BytesPerSec equ 17 ; LD_BPB + 0Bh
21222                                   <1> LD_FS_Reserved2 equ 19 ;2 reserved byte
21223                                   <1> LD_FS_DATLocation equ 20
21224                                   <1> LD_FS_DATSectors equ 24
21225                                   <1> LD_FS_Reserved3 equ 28 ;3 reserved word
21226                                   <1> LD_FS_SecPerTrack equ 30 ; LD_BPB + 18h
21227                                   <1> LD_FS_NumHeads equ 32    ; LD_BPB + 1Ah
21228                                   <1> LD_FS_UnDelDirD equ 34
21229                                   <1> LD_FS_Reserved4 equ 38 ;4 reserved word
21230                                   <1> LD_FS_VolumeSerial equ 40
21231                                   <1> LD_FS_VolumeName equ 44
21232                                   <1> LD_FS_BeginSector equ 108
21233                                   <1> LD_FS_VolumeSize equ 112
21234                                   <1> LD_FS_FreeSectors equ 116
21235                                   <1> LD_FS_FirstFreeSector equ 120
21236                                   <1> LD_FS_PartitionEntry equ 124
21237                                   <1> LD_FS_DParamEntry equ 125
21238                                   <1> LD_FS_MediaChanged equ 126
21239                                   <1> LD_FS_CDirLevel equ 127
21240                                   <1> LD_FS_CDIR_Converted equ 128
21241                                   <1>
21242                                   <1> ; Valid FAT Types
21243                                   <1> FS_FAT12 equ 1
21244                                   <1> FS_FAT16_CHS equ 2
21245                                   <1> FS_FAT32_CHS equ 3
21246                                   <1> FS_FAT16_LBA equ 4
21247                                   <1> FS_FAT32_LBA equ 5
21248                                   <1>
21249                                   <1> ; Cursor Location
21250                                   <1> CCCpointer equ  0450h   ; BIOS data, current cursor column
21251                                   <1> ; FAT Clusters EOC sign
21252                                   <1> FAT12EOC equ 0FFFh
21253                                   <1> FAT16EOC equ 0FFFFh
21254                                   <1> ;FAT32EOC equ 0FFFFFFFh ; It is not direct usable for 8086 code
21255                                   <1> ; BAD Cluster
21256                                   <1> FAT12BADC equ 0FF7h
21257                                   <1> FAT16BADC equ 0FFF7h
21258                                   <1> ;FAT32BADC equ 0FFFFFF7h ; It is not direct usable for 8086 code
21259                                   <1> ; MS-DOS FAT16 FS (Maximum Possible) Last Cluster Number= 0FFF6h
21260                                   <1>
21261                                   <1> ; TRFS
21262                                   <1>
21263                                   <1> bs_FS_JmpBoot equ 0 ; jmp short bsBootCode
21264                                   <1>                     ; db 0EBh, db 3Fh, db 90h
21265                                   <1> bs_FS_Identifier equ 3  ; db 'FS', db 0
21266                                   <1> bs_FS_BytesPerSec equ 6 ; dw 512
21267                                   <1> bs_FS_MediaAttrib equ 8 ; db 3
21268                                   <1> bs_FS_PartitionID equ 9 ; db 0A1h
21269                                   <1> bs_FS_VersionMaj equ 10 ; db 01h
21270                                   <1> bs_FS_VersionMin equ 11 ; db 0
21271                                   <1> bs_FS_BeginSector equ 12   ; dd 0
21272                                   <1> bs_FS_VolumeSize equ 16 ; dd 2880
21273                                   <1> bs_FS_StartupFD equ 20 ; dd 0
21274                                   <1> bs_FS_MATLocation equ 24 ; dd 1
21275                                   <1> bs_FS_RootDirD equ 28 ; dd 8
21276                                   <1> bs_FS_SystemConfFD equ 32 ; dd 0
21277                                   <1> bs_FS_SwapFD equ 36 ; dd 0
21278                                   <1> bs_FS_UnDelDirD equ 40 ; dd 0
21279                                   <1> bs_FS_DriveNumber equ 44 ; db 0
21280                                   <1> bs_FS_LBA_Ready equ 45 ; db 0
21281                                   <1> bs_FS_MagicWord equ 46
21282                                   <1> bs_FS_SecPerTrack equ 46 ; db 0A1h
21283                                   <1> bs_FS_Heads equ 47 ; db 01h
21284                                   <1> bs_FS_OperationSys equ 48 ; db "TR-SINGLIX v1.0b"
21285                                   <1> bs_FS_Terminator equ 64 ; db 0
21286                                   <1> bs_FS_BootCode equ 65
21287                                   <1>
21288                                   <1> FS_MAT_DATLocation equ 12
21289                                   <1> FS_MAT_DATScount equ 16
21290                                   <1> FS_MAT_FreeSectors equ 20
21291                                   <1> FS_MAT_FirstFreeSector equ 24
21292                                   <1> FS_RDT_VolumeSerialNo equ 28
21293                                   <1> FS_RDT_VolumeName equ 64
21294                                   <1>
21295                                   <1> ; FAT12 + FAT16 + FAT32
21296                                   <1> BS_JmpBoot equ 0
21297                                   <1> BS_OEMName equ 3
21298                                   <1> BPB_BytsPerSec equ 11
21299                                   <1> BPB_SecPerClust equ 13
21300                                   <1> BPB_RsvdSecCnt equ 14
21301                                   <1> BPB_NumFATs equ 16
21302                                   <1> BPB_RootEntCnt equ 17
21303                                   <1> BPB_TotalSec16 equ 19
21304                                   <1> BPB_Media equ 21
21305                                   <1> BPB_FATSz16 equ 22
21306                                   <1> BPB_SecPerTrk equ 24
21307                                   <1> BPB_NumHeads equ 26
21308                                   <1> BPB_HiddSec equ 28
21309                                   <1> BPB_TotalSec32 equ 32
21310                                   <1>
21311                                   <1> ; FAT12 and FAT16 only
21312                                   <1> BS_DrvNum equ 36
21313                                   <1> BS_Reserved1 equ 37
21314                                   <1> BS_BootSig equ 38
21315                                   <1> BS_VolID equ 39
21316                                   <1> BS_VolLab equ 43
21317                                   <1> BS_FilSysType equ 54 ; 8 bytes
```

```
21318                                  <1> BS_BootCode equ 62
21319                                  <1>
21320                                  <1> ; FAT32 only
21321                                  <1> BPB_FATSz32 equ 36 ; FAT32, 4 bytes
21322                                  <1> BPB_ExtFlags equ 40 ; FAT32, 2 bytes
21323                                  <1> BPB_FSVer equ 42 ; FAT32, 2 bytes
21324                                  <1> BPB_RootClus equ 44 ; FAT32, 4 bytes
21325                                  <1> BPB_FSInfo equ 48 ; FAT 32, 2 bytes
21326                                  <1> BPB_BkBootSec equ 50 ; FAT32, 2 bytes
21327                                  <1> BPB_Reserved equ 52 ; FAT32, 12 bytes
21328                                  <1> BS_FAT32_DrvNum equ 64 ; FAT32, 1 byte
21329                                  <1> BS_FAT32_Reserved1 equ 65 ; FAT32, 1 byte
21330                                  <1> BS_FAT32_BootSig equ 66 ; FAT32, 1 byte
21331                                  <1> BS_FAT32_VolID equ 67 ; FAT32, 4 bytes
21332                                  <1> BS_FAT32_VolLab equ 71 ; FAT32, 11 bytes
21333                                  <1> BS_FAT32_FilSysType equ 82 ; FAT32, 8 bytes
21334                                  <1> BS_FAT32_BootCode equ 90
21335                                  <1>
21336                                  <1> ; 29/02/2016
21337                                  <1> ;(FAT32 Free Cluster Count & First Free Cluster values)
21338                                  <1> ;[BPB_Reserved] = Free Cluster Count (offset 52)
21339                                  <1> ;[BPB_Reserved+4] = First Free Cluster (offset 56)
21340                                  <1>
21341                                  <1> BS_Validation equ 510
21342                                  <1>
21343                                  <1> ; 15/02/2016
21344                                  <1> ; FILE.ASM - 09/10/2011
21345                                  <1> ; Directory Entry Structure
21346                                  <1> ; 29/10/2009 (According to Microsoft FAT32 File System Specification)
21347                                  <1> DirEntry_Name equ 0
21348                                  <1> DirEntry_Attr equ 11
21349                                  <1> DirEntry_NTRes equ 12
21350                                  <1> DirEntry_CrtTimeTenth equ 13
21351                                  <1> DirEntry_CrtTime equ 14
21352                                  <1> DirEntry_CrtDate equ 16
21353                                  <1> DirEntry_LastAccDate equ 18
21354                                  <1> DirEntry_FstClusHI equ 20
21355                                  <1> DirEntry_WrtTime equ 22
21356                                  <1> DirEntry_WrtDate equ 24
21357                                  <1> DirEntry_FstClusLO equ 26
21358                                  <1> DirEntry_FileSize equ 28
21359                                      %include 'trdosk1.s' ; 04/01/2016
21360                                  <1> ; ****************************************************************************
21361                                  <1> ; TRDOS386.ASM (TRDOS 386 Kernel - v2.0.0) - SYS INIT : trdosk1.s
21362                                  <1> ; ----------------------------------------------------------------------------
21363                                  <1> ; Last Update: 23/01/2017
21364                                  <1> ; ----------------------------------------------------------------------------
21365                                  <1> ; Beginning: 04/01/2016
21366                                  <1> ; ----------------------------------------------------------------------------
21367                                  <1> ; Assembler: NASM version 2.11 (trdos386.s)
21368                                  <1> ; ----------------------------------------------------------------------------
21369                                  <1> ; Derived from TRDOS Operating System v1.0 (8086) source code by Erdogan Tan
21370                                  <1> ; TRDOS2.ASM (09/11/2011)
21371                                  <1> ; ****************************************************************************
21372                                  <1> ; TRDOS2.ASM (c) 2004-2011 Erdogan TAN [ 17/01/2004 ] Last Update: 09/11/2011
21373                                  <1> ;
21374                                  <1>
21375                                  <1> sys_init:
21376                                  <1>       ; 23/01/2017
21377                                  <1>       ; 07/05/2016
21378                                  <1>       ; 02/05/2016
21379                                  <1>       ; 24/04/2016
21380                                  <1>       ; 14/04/2016
21381                                  <1>       ; 13/04/2016
21382                                  <1>       ; 30/03/2016
21383                                  <1>       ; 24/01/2016
21384                                  <1>       ; 06/01/2016
21385                                  <1>       ; 04/01/2016
21386                                  <1>
21387                                  <1>       ; 23/01/2017 - reset timer frequency (to 18.2Hz)
21388 0000624E B036                   <1>       mov   al, 00110110b ; 36h
21389 00006250 E643                   <1>       out   43h, al
21390 00006252 31C0                   <1>       xor   eax, eax  ; sub     al, al ; 0
21391 00006254 E640                   <1>       out   40h, al ; LB
21392 00006256 E640                   <1>       out   40h, al ; HB
21393                                  <1>       ;
21394                                  <1>       ; 30/03/2016
21395                                  <1>       ; Clear Logical DOS Disk Description Tables Area
21396                                  <1>       ;xor   eax, eax
21397 00006258 BF00010900             <1>       mov   edi, Logical_DOSDisks
21398 0000625D B980060000             <1>       mov   ecx, 6656/4 ; 26*256 = 6656 bytes
21399 00006262 F3AB                   <1>       rep   stosd ; 1664 times 4 bytes
21400                                  <1>
21401 00006264 B83F3A2F00             <1>       mov   eax, '?:/'
21402 00006269 A3[E7520100]           <1>       mov   [Current_Dir_Drv], eax
21403                                  <1>
21404                                  <1>       ; Logical DRV INIT (only for hard disks)
21405 0000626E E8B3010000             <1>       call  ldrv_init  ; trdosk2.s
21406                                  <1>
21407                                  <1>       ; When floppy_drv_init call is disabled
21408                                  <1>       ; media changed sign is needed
21409                                  <1>       ; for proper drive initialization
21410                                  <1>
21411 00006273 BE00010900             <1>       mov   esi, Logical_DOSDisks
21412 00006278 B001                   <1>       mov   al, 1 ; Initialization sign (invalid_fd_parameter)
21413 0000627A 83C67E                 <1>       add   esi, LD_MediaChanged ; Media Change Status = 1 (init needed)
21414 0000627D 8806                   <1>       mov   [esi], al ; A:
21415 0000627F 81C600010000           <1>       add   esi, 100h
21416 00006285 8806                   <1>       mov   [esi], al ; B:
21417                                  <1>
21418                                  <1> _current_drive_bootdisk:
21419 00006287 8A15[F25C0000]         <1>       mov   dl, [boot_drv] ; physical drive number
21420 0000628D 80FAFF                 <1>       cmp   dl, 0FFh
```

```
21421 00006290 740A              <1>       je     short _last_dos_diskno_check
21422                            <1> _boot_drive_check:
21423 00006292 80FA80            <1>       cmp    dl, 80h
21424 00006295 7218              <1>       jb     short _current_drive_a
21425 00006297 80EA7E            <1>       sub    dl, 7Eh ; C = 2 , D = 3
21426 0000629A EB13              <1>       jmp    short _current_drive_a
21427                            <1>
21428                            <1> _last_dos_diskno_check:
21429 0000629C 8A15[D2060100]    <1>       mov    dl, [Last_DOS_DiskNo]
21430 000062A2 80FA02            <1>       cmp    dl, 2
21431 000062A5 7706              <1>       ja     short _current_drive_c
21432 000062A7 7406              <1>       je     short _current_drive_a
21433 000062A9 30D2              <1>       xor    dl, dl ; A:
21434 000062AB EB02              <1>       jmp    short _current_drive_a
21435                            <1>
21436                            <1> _current_drive_c:
21437 000062AD B202              <1>       mov    dl, 2 ; C:
21438                            <1>
21439                            <1> _current_drive_a:
21440 000062AF 8815[F35C0000]    <1>       mov    [drv], dl
21441 000062B5 BE[D4060100]      <1>         mov     esi, msg_CRLF_temp
21442 000062BA E89E000000        <1>       call   print_msg
21443                            <1>
21444 000062BF 8A15[F35C0000]    <1>       mov    dl, [drv]
21445 000062C5 E8A6090000        <1>       call   change_current_drive
21446 000062CA 730C              <1>       jnc    short _start_mainprog
21447                            <1>
21448                            <1> _drv_not_ready_error:
21449 000062CC BE[8F090100]      <1>       mov    esi, msgl_drv_not_ready
21450 000062D1 E887000000        <1>       call   print_msg
21451 000062D6 EB63              <1>         jmp     _end_of_mainprog
21452                            <1>
21453                            <1> _start_mainprog:
21454                            <1>       ; 07/01/2017
21455                            <1>       ; 07/05/2016
21456                            <1>       ; 02/05/2016
21457                            <1>       ; 24/04/2016
21458                            <1>       ; Retro UNIX 386 v1, 'sys_init' (u0.s)
21459                            <1>       ; 23/06/2015
21460                            <1>
21461                            <1>       ; 02/05/2016
21462                            <1>       ; 24/04/2016
21463 000062D8 66B80100          <1>       mov    ax, 1
21464 000062DC A2[B3030300]      <1>       mov    [u.uno], al
21465 000062E1 66A3[4E030300]    <1>       mov    [mpid], ax
21466 000062E7 66A3[20000300]    <1>       mov    [p.pid], ax
21467 000062ED A2[B0000300]      <1>       mov    [p.stat], al
21468 000062F2 C605[A8030300]04  <1>       mov    byte [u.quant], time_count  ; 07/01/2017
21469                            <1>       ;
21470 000062F9 A1[20520100]      <1>       mov    eax, [k_page_dir]
21471 000062FE A3[B8030300]      <1>       mov    [u.pgdir], eax ; reset
21472                            <1>       ;
21473 00006303 E872E8FFFF        <1>       call   allocate_page
21474 00006308 0F82A3000000      <1>       jc     panic
21475 0000630E A3[B4030300]      <1>       mov    [u.upage], eax ; user structure page
21476 00006313 A3[C0000300]      <1>       mov    [p.upage], eax
21477 00006318 E8D7E8FFFF        <1>       call   clear_page
21478                            <1>       ;
21479                            <1>       ; 24/08/2015
21480 0000631D FE0D[5B030300]    <1>       dec    byte [sysflg] ; FFh = ready for system call
21481                            <1>                            ; 0 = executing a system call
21482                            <1>       ; 13/04/2016
21483                            <1>       ; Clear Environment Variables Page/Area
21484 00006323 BF00300900        <1>       mov    edi, Env_Page ; 93000h
21485 00006328 B980000000        <1>       mov    ecx, Env_Page_Size / 4    ; 512/4  (4096/4)

21486 0000632D 31C0              <1>       xor    eax, eax
21487 0000632F F3AB              <1>       rep    stosd
21488                            <1>
21489                            <1>       ; 14/04/2016
21490 00006331 E8DE320000        <1>       call   mainprog_startup_configuration
21491                            <1>
21492 00006336 E8760A0000        <1>         call    dos_prompt
21493                            <1>
21494                            <1> _end_of_mainprog:
21495 0000633B BE[D4060100]      <1>         mov     esi, msg_CRLF_temp
21496 00006340 E818000000        <1>       call   print_msg
21497 00006345 BE[DA060100]      <1>       mov    esi, mainprog_Version
21498 0000634A E80E000000        <1>       call   print_msg
21499                            <1>       ; 24/01/2016
21500 0000634F 28E4              <1>       sub    ah, ah
21501 00006351 E8C0A8FFFF        <1>       call   int16h ; call getch
21502 00006356 E9A0ADFFFF        <1>       jmp    cpu_reset
21503                            <1>
21504 0000635B EBFE              <1> infinitiveloop: jmp short infinitiveloop
21505                            <1>
21506                            <1> print_msg:
21507                            <1>       ; 13/05/2016
21508                            <1>       ; 04/01/2016
21509                            <1>       ; 01/07/2015
21510                            <1>       ; 13/03/2015 (Retro UNIX 386 v1)
21511                            <1>       ; 07/03/2014 (Retro UNIX 8086 v1)
21512                            <1>       ; (Modified registers: EAX, EBX, ECX, EDX, ESI, EDI)
21513                            <1>       ;
21514 0000635D 8A3D[4E520100]    <1>       mov    bh, [ACTIVE_PAGE] ; 04/01/2016 (ptty)
21515                            <1>       ;mov   bl, 07h ; Black background, light gray forecolor
21516                            <1>
21517 00006363 AC                <1>       lodsb
21518                            <1> pmsg1:
21519 00006364 56                <1>       push   esi
21520                            <1>       ;mov   bh, [ACTIVE_PAGE] ; 04/01/2016 (ptty)
21521 00006365 B307              <1>       mov    bl, 07h ; Black background, light gray forecolor
21522 00006367 E846B9FFFF        <1>       call   _write_tty
```

```
21523 0000636C 5E                  <1>       pop    esi
21524 0000636D AC                  <1>       lodsb
21525 0000636E 20C0                <1>       and    al, al
21526 00006370 75F2                <1>       jnz    short pmsg1
21527 00006372 C3                  <1>       retn
21528                              <1>
21529                              <1> clear_screen:
21530                              <1>       ; 13/05/2016
21531                              <1>       ; 30/01/2016
21532                              <1>       ; 24/01/2016
21533                              <1>       ; 04/01/2016
21534 00006373 0FB61D[4E520100]    <1>       movzx  ebx, byte [ACTIVE_PAGE] ; video page number (0 to 7)
21535 0000637A 8AA3[D35E0000]      <1>       mov    ah, [ebx+vmode] ; default = 03h (80x25 text)
21536 00006380 80FC04              <1>       cmp    ah, 4
21537 00006383 7205                <1>       jb     short cls1
21538 00006385 80FC07              <1>       cmp    ah, 7
21539 00006388 7526                <1>       jne    short vga_clear
21540                              <1> cls1:
21541                              <1>       ;mov   bh, bl
21542                              <1>       ;mov   bl, 7
21543 0000638A 3A25[C25E0000]      <1>       cmp    ah, [CRT_MODE] ; current video mode ?
21544                              <1>       ;je    short cls2 ; yes (current video mode = 3)
21545                              <1>       ;;call set_mode_3 ; set video mode to 3 (& clear screen)
21546                              <1>       ;;retn
21547                              <1>       ;jmp   set_mode_3
21548 00006390 0F8526B9FFFF        <1>       jne    set_mode_3
21549                              <1> cls2:
21550 00006396 88DF                <1>       mov    bh, bl ; video page (0 to 7)
21551 00006398 B307                <1>       mov    bl, 07h ; attribute to be used on blanked line
21552 0000639A 28C0                <1>       sub    al, al ; 0 =  entire window
21553 0000639C 6631C9              <1>       xor    cx, cx
21554 0000639F 66BA4F18            <1>       mov    dx, 184Fh
21555 000063A3 E862B6FFFF          <1>       call   _scroll_up ; 24/01/2016
21556                              <1>       ;
21557                              <1>       ;mov   bh, [ACTIVE_PAGE] ; video page number (0 to 7)
21558 000063A8 6631D2              <1>       xor    dx, dx
21559 000063AB E898B9FFFF          <1>       call   _set_cpos ; 24/01/2016
21560                              <1>       ;retn
21561                              <1> vga_clear:
21562 000063B0 C3                  <1>       retn
21563                              <1>
21564                              <1> panic:
21565                              <1>       ; 13/05/2016 (TRDOS 386 = TRDOS v2)
21566                              <1>       ; 13/03/2015 (Retro UNIX 386 v1)
21567                              <1>       ; 07/03/2014 (Retro UNIX 8086 v1)
21568 000063B1 BE[72130100]        <1>       mov    esi, panic_msg
21569 000063B6 E8A2FFFFFF          <1>       call   print_msg
21570                              <1> key_to_reboot:
21571                              <1>         ; 24/01/2016
21572 000063BB 28E4                <1>         sub     ah, ah
21573 000063BD E854A8FFFF          <1>         call    int16h ; call   getch
21574                              <1>         ; wait for a character from the current tty
21575                              <1>       ;
21576 000063C2 B00A                <1>       mov    al, 0Ah
21577 000063C4 8A3D[4E520100]      <1>       mov    bh, [ptty] ; [ACTIVE_PAGE]
21578 000063CA B307                <1>       mov    bl, 07h ; Black background,
21579                              <1>                      ; light gray forecolor
21580 000063CC E8E1B8FFFF          <1>       call   _write_tty
21581 000063D1 E925ADFFFF          <1>       jmp    cpu_reset
21582                              <1>
21583                              <1> ctrlbrk:
21584                              <1>       ; 12/11/2015
21585                              <1>       ; 13/03/2015 (Retro UNIX 386 v1)
21586                              <1>       ; 06/12/2013 (Retro UNIX 8086 v1)
21587                              <1>       ;
21588                              <1>       ; INT 1Bh (control+break) handler
21589                              <1>       ;
21590                              <1>             ; Retro Unix 8086 v1 feature only!
21591                              <1>             ;
21592 000063D6 66833D[AA030300]00  <1>       cmp    word [u.intr], 0
21593 000063DE 7645                <1>       jna    short cbrk4
21594                              <1> cbrk0:
21595                              <1>       ; 12/11/2015
21596                              <1>       ; 06/12/2013
21597 000063E0 66833D[AC030300]00  <1>       cmp    word [u.quit], 0
21598 000063E8 743B                <1>       jz     short cbrk4
21599                              <1>       ;
21600                              <1>       ; 20/09/2013
21601 000063EA 6650                <1>       push   ax
21602 000063EC A0[4E520100]        <1>       mov    al, [ptty]
21603                              <1>       ;
21604                              <1>       ; 12/11/2015
21605                              <1>       ;
21606                              <1>       ; ctrl+break (EOT, CTRL+D) from serial port
21607                              <1>       ; or ctrl+break from console (pseudo) tty
21608                              <1>       ; (!redirection!)
21609                              <1>       ;
21610 000063F1 3C08                <1>       cmp    al, 8 ; serial port tty nums > 7
21611 000063F3 7211                <1>         jb      short cbrk1 ; console (pseudo) tty
21612                              <1>       ;
21613                              <1>       ; Serial port interrupt handler sets [ptty]
21614                              <1>       ; to the port's tty number (as temporary).
21615                              <1>       ;
21616                              <1>       ; If active process is using a stdin or
21617                              <1>       ; stdout redirection (by the shell),
21618                              <1>         ; console tty keyboard must be available
21619                              <1>       ; to terminate running process,
21620                              <1>       ; in order to prevent a deadlock.
21621                              <1>       ;
21622 000063F5 52                  <1>       push   edx
21623 000063F6 0FB615[B3030300]    <1>       movzx  edx, byte [u.uno]
21624 000063FD 3A82[7F000300]      <1>       cmp     al, [edx+p.ttyc-1] ; console tty (rw)
21625 00006403 5A                  <1>       pop    edx
```

```
21626 00006404 7412                <1>       je     short cbrk2
21627                              <1> cbrk1:
21628 00006406 FEC0                <1>       inc    al  ; [u.ttyp] : 1 based tty number
21629                              <1>       ; 06/12/2013
21630 00006408 3A05[94030300]      <1>       cmp    al, [u.ttyp] ; recent open tty (r)
21631 0000640E 7408                <1>       je     short cbrk2
21632 00006410 3A05[95030300]      <1>         cmp     al, [u.ttyp+1] ; recent open tty (w)
21633 00006416 750B                <1>       jne    short cbrk3
21634                              <1> cbrk2:
21635                              <1>       ;; 06/12/2013
21636                              <1>       ;mov   ax, [u.quit]
21637                              <1>       ;and   ax, ax
21638                              <1>       ;jz    short cbrk3
21639                              <1>       ;
21640 00006418 6631C0              <1>       xor    ax, ax ; 0
21641 0000641B 6648                <1>       dec    ax
21642                              <1>       ; 0FFFFh = 'ctrl+brk' keystroke
21643 0000641D 66A3[AC030300]      <1>       mov    [u.quit], ax
21644                              <1> cbrk3:
21645 00006423 6658                <1>       pop    ax
21646                              <1> cbrk4:
21647 00006425 C3                  <1>       retn
21648                                  %include 'trdosk2.s' ; 04/01/2016
21649                              <1> ; ****************************************************************************
21650                              <1> ; TRDOS386.ASM (TRDOS 386 Kernel - v2.0.0) - DRV INIT : trdosk2.s
21651                              <1> ; ----------------------------------------------------------------------------
21652                              <1> ; Last Update: 27/12/2017
21653                              <1> ; ----------------------------------------------------------------------------
21654                              <1> ; Beginning: 04/01/2016
21655                              <1> ; ----------------------------------------------------------------------------
21656                              <1> ; Assembler: NASM version 2.11 (trdos386.s)
21657                              <1> ; ----------------------------------------------------------------------------
21658                              <1> ; Derived from TRDOS Operating System v1.0 (8086) source code by Erdogan Tan
21659                              <1> ; TRDOS2.ASM (09/11/2011)
21660                              <1> ; ****************************************************************************
21661                              <1> ; DRV_INIT.ASM (c) 2009-2011 Erdogan TAN  [26/09/2009] Last Update: 07/08/2011
21662                              <1> ;
21663                              <1>
21664                              <1> ldrv_init: ; Logical Drive Initialization
21665                              <1>       ; 27/12/2017
21666                              <1>       ; 12/02/2016
21667                              <1>       ; 06/01/2016
21668                              <1>       ;      ('diskinit.inc', 'diskio.inc' integration)
21669                              <1>       ; 04/01/2016 (TRDOS 386 = TRDOS v2.0)
21670                              <1>       ; 07/08/2011
21671                              <1>       ; 20/09/2009
21672                              <1>       ; 2005
21673 00006426 0FB60D[BC520100]    <1>       movzx  ecx, byte [HF_NUM] ; number of fixed disks
21674 0000642D 80F901              <1>       cmp    cl, 1
21675 00006430 7301                <1>       jnb    short load_hd_partition_tables
21676                              <1>       ; No hard disks
21677 00006432 C3                  <1>       retn
21678                              <1> load_hd_partition_tables:
21679 00006433 8B35[C0520100]      <1>       mov    esi, [HDPM_TBL_VEC] ; primary master disk FDPT
21680 00006439 BF[E6560100]        <1>       mov    edi, PTable_hd0
21681 0000643E B280                <1>       mov    dl, 80h
21682                              <1> load_next_hd_partition_table:
21683 00006440 51                  <1>       push   ecx
21684 00006441 57                  <1>       push   edi
21685 00006442 56                  <1>       push   esi ; FDPT (+ DPTE) address
21686 00006443 8A4614              <1>       mov    al, [esi+20] ; DPTE offset 4
21687 00006446 2440                <1>       and    al, 40h ;  LBA bit (bit 6)
21688                              <1>       ;shr   al, 6
21689 00006448 A2[E7580100]        <1>       mov    [HD_LBA_yes], al
21690 0000644D E82B040000          <1>       call   load_masterboot
21691 00006452 7275                <1>       jc     short pass_pt_this_hard_disk
21692                              <1>
21693 00006454 BE[A4560100]        <1>       mov    esi, PartitionTable
21694 00006459 89F3                <1>       mov    ebx, esi
21695                              <1>       ;mov   ecx, 16
21696 0000645B B110                <1>       mov    cl, 16
21697 0000645D F3A5                <1>       rep    movsd
21698 0000645F 89DE                <1>       mov    esi, ebx
21699 00006461 C605[F55C0000]04    <1>       mov    byte [hdc], 4 ; 4 - partition index
21700                              <1> loc_validate_hdp_partition:
21701 00006468 807E0400            <1>       cmp    byte [esi+ptFileSystemID], 0
21702 0000646C 7641                <1>       jna    short loc_validate_next_hdp_partition2
21703 0000646E 56                  <1>       push   esi ; Masterboot partition table offset
21704 0000646F 52                  <1>       push   edx ; dl = Physical drive number
21705 00006470 FE05[E8580100]      <1>       inc    byte [PP_Counter]
21706 00006476 31FF                <1>         xor  edi, edi ; 0
21707                              <1>       ; Input -> ESI = PartitionTable offset
21708                              <1>       ; DL = Hard disk drive number
21709                              <1>       ; EDI = 0 -> Primary Partition
21710                              <1>       ; EDI > 0 -> Extended Partition's Start Sector
21711 00006478 E879010000          <1>       call   validate_hd_fat_partition
21712 0000647D 730A                <1>       jnc    short loc_set_valid_hdp_partition_entry
21713                              <1>       ;pop   edx
21714                              <1>       ;push  edx
21715 0000647F 8B1424              <1>       mov    edx, [esp]
21716 00006482 E8D4020000          <1>       call   validate_hd_fs_partition
21717 00006487 7224                <1>       jc     short loc_validate_next_hdp_partition1
21718                              <1> loc_set_valid_hdp_partition_entry:
21719 00006489 8A0D[D2060100]      <1>       mov    cl, [Last_DOS_DiskNo]
21720 0000648F 80C141              <1>       add    cl, 'A'
21721                              <1>       ; ESI = Logical dos drive description table address
21722 00006492 880E                <1>       mov    [esi+LD_Name], cl
21723 00006494 8A6602              <1>       mov    ah, [esi+LD_PhyDrvNo]
21724 00006497 88E0                <1>       mov    al, ah ; Physical drive number
21725 00006499 2C80                <1>       sub    al, 80h
21726 0000649B C0E002              <1>       shl    al, 2
21727 0000649E 0404                <1>       add    al, 4 ; 0 Based
21728 000064A0 2A05[F55C0000]      <1>       sub    al, [hdc] ; 4 - partition index
```

```
21729                                  <1>       ; AL = Partition entry/index, 0 based
21730                                  <1>       ;  0 -> hd 0, Partition Table offset = 0
21731                                  <1>       ; 15 -> hd 3, Partition Table offset = 3
21732                                  <1>       ;mov   [esi+LD_PartitionEntry], al
21733 000064A6 80EC7E                 <1>       sub    ah, 7Eh
21734                                  <1>       ; AH = Physical drive index, zero based
21735                                  <1>       ;  0 for drive A:, 2 for drive C:
21736                                  <1>       ;mov   [esi+LD_DParamEntry], ah
21737 000064A9 6689467C               <1>       mov    [esi+LD_PartitionEntry], ax
21738                                  <1> loc_validate_next_hdp_partition1:
21739 000064AD 5A                     <1>       pop    edx ; dl = Physical drive number
21740 000064AE 5E                     <1>       pop    esi ; Masterboot partition table offset
21741                                  <1> loc_validate_next_hdp_partition2:
21742                                  <1>       ; ESI = PartitionTable offset
21743                                  <1>       ; DL = Hard/Fixed disk drive number
21744 000064AF FE0D[F55C0000]         <1>       dec    byte [hdc] ; 4 - partition index
21745 000064B5 7412                   <1>       jz     short pass_pt_this_hard_disk
21746 000064B7 83C610                 <1>       add    esi, 16 ; 10h
21747 000064BA EBAC                   <1>       jmp    short loc_validate_hdp_partition
21748                                  <1> loc_next_hd_partition_table:
21749 000064BC FEC2                   <1>       inc    dl
21750 000064BE 83C620                 <1>       add    esi, 32 ; next FDPT address
21751 000064C1 83C740                 <1>       add    edi, 64 ; next partition table destination
21752 000064C4 E977FFFFFF             <1>         jmp     load_next_hd_partition_table
21753                                  <1> pass_pt_this_hard_disk:
21754 000064C9 5E                     <1>       pop    esi ; FDPT (+ DPTE) address
21755 000064CA 5F                     <1>       pop    edi ; Ptable_hd?
21756 000064CB 59                     <1>       pop    ecx
21757 000064CC E2EE                   <1>       loop   loc_next_hd_partition_table
21758 000064CE 803D[E8580100]01       <1>       cmp    byte [PP_Counter], 1
21759 000064D5 7301                   <1>       jnb    short load_extended_dos_partitions
21760                                  <1>       ; Empty partition table
21761 000064D7 C3                     <1>       retn
21762                                  <1> load_extended_dos_partitions:
21763 000064D8 BE[E6560100]           <1>       mov    esi, PTable_hd0
21764 000064DD BF[E6570100]           <1>       mov    edi, PTable_ep0
21765 000064E2 C605[F55C0000]80       <1>       mov    byte [hdc], 80h
21766                                  <1> next_hd_extd_partition:
21767 000064E9 56                     <1>       push   esi ; PTable_hd? offset
21768 000064EA 57                     <1>       push   edi ; PTable_ep?
21769                                  <1>       ;mov   ecx, 4
21770 000064EB B104                   <1>       mov    cl, 4
21771 000064ED 8A15[F55C0000]         <1>       mov    dl, byte [hdc]
21772                                  <1> hd_check_fs_id_05h:
21773 000064F3 8A4604                 <1>       mov    al, [esi+ptFileSystemID]
21774 000064F6 3C05                   <1>       cmp    al, 05h ; Is it an extended dos partition ?
21775 000064F8 7404                   <1>       je     short loc_set_ep_start_sector
21776 000064FA 3C0F                   <1>       cmp    al, 0Fh ; Is it an extended win4 (LBA mode) partition ?
21777 000064FC 7546                   <1>       jne    short continue_to_check_ep
21778                                  <1> loc_set_ep_start_sector:
21779 000064FE FE05[E9580100]         <1>       inc    byte [EP_Counter]
21780 00006504 88D4                   <1>       mov    ah, dl ; byte [hdc]
21781 00006506 86E0                   <1>       xchg   ah, al ; al = Drv Number, ah = Partition Identifier
21782 00006508 50                     <1>       push   eax
21783 00006509 30E4                   <1>       xor    ah, ah
21784 0000650B 2C80                   <1>       sub    al, 80h
21785 0000650D 50                     <1>       push   eax
21786 0000650E C0E002                 <1>       shl    al, 2 ; al = al * 4
21787 00006511 0FB6D8                 <1>       movzx  ebx, al
21788 00006514 81C3[EA580100]         <1>       add    ebx, EP_StartSector
21789 0000651A 8B4608                 <1>       mov    eax, [esi+ptStartSector]
21790                                  <1>         ; EAX = Extended partition's start sector
21791 0000651D 8903                   <1>         mov [ebx], eax
21792 0000651F 58                     <1>       pop    eax ; AL = Drv number - 80h, AH = 0
21793 00006520 5A                     <1>       pop    edx ; DL = Drv number, DH = Partition ID
21794 00006521 BB[E6540100]           <1>       mov    ebx, MasterBootBuff
21795 00006526 803D[E7580100]01       <1>       cmp    byte [HD_LBA_yes], 1 ; LBA ready = Yes
21796 0000652D 7240                   <1>       jb     short loc_hd_load_ep_05h
21797 0000652F 80FE05                 <1>       cmp    dh, 05h
21798 00006532 743B                   <1>       je     short loc_hd_load_ep_05h
21799                                  <1> loc_hd_load_ep_0Fh:
21800                                  <1>       ; 04/01/2016
21801 00006534 51                     <1>       push   ecx
21802 00006535 8B4E08                 <1>       mov    ecx, [esi+ptStartSector] ; sector number
21803                                  <1>       ;mov   ebx, MasterBootBuff ; buffer address
21804                                  <1>       ; LBA read/write (with private LBA function)
21805                                  <1>       ;((Retro UNIX 386 v1 - DISK I/O code by Erdogan Tan))
21806                                  <1>       ; dl = physical drive number (0,1, 80h, 81h, 82h, 83h)
21807 00006538 B41B                   <1>       mov    ah, 1Bh ; LBA read
21808 0000653A B001                   <1>       mov    al, 1 ; sector count
21809 0000653C E8C5DCFFFF             <1>       call   int13h
21810 00006541 59                     <1>       pop    ecx
21811 00006542 733F                   <1>       jnc    short loc_hd_move_ep_table
21812                                  <1> continue_to_check_ep:
21813 00006544 83C610                 <1>       add    esi, 16
21814 00006547 E2AA                   <1>       loop   hd_check_fs_id_05h
21815                                  <1> continue_check_ep_next_disk:
21816 00006549 5F                     <1>       pop    edi ; PTable_ep?
21817 0000654A 5E                     <1>       pop    esi ; PTable_hd?
21818 0000654B A0[BC520100]           <1>       mov    al, [HF_NUM] ; number of hard disks
21819 00006550 047F                   <1>       add    al, 7Fh
21820 00006552 3805[F55C0000]         <1>       cmp    [hdc], al
21821 00006558 0F8392000000           <1>       jnb    loc_validating_hd_partitions_ok
21822 0000655E 83C640                 <1>       add    esi, 64
21823 00006561 83C740                 <1>       add    edi, 64
21824 00006564 FE05[F55C0000]         <1>       inc    byte [hdc]
21825 0000656A E97AFFFFFF             <1>       jmp    next_hd_extd_partition
21826                                  <1> loc_hd_load_ep_05h:
21827 0000656F 51                     <1>       push   ecx
21828 00006570 8A7601                 <1>       mov    dh, [esi+ptBeginHead]
21829 00006573 668B4E02               <1>         mov     cx, word [esi+ptBeginSector]
21830 00006577 66B80102               <1>       mov    ax, 0201h ; Read 1 sector
21831                                  <1>       ;mov   ebx, MasterBootBuff
```

```
21832 0000657B E886DCFFFF          <1>       call   int13h
21833 00006580 59                  <1>       pop    ecx
21834 00006581 72C1                <1>       jc     short continue_to_check_ep
21835                              <1> loc_hd_move_ep_table:
21836                              <1>         ;pop edi
21837                              <1>       ;push  edi  ; PTable_ep?
21838 00006583 8B3C24              <1>       mov    edi, [esp]
21839 00006586 BE[A4560100]        <1>         mov  esi, PartitionTable ; Extended
21840 0000658B 89F3                <1>       mov    ebx, esi
21841                              <1>       ;mov   ecx, 16
21842 0000658D B110                <1>       mov    cl, 16
21843 0000658F F3A5                <1>              rep    movsd
21844 00006591 89DE                <1>       mov    esi, ebx
21845                              <1> loc_set_hde_sub_partition_count:
21846 00006593 C605[E8580100]04    <1>       mov    byte [PP_Counter], 4
21847                              <1> loc_validate_hde_partition:
21848 0000659A 807E0400            <1>       cmp    byte [esi+ptFileSystemID], 0
21849 0000659E 763F                <1>       jna    short loc_validate_next_hde_partition2
21850 000065A0 56                  <1>       push   esi ; Extended partition table offset
21851 000065A1 8A15[F55C0000]      <1>       mov    dl, byte [hdc]
21852 000065A7 0FB6C2              <1>       movzx  eax, dl
21853 000065AA 2C80                <1>       sub    al, 80h
21854 000065AC C0E002              <1>       shl    al, 2
21855                              <1>       ; 06/01/2016
21856                              <1>       ; (TRDOS v1.0 had a bug here, in 'DRV_INIT.ASM')
21857                              <1>       ; BUGFIX *
21858                              <1>       ;mov   ecx, eax
21859 000065AF 88C1                <1>       mov    cl, al
21860 000065B1 80C104              <1>       add    cl, 4
21861 000065B4 2A0D[E8580100]      <1>       sub    cl, [PP_Counter] ; 4 to 1
21862                              <1>       ; CL = Partition entry/index, 0 based
21863                              <1>         ;  0 -> hd 0, Partition Table offset = 0
21864                              <1>         ; 15 -> hd 3, Partition Table offset = 3
21865 000065BA 88D5                <1>            mov    ch, dl
21866 000065BC 80ED7E              <1>       sub    ch, 7Eh ;
21867                              <1>       ; CH = Physical drive index, zero based
21868                              <1>       ;  0 for drive A:, 2 for drive C:
21869                              <1>       ; BUGFIX *
21870 000065BF 51                  <1>       push   ecx ; *
21871 000065C0 BF[EA580100]        <1>       mov    edi, EP_StartSector
21872 000065C5 01C7                <1>       add    edi, eax
21873                              <1>       ; Input -> ESI = PartitionTable offset
21874                              <1>       ; DL = Hard disk drive number
21875                              <1>       ; EDI = Extended partition start sector pointer
21876 000065C7 E82A000000          <1>       call   validate_hd_fat_partition
21877 000065CC 59                  <1>       pop    ecx ; *
21878 000065CD 720F                <1>       jc     short loc_validate_next_hde_partition1
21879                              <1> loc_set_valid_hde_partition_entry:
21880                              <1>       ; 06/01/2016 (TRDOS v2.0)
21881                              <1>       ; BUGFIX *
21882                              <1>       ;mov   [esi+LD_PartitionEntry], cl
21883                              <1>       ;mov   [esi+LD_DParamEntry], ch
21884 000065CF 66894E7C            <1>       mov    [esi+LD_PartitionEntry], cx
21885                              <1>       ;
21886 000065D3 8A0D[D2060100]      <1>       mov    cl, [Last_DOS_DiskNo]
21887 000065D9 80C141              <1>       add    cl, 'A'
21888 000065DC 880E                <1>       mov    [esi+LD_Name], cl
21889                              <1> loc_validate_next_hde_partition1:
21890 000065DE 5E                  <1>       pop    esi ; Extended partition table offset
21891                              <1> loc_validate_next_hde_partition2:
21892                              <1>       ; ESI = Extended partition table offset
21893                              <1>       ; DL = Hard disk drive number
21894 000065DF FE0D[E8580100]      <1>       dec    byte [PP_Counter]
21895 000065E5 0F845EFFFFFF        <1>       jz     continue_check_ep_next_disk
21896 000065EB 83C610              <1>       add    esi, 16 ; 10h
21897 000065EE EBAA                <1>       jmp    short loc_validate_hde_partition
21898                              <1> loc_validating_hd_partitions_ok:
21899 000065F0 A0[D2060100]        <1>       mov    al, [Last_DOS_DiskNo]
21900                              <1> loc_drv_init_retn:
21901 000065F5 C3                  <1>       retn
21902                              <1>
21903                              <1> validate_hd_fat_partition:
21904                              <1>       ; 27/12/2017
21905                              <1>       ; 12/02/2016
21906                              <1>       ; 07/01/2016 (TRDOS 386 = TRDOS v2.0)
21907                              <1>       ; 07/08/2011
21908                              <1>       ; 23/07/2011
21909                              <1>       ; Input
21910                              <1>       ;   DL = Hard/Fixed Disk Drive Number
21911                              <1>       ;   ESI = PartitionTable offset
21912                              <1>       ;   EDI = Extend. Part. Start Sector Pointer
21913                              <1>       ;   EDI = 0 -> Primary Partition
21914                              <1>       ;   byte [Last_DOS_DiskNo]
21915                              <1>       ; Output
21916                              <1>       ;  cf=0 -> Validated
21917                              <1>       ;   ESI = Logical dos drv desc. table
21918                              <1>       ;   EBX = FAT boot sector buffer
21919                              <1>       ;   byte [Last_DOS_DiskNo]
21920                              <1>       ;  cf=1 -> Not a valid FAT partition
21921                              <1>       ; EAX, EDX, ECX, EDI -> changed
21922                              <1>
21923                              <1>       ;mov   esi, PartitionTable
21924 000065F6 8A6604              <1>       mov    ah, [esi+ptFileSystemID]
21925 000065F9 B002                <1>       mov    al, 2 ; 27/12/2017
21926 000065FB 80FC06              <1>       cmp    ah, 06h ; FAT16 CHS partition
21927                              <1>       ; 12/02/2016
21928                              <1>       ;jb    short loc_not_a_valid_fat_partition2
21929 000065FE 7310                <1>       jnb    short vhdp_FAT16_32
21930                              <1>       ;
21931                              <1>       ; 27/12/2017
21932 00006600 FEC8                <1>       dec    al ; mov al, 1
21933 00006602 38C4                <1>       cmp    ah, al ; 1 ; FAT12 partition
21934 00006604 7421                <1>       je     short loc_set_valid_hd_partition_params
```

```
21935                                  <1>       ;
21936 00006606 FEC0                    <1>       inc    al ; mov al, 2
21937 00006608 80FC04                  <1>       cmp    ah, 04h ; FAT16 CHS partition (< 32MB)
21938 0000660B 741A                    <1>       je     short loc_set_valid_hd_partition_params
21939 0000660D 7716                    <1>       ja     short loc_not_a_valid_fat_partition1
21940                                  <1>       ; cf=1
21941 0000660F C3                      <1>       retn
21942                                  <1> vhdp_FAT16_32:
21943 00006610 80FC0E                  <1>       cmp    ah, 0Eh ; FAT16 LBA partition
21944 00006613 7710                    <1>       ja     short loc_not_a_valid_fat_partition1
21945 00006615 7410                    <1>       je     short loc_set_valid_hd_partition_params
21946                                  <1>       ;mov   al, 3
21947 00006617 FEC0                    <1>       inc    al
21948 00006619 80FC0B                  <1>       cmp    ah, 0Bh ; FAT32 CHS partition
21949 0000661C 7409                    <1>       je     short loc_set_valid_hd_partition_params
21950 0000661E 7206                    <1>       jb     short loc_not_a_valid_fat_partition2
21951 00006620 80FC0C                  <1>       cmp    ah, 0Ch ; FAT32 LBA partition
21952 00006623 7402                    <1>       je     short loc_set_valid_hd_partition_params
21953                                  <1> loc_not_a_valid_fat_partition1:
21954 00006625 F9                      <1>       stc
21955                                  <1> loc_not_a_valid_fat_partition2:
21956 00006626 C3                      <1>       retn
21957                                  <1>
21958                                  <1> loc_set_valid_hd_partition_params:
21959 00006627 FE05[D2060100]          <1>       inc    byte [Last_DOS_DiskNo] ; > 1
21960                                  <1>       ;
21961 0000662D 31DB                    <1>       xor    ebx, ebx
21962 0000662F 8A3D[D2060100]          <1>       mov    bh, [Last_DOS_DiskNo] ; * 256
21963 00006635 81C300010900            <1>       add    ebx, Logical_DOSDisks
21964                                  <1>       ;
21965 0000663B C6430102                <1>       mov    byte [ebx+LD_DiskType], 2
21966 0000663F 885302                  <1>       mov    byte [ebx+LD_PhyDrvNo], dl
21967                                  <1>       ;mov   byte [ebx+LD_FATType], al ; 2 or 3
21968                                  <1>       ;mov   byte [ebx+LD_FSType], ah ; 06h, 0Eh, 0Bh, 0Ch
21969 00006642 66894303                <1>       mov    word [ebx+LD_FATType], ax
21970                                  <1>       ;
21971 00006646 8B4E08                  <1>       mov    ecx, [esi+ptStartSector]
21972 00006649 09FF                    <1>       or     edi, edi
21973 0000664B 7402                    <1>       jz     short pass_hd_FAT_ep_start_sector_adding
21974                                  <1> loc_add_hd_FAT_ep_start_sector:
21975 0000664D 030F                    <1>       add    ecx, [edi]
21976                                  <1> pass_hd_FAT_ep_start_sector_adding:
21977 0000664F 894B6C                  <1>       mov    [ebx+LD_StartSector], ecx
21978                                  <1> loc_hd_FAT_logical_drv_init:
21979 00006652 89DD                    <1>       mov    ebp, ebx
21980                                  <1>       ;mov   dl, [ebx+LD_PhyDrvNo]
21981 00006654 A0[E7580100]            <1>       mov    al, [HD_LBA_yes] ; 07/01/2016
21982 00006659 884305                  <1>       mov    [ebx+LD_LBAYes], al
21983 0000665C BB[FA580100]            <1>       mov    ebx, DOSBootSectorBuff ; buffer address
21984 00006661 08C0                    <1>       or     al, al
21985 00006663 740C                    <1>       jz     short loc_hd_FAT_drv_init_load_bs_chs
21986                                  <1> loc_hd_FAT_drv_init_load_bs_lba:
21987                                  <1>       ; DL = Physical drive number
21988                                  <1>       ;mov   ecx, [esi+ptStartSector] ; sector number
21989                                  <1>       ;mov   ebx, DOSBootSectorBuff ; buffer address
21990                                  <1>       ; LBA read/write (with private LBA function)
21991                                  <1>       ;((Retro UNIX 386 v1 - DISK I/O code by Erdogan Tan))
21992                                  <1>       ; dl = physical drive number (0,1, 80h, 81h, 82h, 83h)
21993 00006665 B41B                    <1>       mov    ah, 1Bh ; LBA read
21994 00006667 B001                    <1>       mov    al, 1 ; sector count
21995 00006669 E898DBFFFF              <1>       call   int13h
21996 0000666E 7313                    <1>       jnc    short loc_hd_drv_FAT_boot_validation
21997                                  <1> loc_not_a_valid_fat_partition3:
21998 00006670 C3                      <1>       retn
21999                                  <1> loc_hd_FAT_drv_init_load_bs_chs:
22000 00006671 8A7601                  <1>       mov    dh, [esi+ptBeginHead]
22001 00006674 668B4E02                <1>       mov    cx, [esi+ptBeginSector]
22002 00006678 66B80102                <1>       mov    ax, 0201h ; Read 1 sector
22003                                  <1>       ;mov   ebx, DOSBootSectorBuff
22004 0000667C E885DBFFFF              <1>       call   int13h
22005 00006681 72ED                    <1>       jc     short loc_not_a_valid_fat_partition3
22006                                  <1> loc_hd_drv_FAT_boot_validation:
22007                                  <1>       ;mov   esi, DOSBootSectorBuff
22008 00006683 89DE                    <1>       mov    esi, ebx
22009 00006685 6681BEFE01000055AA      <1>       cmp    word [esi+BS_Validation], 0AA55h
22010 0000668E 7512                    <1>       jne    short loc_not_a_valid_fat_partition4
22011 00006690 807E15F8                <1>       cmp    byte [esi+BPB_Media], 0F8h
22012 00006694 750C                    <1>       jne    short loc_not_a_valid_fat_partition4
22013                                  <1>
22014                                  <1>       ; 27/12/2017
22015 00006696 807D0303                <1>       cmp    byte [ebp+LD_FATType], 3
22016 0000669A 7508                    <1>       jne    short loc_hd_FAT16_BPB
22017                                  <1>
22018                                  <1> loc_hd_drv_FAT32_boot_validation:
22019 0000669C 807E4229                <1>       cmp    byte [esi+BS_FAT32_BootSig], 29h
22020 000066A0 7416                    <1>       je     short loc_hd_FAT32_BPB
22021                                  <1>
22022                                  <1> loc_not_a_valid_fat_partition4:
22023 000066A2 F9                      <1>       stc
22024 000066A3 C3                      <1>       retn
22025                                  <1>
22026                                  <1> loc_hd_FAT16_BPB:
22027 000066A4 807E2629                <1>       cmp    byte [esi+BS_BootSig], 29h
22028 000066A8 75F8                    <1>       jne    short loc_not_a_valid_fat_partition4
22029                                  <1>
22030 000066AA 66837E1600              <1>       cmp    word [esi+BPB_FATSz16], 0
22031 000066AF 7607                    <1>       jna    short loc_hd_big_FAT16_BPB
22032 000066B1 B920000000              <1>       mov    ecx, 32
22033 000066B6 EB05                    <1>       jmp    short loc_hd_move_FAT_BPB
22034                                  <1>
22035                                  <1> loc_hd_FAT32_BPB:
22036                                  <1>       ;cmp   word [esi+BPB_FATSz16], 0
22037                                  <1>       ;ja    short loc_not_a_valid_fat_partition4
```

```
                                <1> loc_hd_big_FAT16_BPB:
000066B8 B92D000000             <1>         mov    ecx, 45
                                <1> loc_hd_move_FAT_BPB:
000066BD 89EF                   <1>         mov    edi, ebp
                                <1>         ;mov   esi, ebx ; Boot sector
000066BF 57                     <1>         push   edi
000066C0 83C706                 <1>         add    edi, LD_BPB
000066C3 F366A5                 <1>         rep    movsw
000066C6 5E                     <1>         pop    esi
000066C7 0FB74614               <1>         movzx  eax, word [esi+LD_BPB+BPB_RsvdSecCnt]
000066CB 03466C                 <1>         add    eax, [esi+LD_StartSector]
000066CE 894660                 <1>         mov    [esi+LD_FATBegin], eax
000066D1 807E0303               <1>         cmp    byte [esi+LD_FATType], 3
000066D5 7224                   <1>         jb     short loc_set_FAT16_RootDirLoc
                                <1> loc_set_FAT32_RootDirLoc:
000066D7 8B462A                 <1>         mov    eax, [esi+LD_BPB+BPB_FATSz32]
000066DA 0FB65E16               <1>           movzx      ebx, byte [esi+LD_BPB+BPB_NumFATs]
000066DE F7E3                   <1>         mul    ebx
000066E0 034660                 <1>         add    eax, [esi+LD_FATBegin]
                                <1> loc_set_FAT32_data_begin:
000066E3 894668                 <1>         mov    [esi+LD_DATABegin], eax
000066E6 894664                 <1>         mov    [esi+LD_ROOTBegin], eax
                                <1>         ; If Root Directory Cluster <> 2 then
                                <1>         ; change the beginning sector value
                                <1>         ; of the root dir by adding sector offset.
000066E9 8B4632                 <1>         mov    eax, [esi+LD_BPB+BPB_RootClus]
000066EC 83E802                 <1>         sub    eax, 2
000066EF 7442                   <1>         jz     short short loc_set_32bit_FAT_total_sectors
                                <1>         ;movzx ebx, byte [esi+LD_BPB+BPB_SecPerClust]
000066F1 8A5E13                 <1>         mov    bl, byte [esi+LD_BPB+BPB_SecPerClust]
000066F4 F7E3                   <1>         mul    ebx
000066F6 014664                 <1>         add    [esi+LD_ROOTBegin], eax
000066F9 EB38                   <1>         jmp    short loc_set_32bit_FAT_total_sectors
                                <1>         ;
                                <1> loc_set_FAT16_RootDirLoc:
000066FB 0FB64616               <1>         movzx  eax, byte [esi+LD_BPB+BPB_NumFATs]
000066FF 0FB7561C               <1>         movzx  edx, word [esi+LD_BPB+BPB_FATSz16]
00006703 F7E2                   <1>         mul    edx
00006705 034660                 <1>         add    eax, [esi+LD_FATBegin]
00006708 894664                 <1>         mov    [esi+LD_ROOTBegin], eax
                                <1> loc_set_FAT16_data_begin:
0000670B 894668                 <1>         mov    [esi+LD_DATABegin], eax
0000670E B820000000             <1>         mov    eax, 20h  ; Size of a directory entry
                                <1>         ;movzx edx, word [esi+LD_BPB+BPB_RootEntCnt]
00006713 668B5617               <1>           mov     dx, [esi+LD_BPB+BPB_RootEntCnt]
00006717 F7E2                   <1>           mul  edx
                                <1>         ;mov   ecx, 511
00006719 66B9FF01               <1>         mov    cx, 511
0000671D 01C8                   <1>         add    eax, ecx
0000671F 41                     <1>         inc    ecx ; 512
00006720 F7F1                   <1>         div    ecx
00006722 014668                 <1>         add    [esi+LD_DATABegin], eax
00006725 0FB74619               <1>         movzx  eax, word [esi+LD_BPB+BPB_TotalSec16]
00006729 6685C0                 <1>         test   ax, ax
0000672C 7405                   <1>         jz     short loc_set_32bit_FAT_total_sectors
                                <1> loc_set_16bit_FAT_total_sectors:
0000672E 894670                 <1>         mov    [esi+LD_TotalSectors], eax
00006731 EB06                   <1>         jmp    short loc_set_hd_FAT_cluster_count
                                <1> loc_set_32bit_FAT_total_sectors:
00006733 8B4626                 <1>         mov    eax, [esi+LD_BPB+BPB_TotalSec32]
00006736 894670                 <1>         mov    [esi+LD_TotalSectors], eax
                                <1> loc_set_hd_FAT_cluster_count:
00006739 8B5668                 <1>         mov    edx, [esi+LD_DATABegin]
0000673C 2B566C                 <1>         sub    edx, [esi+LD_StartSector]
0000673F 29D0                   <1>         sub    eax, edx
00006741 31D2                   <1>         xor    edx, edx ; 0
00006743 0FB64E13               <1>           movzx   ecx, byte [esi+LD_BPB+BPB_SecPerClust]
00006747 F7F1                   <1>           div  ecx
00006749 894678                 <1>         mov    [esi+LD_Clusters], eax
                                <1>         ; Maximum Valid Cluster Number= EAX +1
                                <1>         ; with 2 reserved clusters= EAX +2
                                <1> loc_set_hd_FAT_fs_free_sectors:
                                <1>         ;mov   dword [esi+LD_FreeSectors], 0
0000674C E859010000             <1>         call   get_free_FAT_sectors
00006751 7207                   <1>         jc     short loc_validate_hd_FAT_partition_retn
00006753 894674                 <1>         mov    [esi+LD_FreeSectors], eax
00006756 C6467E06               <1>         mov    byte [esi+LD_MediaChanged], 6  ; Volume Name Reset
                                <1>         ;mov   cl, [Last_DOS_DiskNo]
                                <1>         ;add   cl, 'A'
                                <1>         ;mov   [esi+LD_FS_Name], cl
                                <1>
                                <1> loc_validate_hd_FAT_partition_retn:
0000675A C3                     <1>         retn
                                <1>
                                <1> validate_hd_fs_partition:
                                <1>         ; 09/12/2017
                                <1>         ; 13/02/2016
                                <1>         ; 10/01/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>         ; 29/01/2011
                                <1>         ; 23/07/2011
                                <1>         ; Input
                                <1>         ;   DL = Hard/Fixed Disk Drive Number
                                <1>         ;   ESI = PartitionTable offset
                                <1>         ;   byte [Last_DOS_DiskNo]
                                <1>         ; Output
                                <1>         ;  cf=0 -> Validated
                                <1>         ;   ESI = Logical dos drv desc. table
                                <1>         ;   EBX = Singlix FS boot sector buffer
                                <1>         ;   byte [Last_DOS_DiskNo]
                                <1>         ;  cf=1 -> Not a valid 'Singlix FS' partition
                                <1>         ; EAX, EDX, ECX, EDI -> changed
                                <1>
                                <1>         ;mov   esi, PartitionTable
```

```
22141 0000675B 8A6604              <1>       mov    ah, [esi+ptFileSystemID]
22142 0000675E 80FCA1              <1>       cmp    ah, 0A1h ; SINGLIX FS1 (trfs1) partition
22143 00006761 7549                <1>       jne    short loc_validate_hd_fs_partition_stc_retn
22144                              <1> loc_set_valid_hd_fs_partition_params:
22145 00006763 FE05[D2060100]      <1>       inc    byte [Last_DOS_DiskNo] ; > 1
22146 00006769 30C0                <1>       xor    al, al ; mov al, 0
22147                              <1>       ;mov   [drv], dl
22148 0000676B 29DB                <1>       sub    ebx, ebx ; 0
22149 0000676D 8A3D[D2060100]      <1>       mov    bh, [Last_DOS_DiskNo]
22150 00006773 81C300010900        <1>       add    ebx, Logical_DOSDisks
22151 00006779 C6430102            <1>       mov    byte [ebx+LD_DiskType], 2
22152 0000677D 885302              <1>       mov    [ebx+LD_PhyDrvNo], dl
22153                              <1>       ;mov   [ebx+LD_FATType], al ; 0
22154                              <1>       ;mov   [ebx+LD_FSType], ah
22155 00006780 66894303            <1>       mov    [ebx+LD_FATType], ax
22156                              <1>       ;mov   eax, [esi+ptStartSector]
22157                              <1>       ;mov   [ebx+LD_StartSector], eax
22158                              <1> loc_hd_fs_logical_drv_init:
22159 00006784 89DD                <1>       mov    ebp, ebx ; 10/01/2016
22160                              <1>       ;mov   dl, [ebx+LD_PhyDrvNo]
22161 00006786 A0[E7580100]        <1>       mov    al, [HD_LBA_yes] ; 10/01/2016
22162 0000678B 884305              <1>       mov    [ebx+LD_LBAYes], al
22163 0000678E 89DE                <1>       mov    esi, ebx
22164 00006790 BB[FA580100]        <1>       mov    ebx, DOSBootSectorBuff ; buffer address
22165 00006795 08C0                <1>       or     al, al
22166 00006797 7515                <1>       jnz    short loc_hd_fs_drv_init_load_bs_lba
22167                              <1> loc_hd_fs_drv_init_load_bs_chs:
22168 00006799 8A7601              <1>       mov    dh, [esi+ptBeginHead]
22169 0000679C 668B4E02            <1>       mov    cx, [esi+ptBeginSector]
22170 000067A0 66B80102            <1>       mov    ax, 0201h ; Read 1 sector
22171                              <1>       ;mov   ebx, DOSBootSectorBuff
22172 000067A4 E85DDAFFFF          <1>       call   int13h
22173 000067A9 7311                <1>       jnc    short loc_hd_drv_fs_boot_validation
22174                              <1> loc_validate_hd_fs_partition_err_retn:
22175 000067AB C3                  <1>       retn
22176                              <1> loc_validate_hd_fs_partition_stc_retn:
22177 000067AC F9                  <1>       stc
22178 000067AD C3                  <1>       retn
22179                              <1> loc_hd_fs_drv_init_load_bs_lba:
22180                              <1>       ; DL = Physical drive number
22181                              <1>       ;mov   esi, ebx
22182 000067AE 8B4E08              <1>       mov    ecx, [esi+ptStartSector] ; sector number
22183                              <1>       ;mov   ebx, DOSBootSectorBuff ; buffer address
22184                              <1>       ; LBA read/write (with private LBA function)
22185                              <1>       ;((Retro UNIX 386 v1 - DISK I/O code by Erdogan Tan))
22186                              <1>       ; dl = physical drive number (0,1, 80h, 81h, 82h, 83h)
22187 000067B1 B41B                <1>       mov    ah, 1Bh ; LBA read
22188 000067B3 B001                <1>       mov    al, 1 ; sector count
22189 000067B5 E84CDAFFFF          <1>       call   int13h
22190 000067BA 72EF                <1>       jc     short loc_validate_hd_fs_partition_err_retn
22191                              <1> loc_hd_drv_fs_boot_validation:
22192                              <1>       ;mov   esi, DOSBootSectorBuff
22193 000067BC 89DE                <1>       mov    esi, ebx ; Boot sector buffer
22194 000067BE 6681BEFE01000055AA  <1>       cmp    word [esi+BS_Validation], 0AA55h
22195 000067C7 75E3                <1>       jne    short loc_validate_hd_fs_partition_stc_retn
22196                              <1>         ;
22197                              <1>       ;Singlix FS Extensions to TR-DOS (7/6/2009)
22198 000067C9 66817E035346        <1>       cmp    word [esi+bs_FS_Identifier], 'SF'
22199 000067CF 75DB                <1>       jne    short loc_validate_hd_fs_partition_stc_retn
22200                              <1>         ;'A1h' check is not necessary
22201                              <1>       ;  if 'FS' check is passed as OK/Yes.
22202 000067D1 807E09A1            <1>       cmp    byte [esi+bs_FS_PartitionID], 0A1h
22203 000067D5 75D5                <1>       jne    short loc_validate_hd_fs_partition_stc_retn
22204                              <1>       ;
22205 000067D7 89EF                <1>       mov    edi, ebp ; 10/01/2016
22206                              <1>       ;
22207 000067D9 8A462D              <1>       mov    al, byte [esi+bs_FS_LBA_Ready]
22208 000067DC 884705              <1>       mov    [edi+LD_FS_LBAYes], al
22209                              <1>       ;
22210                              <1>       ; 03/01/2010 CHS -> DOS FAT/BPB compatibility fix
22211 000067DF 8A4608              <1>       mov    al, [esi+bs_FS_MediaAttrib]
22212 000067E2 884706              <1>       mov    byte [edi+LD_FS_MediaAttrib], al
22213                              <1>       ;
22214 000067E5 8A460A              <1>       mov    al, [esi+bs_FS_VersionMaj]
22215 000067E8 884707              <1>       mov    [edi+LD_FS_VersionMajor], al
22216                              <1>       ;
22217 000067EB 668B4606            <1>       mov    ax, [esi+bs_FS_BytesPerSec]
22218 000067EF 66894711            <1>       mov    [edi+LD_FS_BytesPerSec], ax
22219 000067F3 8A462E              <1>       mov    al, [esi+bs_FS_SecPerTrack]
22220 000067F6 30E4                <1>       xor    ah, ah ; 09/12/2017
22221 000067F8 6689471E            <1>       mov    [edi+LD_FS_SecPerTrack], ax
22222 000067FC 8A462F              <1>       mov    al, [esi+bs_FS_Heads]
22223 000067FF 66894720            <1>       mov    [edi+LD_FS_NumHeads], ax
22224                              <1>       ;
22225 00006803 8B4628              <1>       mov    eax, [esi+bs_FS_UnDelDirD]
22226 00006806 894722              <1>       mov    [edi+LD_FS_UnDelDirD], eax
22227 00006809 8B5618              <1>       mov    edx, [esi+bs_FS_MATLocation]
22228 0000680C 89570C              <1>       mov    [edi+LD_FS_MATLocation], edx
22229 0000680F 8B461C              <1>       mov    eax, [esi+bs_FS_RootDirD]
22230 00006812 894708              <1>       mov    [edi+LD_FS_RootDirD], eax
22231 00006815 8B460C              <1>       mov    eax, [esi+bs_FS_BeginSector]
22232 00006818 89476C              <1>       mov    [edi+LD_FS_BeginSector], eax
22233 0000681B 8B4710              <1>       mov    eax, [edi+bs_FS_VolumeSize]
22234 0000681E 894770              <1>       mov    [edi+LD_FS_VolumeSize], eax
22235                              <1>       ;
22236 00006821 89D0                <1>       mov    eax, edx ; [edi+LD_FS_MATLocation]
22237 00006823 03476C              <1>       add    eax, [edi+LD_FS_BeginSector]
22238 00006826 89FE                <1>       mov    esi, edi
22239                              <1> mread_hd_fs_MAT_sector:
22240                              <1>        ;mov  ebx, DOSBootSectorBuff
22241 00006828 B901000000          <1>       mov    ecx, 1
22242 0000682D E8A6890000          <1>       call   disk_read
22243 00006832 7248                <1>       jc     short loc_validate_hd_fs_partition_retn
```

```
22244                                   <1>       ; EDI will not be changed
22245 00006834 89DE                     <1>       mov    esi, ebx
22246                                   <1> use_hdfs_mat_sector_params:
22247 00006836 8B460C                   <1>       mov    eax, [esi+FS_MAT_DATLocation]
22248 00006839 894714                   <1>       mov    [edi+LD_FS_DATLocation], eax
22249 0000683C 8B4610                   <1>       mov    eax, [esi+FS_MAT_DATScount]
22250 0000683F 894718                   <1>       mov    [edi+LD_FS_DATSectors], eax
22251 00006842 8B4614                   <1>       mov    eax, [esi+FS_MAT_FreeSectors]
22252 00006845 894774                   <1>         mov     [edi+LD_FS_FreeSectors], eax
22253 00006848 8B4618                   <1>       mov    eax, [esi+FS_MAT_FirstFreeSector]
22254 0000684B 894778                   <1>       mov    [edi+LD_FS_FirstFreeSector], eax
22255 0000684E 8B4708                   <1>       mov    eax, [edi+LD_FS_RootDirD]
22256 00006851 03476C                   <1>       add    eax, [edi+LD_FS_BeginSector]
22257 00006854 89FE                     <1>       mov    esi, edi
22258                                   <1> read_hd_fs_RDT_sector:
22259 00006856 BB[FA580100]             <1>       mov    ebx, DOSBootSectorBuff
22260                                   <1>       ;mov   ecx, 1
22261 0000685B B101                     <1>       mov    cl, 1
22262 0000685D E876890000               <1>       call   disk_read
22263 00006862 7218                     <1>       jc     short loc_validate_hd_fs_partition_retn
22264                                   <1>       ; EDI will not be changed
22265 00006864 89DE                     <1>       mov    esi, ebx
22266                                   <1> use_hdfs_RDT_sector_params:
22267 00006866 8B461C                   <1>       mov    eax, [esi+FS_RDT_VolumeSerialNo]
22268 00006869 894728                   <1>       mov    [edi+LD_FS_VolumeSerial], eax
22269 0000686C 57                       <1>       push   edi
22270                                   <1>       ;mov   ecx, 16
22271 0000686D B110                     <1>       mov    cl, 16
22272 0000686F 83C640                   <1>       add    esi, FS_RDT_VolumeName
22273 00006872 83C72C                   <1>       add    edi, LD_FS_VolumeName
22274 00006875 F3A5                     <1>       rep    movsd ; 64 bytes
22275 00006877 5E                       <1>       pop    esi
22276                                   <1>              ; Volume Name Reset
22277 00006878 C6467E06                 <1>         mov     byte [esi+LD_FS_MediaChanged], 6
22278                                   <1>       ;
22279                                   <1>         ;mov cl, [Last_DOS_DiskNo]
22280                                   <1>       ;add   cl, 'A'
22281                                   <1>       ;mov   [esi+LD_FS_Name], cl
22282                                   <1>
22283                                   <1> loc_validate_hd_fs_partition_retn:
22284 0000687C C3                       <1>       retn
22285                                   <1>
22286                                   <1> load_masterboot:
22287                                   <1>       ; 10/01/2016 (TRDOS 386 = TRDOS v2.0)
22288                                   <1>       ; 2005 - 2011
22289                                   <1>       ; input -> DL = drive number
22290 0000687D B40D                     <1>       mov    ah, 0Dh ; Alternate disk reset
22291 0000687F E882D9FFFF               <1>       call   int13h
22292 00006884 7301                     <1>       jnc    short pass_reset_error
22293                                   <1> harddisk_error:
22294 00006886 C3                       <1>       retn
22295                                   <1> pass_reset_error:
22296 00006887 BB[E6540100]             <1>       mov    ebx, MasterBootBuff
22297 0000688C 66B80102                 <1>       mov    ax, 0201h
22298 00006890 66B90100                 <1>       mov    cx, 1
22299 00006894 30F6                     <1>       xor    dh, dh
22300 00006896 E86BD9FFFF               <1>       call   int13h
22301 0000689B 72E9                     <1>       jc     short harddisk_error
22302                                   <1>       ;
22303 0000689D 66813D[E4560100]55-      <1>       cmp    word [MBIDCode], 0AA55h
22304 000068A5 AA                       <1>
22305 000068A6 7401                     <1>       je     short load_masterboot_ok
22306 000068A8 F9                       <1>       stc
22307                                   <1> load_masterboot_ok:
22308 000068A9 C3                       <1>       retn
22309                                   <1>
22310                                   <1> get_free_FAT_sectors:
22311                                   <1>       ; 21/12/2017
22312                                   <1>       ; 29/02/2016
22313                                   <1>       ; 13/02/2016
22314                                   <1>       ; 04/02/2016
22315                                   <1>       ; 07/01/2016 (TRDOS 386 = TRDOS v2.0)
22316                                   <1>       ; 11/07/2010
22317                                   <1>       ; 21/06/2009
22318                                   <1>       ; INPUT: ESI = Logical DOS Drive Description Table address
22319                                   <1>       ; OUTPUT: STC => Error
22320                                   <1>         ;    cf = 0 and EAX = Free FAT sectors
22321                                   <1>       ; Also, related parameters and FAT buffer will be reset and updated
22322                                   <1>
22323 000068AA 31C0                     <1>       xor    eax, eax
22324                                   <1>       ;mov   [esi+LD_FreeSectors], eax ; Reset
22325                                   <1>
22326 000068AC 807E0302                 <1>         cmp     byte [esi+LD_FATType], 2
22327 000068B0 7654                     <1>       jna    short loc_gfc_get_fat_free_clusters
22328                                   <1>
22329                                   <1>       ; 29/02/2016
22330 000068B2 48                       <1>       dec    eax ; 0FFFFFFFFh
22331 000068B3 89463A                   <1>       mov    [esi+LD_BPB+BPB_Reserved], eax ; Free cluster count (reset)
22332 000068B6 89463E                   <1>       mov    [esi+LD_BPB+BPB_Reserved+4], eax ; First Free Cluster (reset)
22333 000068B9 40                       <1>       inc    eax ; 0
22334                                   <1>       ;
22335 000068BA 668B4636                 <1>       mov    ax, [esi+LD_BPB+BPB_FSInfo]
22336 000068BE 03466C                   <1>       add    eax, [esi+LD_StartSector]
22337                                   <1>
22338 000068C1 BB[FA580100]             <1>       mov    ebx, DOSBootSectorBuff
22339 000068C6 B901000000               <1>       mov    ecx, 1
22340 000068CB E808890000               <1>       call   disk_read
22341 000068D0 7301                     <1>       jnc    short loc_gfc_check_fsinfo_signs
22342                                   <1> retn_gfc_get_fsinfo_sec:
22343 000068D2 C3                       <1>       retn
22344                                   <1>
22345                                   <1> loc_gfc_check_fsinfo_signs:
22346 000068D3 BB[FA580100]             <1>       mov    ebx, DOSBootSectorBuff ; 13/02/2016
```

```
22347 000068D8 813B52526141        <1>         cmp     dword [ebx], 41615252h
22348 000068DE 7524                <1>       jne    short retn_gfc_get_fsinfo_stc
22349                              <1>       ;add   ebx, 484
22350                              <1>       ;cmp   dword [ebx], 61417272h
22351 000068E0 81BBE4010000727241- <1>       cmp    dword [ebx+484], 61417272h
22352 000068E9 61                  <1>
22353 000068EA 7518                <1>       jne    short retn_gfc_get_fsinfo_stc
22354                              <1>       ;add   ebx, 4
22355                              <1>       ;mov   eax, [ebx]
22356 000068EC 8B83E8010000        <1>       mov    eax, [ebx+488]
22357                              <1>       ; 29/02/2016
22358 000068F2 89463A              <1>       mov    [esi+LD_BPB+BPB_Reserved], eax ; Free cluster count
22359 000068F5 8B93EC010000        <1>       mov    edx,  [ebx+492]
22360 000068FB 89463E              <1>       mov    [esi+LD_BPB+BPB_Reserved+4], eax ; First Free Cluster
22361                              <1>       ; 21/12/2017
22362 000068FE 89C3                <1>       mov    ebx, eax ; (initial value = 0FFFFFFFFh)
22363 00006900 43                  <1>       inc    ebx ; 0FFFFFFFFh -> 0
22364 00006901 7513                <1>       jnz    short short retn_from_get_free_fat32_clusters
22365 00006903 C3                  <1>       retn
22366                              <1>
22367                              <1> retn_gfc_get_fsinfo_stc:
22368 00006904 F9                  <1>       stc
22369 00006905 C3                  <1>       retn
22370                              <1>
22371                              <1> loc_gfc_get_fat_free_clusters:
22372                              <1>       ;mov   eax, 2
22373 00006906 B002                <1>       mov    al, 2
22374                              <1>       ;mov   [FAT_CurrentCluster], eax
22375                              <1> loc_gfc_loop_get_next_cluster:
22376 00006908 E83A500000          <1>       call   get_next_cluster
22377 0000690D 730E                <1>       jnc    short loc_gfc_free_fat_clusters_cont
22378 0000690F 21C0                <1>       and    eax, eax
22379 00006911 7411                <1>       jz     short loc_gfc_pass_inc_free_cluster_count
22380                              <1>
22381                              <1> retn_from_get_free_fat_clusters:
22382 00006913 8B4674              <1>       mov    eax, [esi+LD_FreeSectors] ; Free clusters !
22383                              <1> retn_from_get_free_fat32_clusters:
22384 00006916 0FB65E13            <1>         movzx      ebx, byte [esi+LD_BPB+BPB_SecPerClust]
22385 0000691A F7E3                <1>             mul    ebx
22386                              <1>       ;mov   [esi+LD_FreeSectors], eax ; Free sectors
22387                              <1> retn_get_free_sectors_calc:
22388 0000691C C3                  <1>       retn
22389                              <1>
22390                              <1> loc_gfc_free_fat_clusters_cont:
22391 0000691D 09C0                <1>       or     eax, eax
22392 0000691F 7503                <1>       jnz    short loc_gfc_pass_inc_free_cluster_count
22393 00006921 FF4674              <1>       inc    dword [esi+LD_FreeSectors] ; Free clusters !
22394                              <1>
22395                              <1> loc_gfc_pass_inc_free_cluster_count:
22396                              <1>       ;mov   eax, [FAT_CurrentCluster]
22397 00006924 89C8                <1>       mov    eax, ecx ; [FAT_CurrentCluster]
22398 00006926 3B4678              <1>       cmp    eax, [esi+LD_Clusters]
22399 00006929 77E8                <1>       ja     short retn_from_get_free_fat_clusters
22400 0000692B 40                  <1>       inc    eax
22401                              <1>       ;mov   [FAT_CurrentCluster], eax
22402 0000692C EBDA                <1>       jmp    short loc_gfc_loop_get_next_cluster
22403                              <1>
22404                              <1> floppy_drv_init:
22405                              <1>       ; 09/12/2017
22406                              <1>       ; 06/07/2016
22407                              <1>       ; 10/01/2016 (TRDOS 386 = TRDOS v2.0)
22408                              <1>       ; 24/07/2011
22409                              <1>       ; 04/07/2009
22410                              <1>       ; INPUT ->
22411                              <1>       ;      DL = Drive number (0,1)
22412                              <1>       ; OUTPUT ->
22413                              <1>       ;      BL = drive name
22414                              <1>       ;      BH = drive number
22415                              <1>       ;      ESI = Logical DOS drv description table
22416                              <1>       ;      EAX = Volume serial number
22417                              <1>
22418 0000692E BE[F65C0000]        <1>       mov    esi, fd0_type ; 10/01/2016
22419 00006933 BF00010900          <1>       mov    edi, Logical_DOSDisks
22420 00006938 08D2                <1>       or     dl, dl
22421 0000693A 7407                <1>       jz     short loc_drv_init_fd0_fd1
22422 0000693C 81C700010000        <1>       add    edi, 100h
22423 00006942 46                  <1>       inc    esi ; fd1_type ; 10/01/2016
22424                              <1> loc_drv_init_fd0_fd1:
22425 00006943 C6477E00            <1>       mov    byte [edi+LD_MediaChanged], 0
22426 00006947 803E01              <1>       cmp    byte [esi], 1 ; type (>0 if it is existing)
22427                              <1>              ; 4 = 1.44 MB, 80 track, 3 1/2"
22428 0000694A 7221                <1>       jb     short read_fd_boot_sector_retn
22429 0000694C 885702              <1>       mov    [edi+LD_PhyDrvNo], dl
22430                              <1> read_fd_boot_sector:
22431 0000694F 30F6                <1>       xor    dh, dh
22432 00006951 B904000000          <1>       mov    ecx, 4 ; Retry Count
22433                              <1> read_fd_boot_sector_again:
22434 00006956 51                  <1>       push   ecx
22435                              <1>       ;mov   cx, 1
22436 00006957 B101                <1>       mov    cl, 1
22437 00006959 66B80102            <1>       mov    ax, 0201h ; Read 1 sector
22438 0000695D BB[FA580100]        <1>       mov    ebx, DOSBootSectorBuff
22439 00006962 E89FD8FFFF          <1>       call   int13h
22440 00006967 59                  <1>       pop    ecx
22441 00006968 7304                <1>       jnc    short use_fd_boot_sector_params
22442 0000696A E2EA                <1>       loop   read_fd_boot_sector_again
22443                              <1>
22444                              <1> read_fd_boot_sector_stc_retn:
22445 0000696C F9                  <1>       stc
22446                              <1> read_fd_boot_sector_retn:
22447 0000696D C3                  <1>       retn
22448                              <1>
22449                              <1> use_fd_boot_sector_params:
```

```
22450                                  <1>       ;mov   esi, DOSBootSectorBuff
22451 0000696E 89DE                    <1>       mov    esi, ebx
22452 00006970 6681BEFE01000055AA      <1>       cmp    word [esi+BS_Validation], 0AA55h
22453 00006979 75F1                    <1>       jne    short read_fd_boot_sector_stc_retn
22454 0000697B 66817E035346            <1>         cmp    word [esi+bs_FS_Identifier], 'SF'
22455 00006981 0F85A2000000            <1>         jne    use_fd_fatfs_boot_sector_params
22456                                  <1>       ;
22457 00006987 8A462D                  <1>       mov    al, [esi+bs_FS_LBA_Ready]
22458 0000698A 884705                  <1>       mov    [edi+LD_FS_LBAYes], al
22459                                  <1>       ;
22460                                  <1>       ; 03/01/2010 CHS -> DOS FAT/BPB compatibility fix
22461 0000698D 8A4608                  <1>       mov    al, [esi+bs_FS_MediaAttrib]
22462 00006990 884706                  <1>       mov    [edi+LD_FS_MediaAttrib], al
22463                                  <1>       ;
22464 00006993 8A460A                  <1>         mov  al, [esi+bs_FS_VersionMaj]
22465 00006996 884707                  <1>       mov    byte [edi+LD_FS_VersionMajor], al
22466 00006999 668B4606                <1>       mov    ax, [esi+bs_FS_BytesPerSec]
22467 0000699D 66894711                <1>       mov    [edi+LD_FS_BytesPerSec], ax
22468 000069A1 8A462E                  <1>       mov    al, [esi+bs_FS_SecPerTrack]
22469 000069A4 28E4                    <1>       sub    ah, ah ; 09/12/2017
22470 000069A6 6689471E                <1>       mov    [edi+LD_FS_SecPerTrack], ax
22471 000069AA 8A462F                  <1>       mov    al, [esi+bs_FS_Heads]
22472 000069AD 66894720                <1>       mov    [edi+LD_FS_NumHeads], ax
22473                                  <1>       ;
22474 000069B1 8B4628                  <1>       mov    eax, [esi+bs_FS_UnDelDirD]
22475 000069B4 894722                  <1>       mov    [edi+LD_FS_UnDelDirD], eax
22476 000069B7 8B4618                  <1>       mov    eax, [esi+bs_FS_MATLocation]
22477 000069BA 89470C                  <1>       mov    [edi+LD_FS_MATLocation], eax
22478 000069BD 8B461C                  <1>       mov    eax, [esi+bs_FS_RootDirD]
22479 000069C0 894708                  <1>       mov    [edi+LD_FS_RootDirD], eax
22480 000069C3 8B460C                  <1>       mov    eax, [esi+bs_FS_BeginSector]
22481 000069C6 89476C                  <1>       mov    [edi+LD_FS_BeginSector], eax
22482 000069C9 8B4610                  <1>       mov    eax, [esi+bs_FS_VolumeSize]
22483 000069CC 894770                  <1>       mov    [edi+LD_FS_VolumeSize], eax
22484                                  <1>       ;
22485 000069CF 89FE                    <1>       mov    esi, edi
22486 000069D1 8B460C                  <1>       mov    eax, [esi+LD_FS_MATLocation]
22487                                  <1>       ;add   eax, [edi+LD_FS_BeginSector]
22488                                  <1> read_fd_MAT_sector_again:
22489                                  <1>       ;mov   ebx, DOSBootSectorBuff
22490                                  <1>       ;mov   ecx, 1
22491 000069D4 B101                    <1>       mov    cl, 1
22492 000069D6 E803880000              <1>       call   chs_read
22493 000069DB 89DE                    <1>       mov    esi, ebx
22494 000069DD 7301                    <1>       jnc    short use_fdfs_mat_sector_params
22495                                  <1>       ;jmp   short read_fd_boot_sector_retn
22496 000069DF C3                      <1>       retn
22497                                  <1> use_fdfs_mat_sector_params:
22498 000069E0 8B460C                  <1>       mov    eax, [esi+FS_MAT_DATLocation]
22499 000069E3 894714                  <1>       mov    [edi+LD_FS_DATLocation], eax
22500 000069E6 8B4610                  <1>       mov    eax, [esi+FS_MAT_DATScount]
22501 000069E9 894718                  <1>       mov    [edi+LD_FS_DATSectors], eax
22502 000069EC 8B4714                  <1>       mov    eax, [edi+FS_MAT_FreeSectors]
22503 000069EF 894774                  <1>       mov    [edi+LD_FS_FreeSectors], eax
22504 000069F2 8B4618                  <1>       mov    eax, [esi+FS_MAT_FirstFreeSector]
22505 000069F5 894778                  <1>       mov    [edi+LD_FS_FirstFreeSector], eax
22506                                  <1>       ;
22507 000069F8 89FE                    <1>       mov    esi, edi
22508 000069FA 8B4608                  <1>       mov    eax, [esi+LD_FS_RootDirD]
22509                                  <1> read_fd_RDT_sector_again:
22510                                  <1>       ;mov   ebx, DOSBootSectorBuff
22511                                  <1>       ;mov   cx, 1
22512 000069FD B101                    <1>       mov    cl, 1
22513 000069FF E8DA870000              <1>       call   chs_read
22514 00006A04 89DE                    <1>       mov    esi, ebx
22515 00006A06 7220                    <1>       jc     short read_fd_RDT_sector_retn
22516                                  <1> use_fdfs_RDT_sector_params:
22517 00006A08 8B461C                  <1>       mov    eax, [esi+FS_RDT_VolumeSerialNo]
22518 00006A0B 894728                  <1>       mov    [edi+LD_FS_VolumeSerial], eax
22519 00006A0E 57                      <1>       push   edi
22520                                  <1>       ;mov   ecx, 16
22521 00006A0F B110                    <1>       mov    cl, 16
22522 00006A11 83C640                  <1>       add    esi, FS_RDT_VolumeName
22523 00006A14 83C72C                  <1>       add    edi, LD_FS_VolumeName
22524 00006A17 F3A5                    <1>       rep    movsd ; 64 bytes
22525 00006A19 5E                      <1>       pop    esi
22526 00006A1A C6460300                <1>       mov    byte [esi+LD_FATType], 0
22527 00006A1E C64604A1                <1>       mov    byte [esi+LD_FSType], 0A1h
22528 00006A22 E9A5000000              <1>         jmp    loc_cont_use_fd_boot_sector_params
22529                                  <1>
22530                                  <1> read_fd_RDT_sector_stc_retn:
22531 00006A27 F9                      <1>       stc
22532                                  <1> read_fd_RDT_sector_retn:
22533 00006A28 C3                      <1>       retn
22534                                  <1>
22535                                  <1> use_fd_fatfs_boot_sector_params:
22536 00006A29 807E2629                <1>       cmp    byte [esi+BS_BootSig], 29h
22537 00006A2D 75F8                    <1>       jne    short read_fd_RDT_sector_stc_retn
22538 00006A2F 807E15F0                <1>       cmp    byte [esi+BPB_Media], 0F0h
22539 00006A33 72F3                    <1>       jb     short read_fd_RDT_sector_retn
22540 00006A35 57                      <1>       push   edi
22541 00006A36 83C706                  <1>       add    edi, LD_BPB
22542                                  <1>       ;mov   ecx, 16
22543 00006A39 B110                    <1>       mov    cl, 16
22544 00006A3B F3A5                    <1>       rep    movsd ; 64 bytes
22545 00006A3D 5E                      <1>       pop    esi
22546 00006A3E 31C0                    <1>       xor    eax, eax
22547 00006A40 89466C                  <1>       mov    [esi+LD_StartSector], eax ; 0
22548 00006A43 668B461C                <1>       mov    ax, [esi+LD_BPB+BPB_FATSz16]
22549 00006A47 8A4E16                  <1>       mov    cl, [esi+LD_BPB+BPB_NumFATs]
22550 00006A4A F7E1                    <1>       mul    ecx
22551                                  <1>       ; edx = 0 !
22552 00006A4C 668B5614                <1>       mov    dx, [esi+LD_BPB+BPB_RsvdSecCnt]
```

```
22553 00006A50 66895660            <1>       mov    [esi+LD_FATBegin], dx
22554                              <1>       ;add   eax, edx
22555 00006A54 6601D0              <1>       add    ax, dx
22556 00006A57 894664              <1>       mov    [esi+LD_ROOTBegin], eax
22557 00006A5A 894668              <1>       mov    [esi+LD_DATABegin], eax
22558 00006A5D 668B5617            <1>       mov    dx, [esi+LD_BPB+BPB_RootEntCnt]
22559                              <1>       ;;shl  edx, 5 ; * 32 (Size of a directory entry)
22560                              <1>       ;shl   dx, 5
22561                              <1>       ;;add  edx, 511
22562                              <1>       ;add   dx, 511
22563                              <1>       ;;shr  edx, 9 ; edx = ((edx*32)+511) / 512
22564                              <1>       ;shr   dx, 9
22565 00006A61 6683C20F            <1>       add    dx, 15 ; 06/07/2016 (+(512/32)-1)
22566 00006A65 66C1EA04            <1>       shr    dx, 4 ; / 16 (==16 entries per sector)
22567 00006A69 015668              <1>       add    [esi+LD_DATABegin], edx ; + rd sectors
22568                              <1>       ;movzx eax, word [esi+LD_BPB+BPB_TotalSec16]
22569 00006A6C 668B4619            <1>       mov    ax, [esi+LD_BPB+BPB_TotalSec16]
22570 00006A70 894670              <1>       mov    [esi+LD_TotalSectors], eax
22571 00006A73 2B4668              <1>       sub    eax, [esi+LD_DATABegin]
22572                              <1>       ;movzx ecx, byte [esi+LD_BPB+BPB_SecPerClust]
22573 00006A76 8A4E13              <1>       mov    cl, [esi+LD_BPB+BPB_SecPerClust]
22574 00006A79 80F901              <1>       cmp    cl, 1
22575 00006A7C 7605                <1>       jna    short save_fd_fatfs_cluster_count
22576                              <1>       ;sub   edx, edx
22577 00006A7E 6629D2              <1>       sub    dx, dx ; 0
22578                              <1>       ;sub   dl, dl ; 06/07/2016
22579 00006A81 F7F1                <1>       div    ecx
22580                              <1> save_fd_fatfs_cluster_count:
22581 00006A83 894678              <1>       mov    [esi+LD_Clusters], eax
22582                              <1>
22583                              <1>       ; Maximum Valid Cluster Number = EAX +1
22584                              <1>       ; with 2 reserved clusters= EAX +2
22585                              <1>
22586                              <1> reset_FAT_buffer_decriptors:
22587 00006A86 29C0                <1>       sub    eax, eax ; 0
22588 00006A88 A2[FE5A0100]        <1>       mov    [FAT_BuffValidData], al ; 0
22589 00006A8D A2[FF5A0100]        <1>       mov    [FAT_BuffDrvName], al ; 0
22590 00006A92 A3[025B0100]        <1>       mov    [FAT_BuffSector], eax ; 0
22591                              <1>
22592                              <1> read_fd_FAT_sectors:
22593 00006A97 BB001C0900          <1>       mov    ebx, FAT_Buffer
22594 00006A9C 668B4614            <1>       mov    ax, [esi+LD_BPB+BPB_RsvdSecCnt]
22595                              <1>       ;mov   ecx, 3
22596 00006AA0 B103                <1>       mov    cl, 3 ; 3 sectors
22597 00006AA2 E837870000          <1>       call   chs_read
22598 00006AA7 7240                <1>       jc     short read_fd_FAT_sectors_retn
22599                              <1> use_fd_FAT_sectors:
22600 00006AA9 8A4602              <1>       mov    al, [esi+LD_PhyDrvNo]
22601 00006AAC 0441                <1>       add    al, 'A'
22602 00006AAE A2[FF5A0100]        <1>       mov    [FAT_BuffDrvName], al
22603 00006AB3 C605[FE5A0100]01    <1>       mov    byte [FAT_BuffValidData], 1
22604 00006ABA E82B000000          <1>       call   fd_init_calculate_free_clusters
22605 00006ABF 7228                <1>       jc     short read_fd_FAT_sectors_retn
22606                              <1>
22607                              <1> loc_use_fd_boot_sector_params_FAT:
22608 00006AC1 C6460301            <1>       mov    byte [esi+LD_FATType], 1 ; FAT 12
22609 00006AC5 C6460401            <1>       mov    byte [esi+LD_FSType], 1
22610 00006AC9 8B462D              <1>         mov     eax, [esi+LD_BPB+VolumeID]
22611                              <1> loc_cont_use_fd_boot_sector_params:
22612 00006ACC 8A7E02              <1>       mov    bh, [esi+LD_PhyDrvNo]
22613 00006ACF 887E7D              <1>       mov    [esi+LD_DParamEntry], bh
22614 00006AD2 88FB                <1>       mov    bl, bh
22615 00006AD4 80C341              <1>       add    bl, 'A'
22616 00006AD7 881E                <1>       mov    byte [esi+LD_Name], bl
22617 00006AD9 C6460101            <1>       mov    byte [esi+LD_DiskType], 1
22618 00006ADD C6460500            <1>       mov    byte [esi+LD_LBAYes], 0
22619 00006AE1 C6467C00            <1>       mov    byte [esi+LD_PartitionEntry], 0
22620 00006AE5 C6467E06            <1>       mov    byte [esi+LD_MediaChanged], 6 ; Volume Name Reset
22621                              <1>
22622                              <1> read_fd_FAT_sectors_retn:
22623 00006AE9 C3                  <1>       retn
22624                              <1>
22625                              <1> fd_init_calculate_free_clusters:
22626                              <1>       ; 09/12/2017
22627                              <1>       ; 10/01/2016 (TRDOS 386 = TRDOS v2.0)
22628                              <1>       ; 04/07/2009
22629                              <1>       ; INPUT ->
22630                              <1>       ;     ESI = Logical DOS drive description table address
22631                              <1>       ; OUTPUT ->
22632                              <1>       ;    [ESI+LD_FreeSectors] will be set
22633                              <1>
22634 00006AEA 29C0                <1>       sub    eax, eax
22635 00006AEC 894674              <1>       mov    [esi+LD_FreeSectors], eax ; 0
22636 00006AEF B002                <1>       mov    al, 2 ; eax = 2
22637                              <1>
22638                              <1> fd_init_loop_get_next_cluster:
22639 00006AF1 E830000000          <1>       call   fd_init_get_next_cluster
22640 00006AF6 722D                <1>       jc     short fd_init_calculate_free_clusters_retn
22641                              <1>
22642                              <1> fd_init_free_fat_clusters:
22643                              <1>       ;cmp   eax, 0
22644                              <1>       ;ja    short fd_init_pass_inc_free_cluster_count
22645                              <1>       ;and   eax, eax
22646                              <1>       ;jnz   short fd_init_pass_inc_free_cluster_count
22647 00006AF8 6621C0              <1>       and    ax, ax
22648 00006AFB 7504                <1>       jnz    short fd_init_pass_inc_free_cluster_count
22649                              <1>       ;inc   dword [esi+LD_FreeSectors]
22650 00006AFD 66FF4674            <1>         inc  word [esi+LD_FreeSectors]
22651                              <1>
22652                              <1> fd_init_pass_inc_free_cluster_count:
22653                              <1>       ;mov   eax, [FAT_CurrentCluster]
22654 00006B01 66A1[FA5A0100]      <1>       mov    ax, [FAT_CurrentCluster]
22655                              <1>       ;cmp   eax, [esi+LD_Clusters]
```

```
22656 00006B07 663B4678            <1>       cmp    ax, [esi+LD_Clusters]
22657 00006B0B 7704                <1>       ja     short short retn_from_fd_init_calculate_free_clusters
22658                              <1>       ;inc   eax
22659 00006B0D 6640                <1>       inc    ax
22660 00006B0F EBE0                <1>       jmp    short fd_init_loop_get_next_cluster
22661                              <1>
22662                              <1> retn_from_fd_init_calculate_free_clusters:
22663 00006B11 8A4613              <1>       mov    al, [esi+LD_BPB+BPB_SecPerClust]
22664 00006B14 3C01                <1>       cmp    al, 1
22665 00006B16 760D                <1>       jna    short fd_init_calculate_free_clusters_retn
22666                              <1>       ;movzx eax, al
22667 00006B18 30E4                <1>       xor    ah, ah ; 09/12/2017
22668                              <1>       ;mov   ecx, [esi+LD_FreeSectors]
22669 00006B1A 668B4E74            <1>       mov    cx, [esi+LD_FreeSectors] ; Count of free clusters
22670                              <1>       ;mul   ecx
22671 00006B1E 66F7E1              <1>       mul    cx
22672                              <1>       ;mov   [esi+LD_FreeSectors], eax
22673 00006B21 66894674            <1>       mov    [esi+LD_FreeSectors], ax
22674                              <1> fd_init_calculate_free_clusters_retn:
22675 00006B25 C3                  <1>       retn
22676                              <1>
22677                              <1> fd_init_get_next_cluster:
22678                              <1>       ; 04/02/2016
22679                              <1>       ; 02/02/2016
22680                              <1>       ; 10/01/2016 (TRDOS 386 = TRDOS v2.0)
22681                              <1>       ; 04/07/2009
22682                              <1>       ; INPUT ->
22683                              <1>       ;    EAX = Current cluster
22684                              <1>       ;    ESI = Logical DOS drive description table address
22685                              <1>       ;    EDX = 0
22686                              <1>       ; OUTPUT ->
22687                              <1>       ;    EAX = Next cluster
22688                              <1>
22689 00006B26 A3[FA5A0100]        <1>       mov    [FAT_CurrentCluster], eax
22690                              <1> fd_init_get_next_cluster_readnext:
22691 00006B2B 29D2                <1>       sub    edx, edx ; 0
22692 00006B2D BB00040000          <1>       mov    ebx, 1024 ; 400h
22693 00006B32 F7F3                <1>       div    ebx
22694                              <1>       ; EAX = Count of 3 FAT sectors
22695                              <1>       ; EDX = Buffer entry index
22696 00006B34 89C1                <1>       mov    ecx, eax
22697                              <1>       ;mov   eax, 3
22698 00006B36 B003                <1>       mov    al, 3
22699 00006B38 F7E2                <1>       mul    edx ; Multiply by 3
22700 00006B3A 66D1E8              <1>       shr    ax, 1 ; Divide by 2
22701 00006B3D 89C3                <1>       mov    ebx, eax ; Buffer byte offset
22702 00006B3F 81C3001C0900        <1>       add    ebx, FAT_Buffer
22703 00006B45 89C8                <1>       mov    eax, ecx
22704                              <1>       ;mov   edx, 3
22705 00006B47 66BA0300            <1>       mov    dx, 3
22706 00006B4B F7E2                <1>       mul    edx
22707                              <1>       ; EAX = FAT Beginning Sector
22708                              <1>       ; EDX = 0
22709 00006B4D 8A0E                <1>       mov    cl, [esi+LD_Name]
22710                              <1>       ;cmp   byte [FAT_BuffValidData], 0
22711                              <1>       ;jna   short fd_init_load_FAT_sectors0
22712 00006B4F 3A0D[FF5A0100]      <1>       cmp    cl, [FAT_BuffDrvName]
22713 00006B55 751E                <1>       jne    short fd_init_load_FAT_sectors0
22714 00006B57 3B05[025B0100]      <1>       cmp    eax, [FAT_BuffSector]
22715 00006B5D 751C                <1>       jne    short fd_init_load_FAT_sectors1
22716                              <1>       ;mov   eax, [FAT_CurrentCluster]
22717 00006B5F A0[FA5A0100]        <1>       mov    al, [FAT_CurrentCluster]
22718                              <1>       ;shr   eax, 1
22719 00006B64 D0E8                <1>       shr    al, 1
22720 00006B66 668B03              <1>       mov    ax, [ebx]
22721 00006B69 7306                <1>       jnc    short fd_init_gnc_even
22722 00006B6B 66C1E804            <1>       shr    ax, 4
22723                              <1> fd_init_gnc_clc_retn:
22724 00006B6F F8                  <1>       clc
22725 00006B70 C3                  <1>       retn
22726                              <1>
22727                              <1> fd_init_gnc_even:
22728 00006B71 80E40F              <1>       and    ah, 0Fh
22729 00006B74 C3                  <1>       retn
22730                              <1>
22731                              <1> fd_init_load_FAT_sectors0:
22732 00006B75 880D[FF5A0100]      <1>       mov    [FAT_BuffDrvName], cl
22733                              <1> fd_init_load_FAT_sectors1:
22734 00006B7B C605[FE5A0100]00    <1>       mov    byte [FAT_BuffValidData], 0
22735 00006B82 A3[025B0100]        <1>       mov    [FAT_BuffSector], eax
22736 00006B87 034660              <1>       add    eax, [esi+LD_FATBegin]
22737 00006B8A BB001C0900          <1>       mov    ebx, FAT_Buffer
22738                              <1>       ;movzx ecx, word [esi+LD_BPB+BPB_FATSz16]
22739 00006B8F 668B4E1C            <1>       mov    cx, [esi+LD_BPB+BPB_FATSz16]
22740 00006B93 662B0D[025B0100]    <1>       sub    cx, [FAT_BuffSector]
22741                              <1>         ;cmp ecx, 3
22742 00006B9A 6683F903            <1>       cmp    cx, 3
22743 00006B9E 7605                <1>       jna    short fdinit_pass_fix_sector_count_3
22744                              <1>       ;mov   ecx, 3
22745 00006BA0 B903000000          <1>       mov    ecx, 3
22746                              <1> fdinit_pass_fix_sector_count_3:
22747 00006BA5 E834860000          <1>       call   chs_read
22748 00006BAA 730D                <1>       jnc    short fd_init_FAT_sectors_no_load_error
22749 00006BAC C605[FE5A0100]00    <1>       mov    byte [FAT_BuffValidData], 0
22750                              <1>              ; Drv not ready or read Error !
22751 00006BB3 B80F000000          <1>       mov    eax, ERR_DRV_NOT_RDY ; 15
22752                              <1>       ;xor   edx, edx
22753 00006BB8 C3                  <1>       retn
22754                              <1>
22755                              <1> fd_init_FAT_sectors_no_load_error:
22756 00006BB9 C605[FE5A0100]01    <1>       mov    byte [FAT_BuffValidData], 1
22757 00006BC0 A1[FA5A0100]        <1>       mov    eax, [FAT_CurrentCluster]
22758 00006BC5 E961FFFFFF          <1>         jmp     fd_init_get_next_cluster_readnext
```

```
22759                                  <1>
22760                                  <1> get_FAT_volume_name:
22761                                  <1>       ; 10/01/2016 (TRDOS 386 = TRDOS v2.0)
22762                                  <1>       ; 12/09/2009
22763                                  <1>       ; INPUT ->
22764                                  <1>       ;      BH = Logical DOS drive number (0,1,2,3,4 ...)
22765                                  <1>       ;       BL = 0
22766                                  <1>       ; OUTPUT ->
22767                                  <1>       ;      CF = 0 -> ESI = Volume name address
22768                                  <1>       ;      CF = 1 -> Root volume name not found
22769                                  <1>
22770                                  <1>       ;mov   ah, 0FFh
22771                                  <1>       ;mov   al, [Last_Dos_DiskNo]
22772                                  <1>       ;cmp   al, bh
22773                                  <1>       ;jb     short loc_gfvn_dir_load_err
22774                                  <1>
22775 00006BCA 89DE                   <1>       mov    esi, ebx
22776 00006BCC 81E600FF0000           <1>       and    esi, 0FF00h ; esi = bh
22777 00006BD2 81C600010900           <1>       add    esi, Logical_DOSDisks
22778 00006BD8 8A06                   <1>       mov     al, [esi+LD_Name]
22779 00006BDA 8A6603                 <1>       mov     ah, [esi+LD_FATType]
22780 00006BDD 80FC01                 <1>       cmp     ah, 1
22781 00006BE0 7210                   <1>       jb     short loc_gfvn_dir_load_err
22782 00006BE2 3C41                   <1>       cmp    al, 'A'
22783 00006BE4 720C                   <1>       jb      short loc_gfvn_dir_load_err
22784 00006BE6 80FC02                 <1>       cmp    ah, 2
22785 00006BE9 7708                   <1>       ja      short get_FAT32_root_cluster
22786                                  <1>
22787 00006BEB E8B24E0000             <1>       call    load_FAT_root_directory
22788 00006BF0 730B                   <1>       jnc     short loc_get_volume_name
22789                                  <1>
22790                                  <1> loc_gfvn_dir_load_err:
22791 00006BF2 C3                     <1>       retn
22792                                  <1>
22793                                  <1> get_FAT32_root_cluster:
22794 00006BF3 8B4632                 <1>       mov    eax, [esi+LD_BPB+BPB_RootClus]
22795 00006BF6 E8324F0000             <1>       call    load_FAT_sub_directory
22796 00006BFB 7224                   <1>       jc     short loc_get_volume_name_retn
22797                                  <1>
22798                                  <1> loc_get_volume_name:
22799 00006BFD BE00000800             <1>         mov     esi, Directory_Buffer
22800 00006C02 6631C9                 <1>       xor    cx, cx ; 0
22801                                  <1> check_root_volume_name:
22802 00006C05 8A06                   <1>       mov    al, [esi]
22803 00006C07 08C0                   <1>       or      al, al
22804 00006C09 7416                   <1>       jz      short loc_get_volume_name_retn
22805 00006C0B 807E0B08               <1>       cmp     byte [esi+0Bh], 08h
22806 00006C0F 7410                   <1>       je      short loc_get_volume_name_retn
22807 00006C11 663B0D[135B0100]       <1>       cmp     cx, [DirBuff_LastEntry]
22808 00006C18 7308                   <1>       jnb     short pass_check_root_volume_name
22809 00006C1A 6641                   <1>       inc     cx
22810 00006C1C 83C620                 <1>       add     esi, 32
22811 00006C1F EBE4                   <1>       jmp     short check_root_volume_name
22812                                  <1>
22813                                  <1> loc_get_volume_name_retn:
22814 00006C21 C3                     <1>       retn
22815                                  <1>
22816                                  <1> pass_check_root_volume_name:
22817 00006C22 803D[0F5B0100]03       <1>       cmp    byte [DirBuff_FATType], 3
22818 00006C29 7230                   <1>       jb     short loc_get_volume_name_retn_xor
22819                                  <1>
22820 00006C2B BB001C0900             <1>       mov    ebx, FAT_Buffer
22821 00006C30 BE00010900             <1>       mov    esi, Logical_DOSDisks
22822 00006C35 31C0                   <1>       xor    eax, eax
22823 00006C37 8A25[0E5B0100]         <1>       mov    ah, [DirBuff_DRV]
22824 00006C3D 80EC41                 <1>       sub    ah, 'A'
22825 00006C40 01C6                   <1>       add    esi, eax
22826 00006C42 A1[155B0100]           <1>       mov    eax, [DirBuff_Cluster]
22827 00006C47 E8FB4C0000             <1>       call   get_next_cluster
22828 00006C4C 7305                   <1>       jnc    short loc_gfvn_load_FAT32_dir_cluster
22829                                  <1>
22830 00006C4E 83F801                 <1>       cmp     eax, 1
22831 00006C51 F5                     <1>       cmc
22832 00006C52 C3                     <1>       retn
22833                                  <1>
22834                                  <1> loc_gfvn_load_FAT32_dir_cluster:
22835 00006C53 E8D54E0000             <1>       call   load_FAT_sub_directory
22836 00006C58 73A3                   <1>       jnc    short loc_get_volume_name
22837 00006C5A C3                     <1>       retn
22838                                  <1>
22839                                  <1> loc_get_volume_name_retn_xor:
22840 00006C5B 31C0                   <1>       xor    eax, eax
22841 00006C5D C3                     <1>       retn
22842                                  <1>
22843                                  <1> get_media_change_status:
22844                                  <1>       ; 10/01/2016 (TRDOS 386 = TRDOS v2.0)
22845                                  <1>       ; 09/09/2009
22846                                  <1>       ; INPUT:
22847                                  <1>       ;     DL = Drive number (physical)
22848                                  <1>       ; OUTPUT: clc & AH = 6 media changed
22849                                  <1>       ;     clc & AH = 0 media not changed
22850                                  <1>       ;     stc -> Drive not ready or an error
22851                                  <1>
22852 00006C5E B416                   <1>       mov    ah, 16h
22853 00006C60 E8A1D5FFFF             <1>       call   int13h
22854 00006C65 80FC06                 <1>       cmp    ah, 06h
22855 00006C68 7405                   <1>       je     short loc_gmc_status_retn
22856 00006C6A 08E4                   <1>       or     ah, ah
22857 00006C6C 7401                   <1>       jz     short loc_gmc_status_retn
22858                                  <1> loc_gmc_status_stc_retn:
22859 00006C6E F9                     <1>       stc
22860                                  <1> loc_gmc_status_retn:
22861 00006C6F C3                     <1>       retn
```

```
                            %include 'trdosk3.s' ; 06/01/2016
                            ; ****************************************************************************
                            ; TRDOS386.ASM (TRDOS 386 Kernel - v2.0.0) - MAIN PROGRAM : trdosk3.s
                            ; ----------------------------------------------------------------------------
                            ; Last Update: 22/11/2017
                            ; ----------------------------------------------------------------------------
                            ; Beginning: 06/01/2016
                            ; ----------------------------------------------------------------------------
                            ; Assembler: NASM version 2.11 (trdos386.s)
                            ; ----------------------------------------------------------------------------
                            ; Derived from TRDOS Operating System v1.0 (8086) source code by Erdogan Tan
                            ; MAINPROG.ASM (09/11/2011)
                            ; ****************************************************************************
                            ; MAINPROG.ASM [ TRDOS KERNEL - COMMAND EXECUTER SECTION - MAIN PROGRAM ]
                            ; (c) 2004-2011  Erdogan TAN  [ 17/01/2004 ]  Last Update: 09/11/2011
                            ; CMD_INTR.ASM [ TRDOS Command Interpreter Procedure ] Last Update: 09/11/2011
                            ; DIR.ASM [ DIRECTORY FUNCTIONS ] Last Update: 09/10/2011
                            ; FILE.ASM [ FILE FUNCTIONS ] Last Update: 09/10/2011
                            
                            change_current_drive:
                                  ; 16/10/2016
                                  ; 02/02/2016
                                  ; 15/01/2016 (TRDOS 386 = TRDOS v2.0)
                                  ; 18/08/2011
                                  ; 09/09/2009
                                  ; INPUT:
                                  ;   DL = Logical DOS Drive Number
                                  ; OUTPUT:
                                  ;  cf=1 -> Not successful
                                  ;   EAX = Error code
                                  ;  cf=0 ->
                                  ;   EAX = 0 (successful)
                            
00006C70 31DB                       xor    ebx, ebx
00006C72 88D7                       mov    bh, dl
                            
                                    ;cmp   dl, 1
                                    ;jna   short loc_ccdrv_initial_media_change_check
                                    ;cmp   bh, [Last_Dos_DiskNo]
                                    ;ja    short loc_ccdrv_drive_not_ready_err
                            
                            loc_ccdrv_initial_media_change_check:
00006C74 BE00010900                 mov    esi, Logical_DOSDisks
00006C79 01DE                       add    esi, ebx
                            loc_ccdrv_dos_drive_name_check:
00006C7B 80FA02                     cmp    dl, 2
00006C7E 720F                       jb     short loc_ccdrv_dos_drive_name_check_ok
                            
00006C80 8A06                       mov    al, [esi+LD_Name]
00006C82 2C41                       sub    al, 'A'
00006C84 38D0                       cmp    al, dl
00006C86 7407                       je     short loc_ccdrv_dos_drive_name_check_ok
                            
                            loc_ccdrv_drive_not_ready_err:
                                    ; 16/10/2016 (15h -> 15)
00006C88 B80F000000                 mov    eax, 15 ; Drive not ready
                            loc_change_current_drive_stc_retn:
00006C8D F9                         stc
00006C8E C3                         retn
                            
                            loc_ccdrv_dos_drive_name_check_ok:
00006C8F 8A667E                     mov    ah, [esi+LD_MediaChanged]
00006C92 80FC06                     cmp    ah, 6  ; VOLUME NAME CHECK/MOVE SIGN
00006C95 7455                       je     short loc_ccdrv_get_FAT_volume_name_0
                            
00006C97 80FA01                     cmp    dl, 1
00006C9A 777D                       ja     short loc_gmcs_init_drv_hd
                            
                            loc_gmcs_init_drv_fd:
00006C9C 08E4                       or     ah, ah
                                    ; AH = 1 is initialization sign (invalid_fd_parameter)
00006C9E 7517                       jnz    short loc_ccdrv_call_fd_init
                            
00006CA0 E8B9FFFFFF                 call   get_media_change_status
00006CA5 72E1                       jc     short loc_ccdrv_drive_not_ready_err
                            
00006CA7 20E4                       and    ah, ah
00006CA9 7476                       jz     short loc_change_current_drv3
                            
00006CAB 80F406                     xor    ah, 6
00006CAE 75D8                       jnz    short loc_ccdrv_drive_not_ready_err
                            
                            loc_ccdrv_call_fd_init_check_vol_id:
00006CB0 E8490A0000                 call   get_volume_serial_number
00006CB5 730D                       jnc    short loc_ccdrv_check_vol_serial
                            
                            loc_ccdrv_call_fd_init:
00006CB7 E872FCFFFF                 call   floppy_drv_init
00006CBC 731A                       jnc    short loc_reset_drv_fd_current_dir
                            
                            loc_ccdrv_fdinit_fail_retn:
                                    ; 16/10/2016
00006CBE B80F000000                 mov    eax, 15 ; Drive not ready
00006CC3 C3                         retn
                            
                            loc_ccdrv_check_vol_serial:
00006CC4 A3[DC520100]               mov    [Current_VolSerial], eax
                                    ;mov   dl, bh
00006CC9 E860FCFFFF                 call   floppy_drv_init
00006CCE 72EE                       jc     short loc_ccdrv_fdinit_fail_retn
                            
00006CD0 3B05[DC520100]             cmp    eax, [Current_VolSerial]
00006CD6 7445                       je     short loc_change_current_drv2
```

```
22965                                  <1>
22966                                  <1> loc_reset_drv_fd_current_dir:
22967 00006CD8 31C0                     <1>       xor    eax, eax
22968 00006CDA 88467F                   <1>         mov  [esi+LD_CDirLevel], al
22969 00006CDD 89F7                     <1>       mov    edi, esi
22970 00006CDF 81C780000000             <1>       add    edi, LD_CurrentDirectory
22971 00006CE5 B920000000               <1>       mov    ecx, 32
22972 00006CEA F3AB                     <1>       rep    stosd
22973                                  <1>
22974                                  <1> loc_ccdrv_get_FAT_volume_name_0:
22975 00006CEC 8A4603                   <1>       mov    al, [esi+LD_FATType]
22976 00006CEF 08C0                     <1>       or     al, al
22977 00006CF1 742A                     <1>       jz     short loc_change_current_drv2
22978                                  <1>
22979 00006CF3 56                       <1>       push   esi
22980 00006CF4 3C02                     <1>       cmp    al, 2
22981 00006CF6 7705                     <1>       ja     short loc_ccdrv_get_FAT32_vol_name
22982                                  <1>
22983                                  <1> loc_ccdrv_get_FAT2_16_vol_name:
22984 00006CF8 83C631                   <1>       add    esi, LD_BPB + VolumeLabel
22985 00006CFB EB03                     <1>       jmp    short loc_ccdrv_get_FAT_volume_name_1
22986                                  <1>
22987                                  <1> loc_ccdrv_get_FAT32_vol_name:
22988 00006CFD 83C64D                   <1>       add    esi, LD_BPB + FAT32_VolLab
22989                                  <1> loc_ccdrv_get_FAT_volume_name_1:
22990 00006D00 53                       <1>       push   ebx
22991 00006D01 56                       <1>       push   esi
22992 00006D02 E8C3FEFFFF               <1>       call   get_FAT_volume_name
22993 00006D07 5F                       <1>       pop    edi
22994 00006D08 5B                       <1>       pop    ebx
22995                                  <1>       ; BL = 0
22996 00006D09 720B                     <1>       jc     short loc_change_current_drv1
22997 00006D0B 20C0                     <1>       and    al, al
22998 00006D0D 7407                     <1>       jz     short loc_change_current_drv1
22999                                  <1>
23000                                  <1> loc_ccdrv_move_FAT_volume_name:
23001 00006D0F B90B000000               <1>       mov    ecx, 11
23002 00006D14 F3A4                     <1>       rep    movsb
23003                                  <1>
23004                                  <1> loc_change_current_drv1:
23005 00006D16 5E                       <1>       pop    esi
23006 00006D17 EB04                     <1>       jmp    short loc_change_current_drv2
23007                                  <1>
23008                                  <1> loc_gmcs_init_drv_hd:
23009 00006D19 08E4                     <1>       or     ah, ah
23010 00006D1B 7404                     <1>       jz     short loc_change_current_drv3
23011                                  <1>       ; BL = 0, BH = Logical DOS drive number
23012                                  <1> loc_change_current_drv2:
23013 00006D1D C6467E00                 <1>       mov    byte [esi+LD_MediaChanged], 0
23014                                  <1> loc_change_current_drv3:
23015 00006D21 883D[E6520100]           <1>       mov    [Current_Drv], bh
23016                                  <1>
23017                                  <1>       ;call  restore_current_directory
23018                                  <1>       ;retn
23019                                  <1>
23020                                  <1> restore_current_directory:
23021                                  <1>       ; 11/02/2016
23022                                  <1>       ; 15/01/2016 (TRDOS 386 = TRDOS v2.0)
23023                                  <1>       ; 25/01/2010
23024                                  <1>       ; 12/10/2009
23025                                  <1>       ;
23026                                  <1>       ; INPUT:
23027                                  <1>       ;   ESI = Logical DOS Drive Description Table
23028                                  <1>       ;
23029                                  <1>       ; OUTPUT:
23030                                  <1>       ;   ESI = Logical DOS Drive Description Table
23031                                  <1>       ;   EDI = offset Current_Dir_Drv
23032                                  <1>
23033 00006D27 8A4603                   <1>       mov    al, [esi+LD_FATType]
23034 00006D2A A2[E5520100]             <1>       mov    [Current_FATType], al
23035                                  <1>
23036 00006D2F 8A26                     <1>       mov    ah, [esi+LD_Name]
23037 00006D31 8825[E7520100]           <1>       mov    [Current_Dir_Drv], ah
23038                                  <1>
23039 00006D37 20C0                     <1>       and    al, al
23040 00006D39 741D                     <1>       jz     short loc_restore_FS_current_directory
23041                                  <1>
23042                                  <1> loc_restore_FAT_current_directory:
23043 00006D3B 8A667F                   <1>       mov    ah, [esi+LD_CDirLevel]
23044 00006D3E 8825[E4520100]           <1>       mov    [Current_Dir_Level], ah
23045 00006D44 08E4                     <1>       or     ah, ah
23046 00006D46 7416                     <1>         jz   short loc_ccdrv_reset_cdir_FAT_12_16_32_fcluster
23047                                  <1>
23048 00006D48 0FB6D4                   <1>       movzx  edx, ah
23049 00006D4B C0E204                   <1>       shl    dl, 4 ; * 16
23050 00006D4E 01F2                     <1>         add  edx, esi
23051 00006D50 8B828C000000             <1>       mov    eax, [edx+LD_CurrentDirectory+12]
23052 00006D56 EB2C                     <1>       jmp    short loc_ccdrv_reset_cdir_FAT_fcluster
23053                                  <1>
23054                                  <1> loc_restore_FS_current_directory:
23055 00006D58 E80B4E0000               <1>       call   load_current_FS_directory
23056 00006D5D C3                       <1>       retn
23057                                  <1>
23058                                  <1> loc_ccdrv_reset_cdir_FAT_12_16_32_fcluster:
23059 00006D5E 3C03                     <1>       cmp    al, 3
23060 00006D60 7205                     <1>       jb     short loc_ccdrv_reset_cdir_FAT_12_16_fcluster
23061                                  <1> loc_ccdrv_reset_cdir_FAT32_fcluster:
23062 00006D62 8B4632                   <1>       mov    eax, [esi+LD_BPB+FAT32_RootFClust]
23063 00006D65 EB04                     <1>       jmp    short loc_ccdrv_check_rootdir_sign
23064                                  <1> loc_ccdrv_reset_cdir_FAT_12_16_fcluster:
23065 00006D67 30C0                     <1>       xor    al, al  ; xor eax, eax
23066 00006D69 31D2                     <1>       xor    edx, edx
23067                                  <1> loc_ccdrv_check_rootdir_sign:
```

```
23068 00006D6B 80BE8000000000      <1>       cmp    byte [esi+LD_CurrentDirectory], 0
23069 00006D72 7510                <1>       jne    short loc_ccdrv_reset_cdir_FAT_fcluster
23070                              <1> loc_ccdrv_set_rootdir_FAT_fcluster:
23071 00006D74 89868C000000        <1>         mov     [esi+LD_CurrentDirectory+12], eax
23072 00006D7A C78680000000524F4F- <1>       mov    dword [esi+LD_CurrentDirectory], 'ROOT'
23073 00006D83 54                  <1>
23074                              <1>
23075                              <1> loc_ccdrv_reset_cdir_FAT_fcluster:
23076 00006D84 A3[E0520100]        <1>       mov    [Current_Dir_FCluster], eax
23077                              <1>
23078 00006D89 BF[475B0100]        <1>       mov    edi, PATH_Array
23079 00006D8E 89F2                <1>       mov    edx, esi
23080 00006D90 81C680000000        <1>       add    esi, LD_CurrentDirectory
23081 00006D96 B920000000          <1>       mov    ecx, 32
23082 00006D9B F3A5                <1>       rep    movsd
23083                              <1>
23084 00006D9D E8992D0000          <1>       call   change_prompt_dir_string
23085                              <1>
23086 00006DA2 89D6                <1>       mov    esi, edx
23087                              <1>
23088 00006DA4 29C0                <1>         sub  eax, eax
23089                              <1>        ;sub  edx, edx
23090 00006DA6 BF[E7520100]        <1>       mov    edi, Current_Dir_Drv
23091                              <1>
23092 00006DAB A2[D3060100]        <1>       mov    [Restore_CDIR], al ; 0
23093 00006DB0 C3                  <1>       retn
23094                              <1>
23095                              <1> dos_prompt:
23096                              <1>       ; 06/05/2016
23097                              <1>       ; 30/01/2016
23098                              <1>       ; 29/01/2016
23099                              <1>       ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
23100                              <1>       ; 15/09/2011
23101                              <1>       ; 13/09/2009
23102                              <1>       ; 2004-2005
23103                              <1>
23104                              <1>       ; 06/05/2016
23105 00006DB1 C705[A45F0100]-     <1>       mov    dword [mainprog_return_addr], return_from_cmd_interpreter
23106 00006DB7 [656E0000]          <1>
23107                              <1>
23108                              <1> loc_TRDOS_prompt:
23109 00006DBB BF[E6530100]        <1>       mov    edi, TextBuffer
23110 00006DC0 C6075B              <1>       mov    byte [edi], "["
23111 00006DC3 47                  <1>       inc    edi
23112 00006DC4 BE[26070100]        <1>       mov    esi, TRDOSPromptLabel
23113                              <1> get_next_prompt_label_char:
23114 00006DC9 803E20              <1>       cmp    byte [esi], 20h
23115 00006DCC 7203                <1>       jb     short pass_prompt_label
23116 00006DCE A4                  <1>       movsb
23117 00006DCF EBF8                <1>       jmp    short get_next_prompt_label_char
23118                              <1> pass_prompt_label:
23119 00006DD1 C6075D              <1>       mov    byte [edi], "]"
23120 00006DD4 47                  <1>       inc    edi
23121 00006DD5 C60720              <1>       mov    byte [edi], 20h
23122 00006DD8 47                  <1>       inc    edi
23123 00006DD9 BE[E7520100]        <1>       mov    esi, Current_Dir_Drv
23124 00006DDE 66A5                <1>       movsw
23125 00006DE0 A4                  <1>       movsb
23126                              <1> loc_prompt_current_directory:
23127 00006DE1 803E20              <1>       cmp    byte [esi], 20h
23128 00006DE4 7203                <1>       jb     short pass_prompt_current_directory
23129 00006DE6 A4                  <1>       movsb
23130 00006DE7 EBF8                <1>       jmp    short loc_prompt_current_directory
23131                              <1> pass_prompt_current_directory:
23132 00006DE9 C6073E              <1>       mov    byte [edi], '>'
23133 00006DEC 47                  <1>       inc    edi
23134 00006DED C60700              <1>       mov    byte [edi], 0
23135 00006DF0 BE[E6530100]        <1>       mov    esi, TextBuffer
23136 00006DF5 E863F5FFFF          <1>       call   print_msg
23137                              <1>
23138                              <1>       ;sub   bh, bh ; video page = 0
23139                              <1>       ;call  get_cpos ; get cursor position
23140 00006DFA 668B15[3E520100]    <1>       mov    dx, [CURSOR_POSN] ; video page 0
23141 00006E01 8815[46530100]      <1>       mov    [CursorColumn], dl
23142                              <1>
23143                              <1>       ; 30/01/2016 (to show cursor on the row, again)
23144                              <1>       ; (Initial color attributes of video page 0 is 0)
23145                              <1>       ; (see: 'StartPMP' in trdos386.s)
23146                              <1>       ;
23147                              <1>       ;mov   edi, 0B8000h ; start of video page 0
23148                              <1>       ;movzx ecx, dl ; column
23149                              <1>       ;mov   al, 80
23150                              <1>       ;mul   dh
23151                              <1>       ;add   ax, cx
23152                              <1>       ;shl   ax, 1 ; character + attribute
23153                              <1>       ;add   di, ax ; (2*80*row) + (2*column)
23154                              <1>       ;neg   cl
23155                              <1>       ;add   cl, 80
23156                              <1>       ;mov   ax, 700h ;  ah = 7 (color attribute)
23157                              <1>       ;rep   stosw
23158                              <1>
23159                              <1> loc_rw_char:
23160 00006E07 E899000000          <1>       call   rw_char
23161                              <1> loc_move_command:
23162 00006E0C BE[96530100]        <1>       mov    esi, CommandBuffer
23163 00006E11 89F7                <1>       mov    edi, esi
23164 00006E13 31C9                <1>       xor    ecx, ecx
23165                              <1> first_command_char:
23166 00006E15 AC                  <1>       lodsb
23167 00006E16 3C20                <1>       cmp    al, 20h
23168 00006E18 772E                <1>       ja     short pass_space_control
23169 00006E1A 7241                <1>       jb     short loc_move_cmd_arguments_ok
23170 00006E1C 81FE[E5530100]      <1>       cmp    esi, CommandBuffer + 79
```

```
23171 00006E22 72F1               <1>       jb     short first_command_char
23172 00006E24 EB37               <1>       jmp    short loc_move_cmd_arguments_ok
23173                             <1>
23174                             <1> next_command_char:
23175 00006E26 AC                 <1>       lodsb
23176 00006E27 3C20               <1>       cmp    al, 20h
23177 00006E29 771D               <1>       ja     short pass_space_control
23178 00006E2B 7230               <1>       jb     short loc_move_cmd_arguments_ok
23179                             <1>
23180                             <1> loc_1st_cmd_arg: ; 30/01/2016
23181 00006E2D AC                 <1>       lodsb
23182 00006E2E 3C20               <1>       cmp    al, 20h
23183 00006E30 74FB               <1>       je     short loc_1st_cmd_arg
23184 00006E32 7229               <1>       jb     short loc_move_cmd_arguments_ok
23185                             <1>
23186 00006E34 C60700             <1>         mov     byte [edi], 0
23187 00006E37 47                 <1>       inc    edi
23188                             <1>
23189                             <1> loc_move_cmd_arguments:
23190 00006E38 AA                 <1>       stosb
23191 00006E39 81FE[E5530100]     <1>       cmp    esi, CommandBuffer + 79
23192 00006E3F 731C               <1>       jnb    short loc_move_cmd_arguments_ok
23193 00006E41 AC                 <1>         lodsb
23194 00006E42 3C20               <1>       cmp    al, 20h
23195 00006E44 73F2               <1>       jnb    short loc_move_cmd_arguments
23196 00006E46 EB15               <1>       jmp    short loc_move_cmd_arguments_ok
23197                             <1>
23198                             <1> pass_space_control:
23199 00006E48 3C61               <1>       cmp    al, 61h
23200 00006E4A 7206               <1>       jb     short pass_capitalize
23201 00006E4C 3C7A               <1>       cmp    al, 7Ah
23202 00006E4E 7702               <1>       ja     short pass_capitalize
23203 00006E50 24DF               <1>       and    al, 0DFh
23204                             <1> pass_capitalize:
23205 00006E52 AA                 <1>       stosb
23206 00006E53 FEC1               <1>       inc     cl
23207 00006E55 81FE[E5530100]     <1>         cmp     esi, CommandBuffer + 79
23208 00006E5B 72C9               <1>       jb      short next_command_char
23209                             <1>
23210                             <1> loc_move_cmd_arguments_ok:
23211 00006E5D C60700             <1>         mov     byte [edi], 0
23212                             <1>
23213                             <1> call_command_interpreter:
23214 00006E60 E8D4080000         <1>       call    command_interpreter
23215                             <1>
23216                             <1> return_from_cmd_interpreter:
23217 00006E65 B950000000         <1>         mov  ecx, 80
23218                             <1>       ;mov   cx, 80
23219 00006E6A BF[96530100]       <1>       mov    edi, CommandBuffer
23220 00006E6F 30C0               <1>       xor    al, al
23221 00006E71 F3AA               <1>       rep    stosb
23222                             <1>       ;cmp   byte [Program_Exit], 0
23223                             <1>       ;ja    short loc_terminate_trdos
23224                             <1>
23225                             <1>       ; 16/01/2016
23226 00006E73 803D[C25E0000]03   <1>       cmp    byte [CRT_MODE], 3 ; 80*25 color
23227 00006E7A 741D               <1>       je     short pass_set_txt_mode
23228                             <1>
23229 00006E7C E8E2A6FFFF         <1>       call   set_txt_mode ; set vide mode to 03h
23230                             <1>       ; 07/01/2017
23231 00006E81 30C0               <1>       xor    al, al
23232                             <1>
23233                             <1> loc_check_active_page:
23234                             <1>       ;xor   al, al
23235 00006E83 3805[4E520100]     <1>       cmp    [ACTIVE_PAGE], al ; 0
23236 00006E89 0F842CFFFFFF       <1>         je      loc_TRDOS_prompt
23237                             <1>       ; AL = 0 = video page 0
23238 00006E8F E8E8AAFFFF         <1>       call   set_active_page
23239 00006E94 E922FFFFFF         <1>         jmp     loc_TRDOS_prompt ; infinitive loop
23240                             <1>
23241                             <1> pass_set_txt_mode:
23242 00006E99 BE[6F130100]       <1>       mov    esi, nextline
23243 00006E9E E8BAF4FFFF         <1>       call   print_msg
23244 00006EA3 EBDE               <1>       jmp     short loc_check_active_page
23245                             <1>
23246                             <1> rw_char:
23247                             <1>       ; 13/05/2016
23248                             <1>       ; 30/01/2016
23249                             <1>       ; 29/01/2016
23250                             <1>       ; 17/01/2016 (TRDOS 386 = TRDOS v2.0)
23251                             <1>       ; 2004-2005
23252                             <1>
23253                             <1>       ; DH = cursor row, DL = cursor column
23254                             <1>       ; BH = 0 = video page number (active page)
23255                             <1>
23256                             <1>       ;xor   bh, bh ; 0 = video page 0
23257                             <1>
23258                             <1> readnextchar:
23259 00006EA5 30E4               <1>       xor     ah, ah
23260 00006EA7 E86A9DFFFF         <1>       call   int16h
23261 00006EAC 20C0               <1>       and    al, al
23262 00006EAE 7434               <1>       jz     short loc_arrow
23263 00006EB0 3CE0               <1>       cmp    al, 0E0h
23264 00006EB2 7430               <1>       je     short loc_arrow
23265 00006EB4 3C08               <1>       cmp    al, 08h
23266 00006EB6 7544               <1>       jne    short char_return
23267                             <1> loc_back:
23268 00006EB8 3A15[46530100]     <1>       cmp    dl, [CursorColumn]
23269 00006EBE 76E5               <1>       jna     short readnextchar
23270                             <1> prev_column:
23271 00006EC0 FECA               <1>       dec    dl
23272                             <1> set_cursor_pos:
23273 00006EC2 6652               <1>       push   dx
```

```
23274                                   <1>       ;xor   bh, bh ; 0 = video page 0
23275                                   <1>       ; DH = Row, DL = Column
23276 00006EC4 E87FAEFFFF              <1>       call  _set_cpos ; 17/01/2016
23277 00006EC9 665A                    <1>         pop  dx
23278                                   <1>       ;movzx ebx, dl
23279 00006ECB 88D3                    <1>       mov   bl, dl
23280 00006ECD 2A1D[46530100]          <1>       sub   bl, [CursorColumn]
23281 00006ED3 B020                    <1>       mov   al, 20h
23282 00006ED5 8883[96530100]          <1>       mov   [CommandBuffer+ebx], al
23283                                   <1>       ;sub  bh, bh ; video page 0
23284                                   <1>       ;mov  cx, 1
23285 00006EDB B307                    <1>       mov   bl, 7 ; color attribute
23286 00006EDD E857ADFFFF              <1>       call  _write_c_current ; 17/01/2016
23287                                   <1>       ;mov  dx, [CURSOR_POSN]
23288 00006EE2 EBC1                    <1>       jmp   short readnextchar
23289                                   <1> loc_arrow:
23290 00006EE4 80FC4B                  <1>       cmp   ah, 4Bh
23291 00006EE7 74CF                    <1>       je    short loc_back
23292 00006EE9 80FC53                  <1>       cmp   ah, 53h
23293 00006EEC 74CA                    <1>       je     short loc_back
23294 00006EEE 80FC4D                  <1>       cmp   ah, 4Dh
23295 00006EF1 75B2                    <1>       jne   short readnextchar
23296 00006EF3 80FA4F                  <1>       cmp   dl, 79
23297 00006EF6 73AD                    <1>       jnb   short readnextchar
23298 00006EF8 FEC2                    <1>       inc   dl
23299 00006EFA EBC6                    <1>       jmp   short set_cursor_pos
23300                                   <1> char_return:
23301 00006EFC 0FB6DA                  <1>       movzx  ebx, dl
23302 00006EFF 2A1D[46530100]          <1>       sub   bl, [CursorColumn]
23303 00006F05 3C20                    <1>       cmp   al, 20h
23304 00006F07 7220                    <1>       jb    short loc_escape
23305 00006F09 8883[96530100]          <1>       mov   [CommandBuffer+ebx], al
23306 00006F0F 80FA4F                  <1>       cmp   dl, 79
23307 00006F12 7391                    <1>       jnb   short readnextchar
23308 00006F14 66BB0700                <1>       mov   bx, 7 ; color attribute
23309 00006F18 E895ADFFFF              <1>       call  _write_tty
23310 00006F1D 668B15[3E520100]        <1>       mov   dx, [CURSOR_POSN] ; video page 0
23311 00006F24 E97CFFFFFF              <1>         jmp    readnextchar
23312                                   <1> loc_escape:
23313 00006F29 3C1B                    <1>       cmp   al, 1Bh
23314 00006F2B 7418                    <1>       je    short rw_char_retn
23315                                   <1>       ;
23316 00006F2D 3C0D                    <1>       cmp   al, 0Dh ; CR
23317 00006F2F 0F8570FFFFFF            <1>         jne    readnextchar
23318                                   <1>       ; 13/05/2016
23319 00006F35 66BB0700                <1>       mov   bx, 7 ; attribute/color (bl)
23320                                   <1>                   ; video page 0 (bh=0)
23321 00006F39 E874ADFFFF              <1>       call  _write_tty
23322                                   <1>       ;mov  bx, 7  ; attribute/color
23323                                   <1>                   ; video page 0 (bh=0)
23324 00006F3E B00A                    <1>       mov   al, 0Ah ; LF
23325 00006F40 E86DADFFFF              <1>       call  _write_tty
23326                                   <1> rw_char_retn:
23327 00006F45 C3                      <1>       retn
23328                                   <1>
23329                                   <1> show_date:
23330                                   <1>       ; 18/01/2016 (TRDOS 386 = TRDOS v2.0)
23331                                   <1>         ; 2004-2005
23332                                   <1>
23333                                   <1>       ;mov  ah, 04h
23334                                   <1>       ;call int1Ah
23335 00006F46 E814EBFFFF              <1>       call  RTC_40 ; GET RTC DATE
23336                                   <1>
23337 00006F4B 88D0                    <1>       mov   al, dl
23338 00006F4D E8BB9CFFFF              <1>       call  bcd_to_ascii
23339 00006F52 66A3[12080100]          <1>       mov   [Day], ax
23340                                   <1>
23341 00006F58 88F0                    <1>       mov   al, dh
23342 00006F5A E8AE9CFFFF              <1>       call  bcd_to_ascii
23343 00006F5F 66A3[15080100]          <1>       mov   [Month], ax
23344                                   <1>
23345 00006F65 88E8                    <1>       mov   al, ch
23346 00006F67 E8A19CFFFF              <1>       call  bcd_to_ascii
23347 00006F6C 66A3[18080100]          <1>       mov   [Century], ax
23348                                   <1>
23349 00006F72 88C8                    <1>       mov   al, cl
23350 00006F74 E8949CFFFF              <1>       call  bcd_to_ascii
23351 00006F79 66A3[1A080100]          <1>       mov   word [Year], ax
23352                                   <1>
23353 00006F7F BE[02080100]            <1>       mov   esi, Msg_Show_Date
23354 00006F84 E8D4F3FFFF              <1>       call  print_msg
23355                                   <1>
23356 00006F89 C3                      <1>       retn
23357                                   <1>
23358                                   <1> set_date:
23359                                   <1>       ; 13/05/2016
23360                                   <1>       ; 18/01/2016 (TRDOS 386 = TRDOS v2.0)
23361                                   <1>         ; 2004-2005
23362                                   <1>
23363 00006F8A BE[E6070100]            <1>       mov   esi, Msg_Enter_Date
23364 00006F8F E8C9F3FFFF              <1>       call  print_msg
23365                                   <1>
23366                                   <1> loc_enter_day_1:
23367 00006F94 30E4                    <1>       xor     ah, ah
23368 00006F96 E87B9CFFFF              <1>       call  int16h
23369                                   <1>       ; AL = ASCII Code of the Character
23370 00006F9B 3C0D                    <1>       cmp   al, 13
23371 00006F9D 0F84B7010000            <1>       je    loc_set_date_retn
23372 00006FA3 3C1B                    <1>       cmp   al, 27
23373 00006FA5 0F84AF010000            <1>       je    loc_set_date_retn
23374 00006FAB A2[12080100]            <1>       mov   [Day], al
23375 00006FB0 3C30                    <1>       cmp   al, '0'
23376 00006FB2 0F82AD010000            <1>       jb    loc_set_date_stc_0
```

```
23377 00006FB8 3C33                <1>       cmp    al, '3'
23378 00006FBA 0F87A5010000        <1>       ja     loc_set_date_stc_0
23379                              <1>       ; 13/05/2016
23380                              <1>       ;mov   bx, 7 ; attribute/color (bl)
23381                              <1>                    ; video page 0 (bh)
23382 00006FC0 B307                <1>       mov    bl, 7
23383 00006FC2 E8EBACFFFF          <1>       call   _write_tty
23384                              <1> loc_enter_day_2:
23385 00006FC7 30E4                <1>       xor     ah, ah
23386 00006FC9 E8489CFFFF          <1>       call   int16h
23387                              <1>       ; AL = ASCII Code of the Character
23388 00006FCE 3C1B                <1>       cmp    al, 27
23389 00006FD0 0F8484010000        <1>         je      loc_set_date_retn
23390 00006FD6 A2[13080100]        <1>       mov    [Day+1], al
23391 00006FDB 3C30                <1>       cmp    al, '0'
23392 00006FDD 0F828C010000        <1>         jb      loc_set_date_stc_1
23393 00006FE3 3C39                <1>       cmp    al, '9'
23394 00006FE5 0F8784010000        <1>         ja      loc_set_date_stc_1
23395 00006FEB 803D[12080100]33    <1>       cmp    byte [Day], '3'
23396 00006FF2 7208                <1>       jb     short pass_set_day_31
23397 00006FF4 3C31                <1>       cmp    al, '1'
23398 00006FF6 0F8773010000        <1>         ja      loc_set_date_stc_1
23399                              <1> pass_set_day_31:
23400                              <1>       ; 13/05/2016
23401                              <1>       ;mov   bx, 7 ; attribute/color (bl)
23402                              <1>                    ; video page 0 (bh)
23403 00006FFC B307                <1>       mov    bl, 7
23404 00006FFE E8AFACFFFF          <1>       call   _write_tty
23405                              <1> loc_enter_separator_1:
23406 00007003 28E4                <1>       sub     ah, ah ; 0
23407 00007005 E80C9CFFFF          <1>       call   int16h
23408                              <1>       ; AL = ASCII Code of the Character
23409 0000700A 3C1B                <1>       cmp    al, 27
23410 0000700C 0F8448010000        <1>         je      loc_set_date_retn
23411 00007012 3C2D                <1>       cmp    al, '-'
23412 00007014 7408                <1>       je     short pass_set_date_separator_1
23413 00007016 3C2F                <1>       cmp    al, '/'
23414 00007018 0F856C010000        <1>         jne     loc_set_date_stc_2
23415                              <1> pass_set_date_separator_1:
23416                              <1>       ; 13/05/2016
23417                              <1>       ;mov   bx, 7 ; attribute/color (bl)
23418                              <1>                    ; video page 0 (bh)
23419 0000701E B307                <1>       mov    bl, 7
23420 00007020 E88DACFFFF          <1>       call   _write_tty
23421                              <1> loc_enter_month_1:
23422 00007025 30E4                <1>       xor     ah, ah ; 0
23423 00007027 E8EA9BFFFF          <1>       call   int16h
23424                              <1>       ; AL = ASCII Code of the Character
23425 0000702C 3C1B                <1>       cmp    al, 27
23426 0000702E 0F8426010000        <1>         je      loc_set_date_retn
23427 00007034 A2[15080100]        <1>       mov    [Month], al
23428 00007039 3C30                <1>       cmp    al, '0'
23429 0000703B 0F8264010000        <1>         jb      loc_set_date_stc_3
23430 00007041 3C31                <1>       cmp    al, '1'
23431 00007043 0F875C010000        <1>         ja      loc_set_date_stc_3
23432                              <1>       ; 13/05/2016
23433                              <1>       ;mov   bx, 7 ; attribute/color (bl)
23434                              <1>                    ; video page 0 (bh)
23435 00007049 B307                <1>       mov    bl, 7
23436 0000704B E862ACFFFF          <1>       call   _write_tty
23437                              <1> loc_enter_month_2:
23438 00007050 30E4                <1>       xor     ah, ah
23439 00007052 E8BF9BFFFF          <1>       call   int16h
23440                              <1>       ; AL = ASCII Code of the Character
23441 00007057 3C1B                <1>       cmp    al, 27
23442 00007059 0F84FB000000        <1>         je      loc_set_date_retn
23443 0000705F A2[16080100]        <1>       mov    [Month+1], al
23444 00007064 3C30                <1>       cmp    al, '0'
23445 00007066 0F8254010000        <1>         jb      loc_set_date_stc_4
23446 0000706C 3C39                <1>       cmp    al, '9'
23447 0000706E 0F874C010000        <1>         ja      loc_set_date_stc_4
23448 00007074 803D[15080100]31    <1>       cmp    byte [Month], '1'
23449 0000707B 7208                <1>       jb     short pass_set_month_12
23450 0000707D 3C32                <1>       cmp    al, '2'
23451 0000707F 0F873B010000        <1>         ja      loc_set_date_stc_4
23452                              <1> pass_set_month_12:
23453                              <1>       ; 13/05/2016
23454                              <1>       ;mov   bx, 7 ; attribute/color (bl)
23455                              <1>                    ; video page 0 (bh)
23456 00007085 B307                <1>       mov    bl, 7
23457 00007087 E826ACFFFF          <1>       call   _write_tty
23458                              <1> loc_enter_separator_2:
23459 0000708C 28E4                <1>       sub     ah, ah
23460 0000708E E8839BFFFF          <1>       call   int16h
23461                              <1>       ; AL = ASCII Code of the Character
23462 00007093 3C1B                <1>       cmp    al, 27
23463 00007095 0F84BF000000        <1>         je      loc_set_date_retn
23464 0000709B 3C2D                <1>       cmp    al, '-'
23465 0000709D 7408                <1>       je     short pass_set_date_separator_2
23466 0000709F 3C2F                <1>       cmp    al, '/'
23467 000070A1 0F8534010000        <1>         jne     loc_set_date_stc_5
23468                              <1> pass_set_date_separator_2:
23469                              <1>       ; 13/05/2016
23470                              <1>       ;mov   bx, 7 ; attribute/color (bl)
23471                              <1>                    ; video page 0 (bh)
23472 000070A7 B307                <1>       mov    bl, 7
23473 000070A9 E804ACFFFF          <1>       call   _write_tty
23474                              <1> loc_enter_year_1:
23475 000070AE 30E4                <1>       xor    ah, ah
23476 000070B0 E8619BFFFF          <1>       call   int16h
23477                              <1>       ; AL = ASCII Code of the Character
23478 000070B5 3C1B                <1>       cmp    al, 27
23479 000070B7 0F849D000000        <1>         je      loc_set_date_retn
```

```
23480 000070BD A2[1A080100]       <1>       mov   [Year], al
23481 000070C2 3C30               <1>       cmp   al, '0'
23482 000070C4 0F822C010000       <1>         jb      loc_set_date_stc_6
23483 000070CA 3C39               <1>       cmp   al, '9'
23484 000070CC 0F8724010000       <1>         ja      loc_set_date_stc_6
23485                             <1>       ; 13/05/2016
23486                             <1>       ;mov   bx, 7 ; attribute/color (bl)
23487                             <1>                    ; video page 0 (bh)
23488 000070D2 B307               <1>       mov   bl, 7
23489 000070D4 E8D9ABFFFF         <1>       call  _write_tty
23490                             <1> loc_enter_year_2:
23491 000070D9 30E4               <1>       xor   ah, ah
23492 000070DB E8369BFFFF         <1>       call  int16h
23493                             <1>       ; AL = ASCII Code of the Character
23494 000070E0 3C1B               <1>       cmp   al, 27
23495 000070E2 7476               <1>       je    short loc_set_date_retn
23496 000070E4 A2[1B080100]       <1>       mov   byte [Year+1], al
23497 000070E9 3C30               <1>       cmp   al, '0'
23498 000070EB 0F8220010000       <1>         jb      loc_set_date_stc_7
23499 000070F1 3C39               <1>       cmp   al, '9'
23500 000070F3 0F8718010000       <1>         ja      loc_set_date_stc_7
23501                             <1>       ; 13/05/2016
23502                             <1>       ;mov   bx, 7 ; attribute/color (bl)
23503                             <1>                    ; video page 0 (bh)
23504 000070F9 B307               <1>       mov   bl, 7
23505 000070FB E8B2ABFFFF         <1>       call  _write_tty
23506                             <1> loc_set_date_get_lchar_again:
23507 00007100 28E4               <1>       sub   ah, ah ; 0
23508 00007102 E80F9BFFFF         <1>       call  int16h
23509                             <1>       ; AL = ASCII Code of the Character
23510 00007107 3C0D               <1>       cmp   al, 13 ; ENTER key
23511 00007109 7412               <1>       je    short loc_set_date_progress
23512 0000710B 3C1B               <1>       cmp   al, 27 ; ESC key
23513 0000710D 744B               <1>       je    short loc_set_date_retn
23514                             <1>       ;
23515 0000710F E82A010000         <1>       call  check_for_backspace
23516 00007114 75EA               <1>       jne   short loc_set_date_get_lchar_again
23517                             <1>
23518                             <1> loc_set_date_bs_8:
23519 00007116 E811010000         <1>       call  write_backspace
23520 0000711B EBBC               <1>       jmp   short loc_enter_year_2
23521                             <1>
23522                             <1> loc_set_date_progress:
23523                             <1>       ; Get Current Date
23524                             <1>       ;mov   ah, 04h
23525                             <1>       ;call  int1Ah
23526 0000711D E83DE9FFFF         <1>       call  RTC_40 ; GET RTC DATE
23527                             <1>       ; CH = century (in BCD)
23528                             <1>
23529 00007122 66A1[1A080100]     <1>       mov   ax, [Year]
23530 00007128 662D3030           <1>       sub   ax, '00'
23531 0000712C C0E004             <1>       shl   al, 4 ; * 16
23532 0000712F 88C1               <1>       mov   cl, al
23533 00007131 00E1               <1>       add   cl, ah
23534 00007133 66A1[15080100]     <1>       mov   ax, [Month]
23535 00007139 662D3030           <1>       sub   ax, '00'
23536 0000713D C0E004             <1>       shl   al, 4 ; * 16
23537 00007140 88C6               <1>       mov   dh, al
23538 00007142 00E6               <1>       add   dh, ah
23539 00007144 66A1[12080100]     <1>       mov   ax, [Day]
23540 0000714A 662D3030           <1>       sub   ax, '00'
23541 0000714E C0E004             <1>       shl   al, 4 ; * 16
23542 00007151 88C2               <1>       mov   dl, al
23543 00007153 00E2               <1>       add   dl, ah
23544                             <1>
23545                             <1>       ;mov   ah, 05h
23546                             <1>       ;call  int1Ah
23547 00007155 E832E9FFFF         <1>       call  RTC_50 ; SET RTC DATE
23548                             <1>
23549                             <1> loc_set_date_retn:
23550 0000715A BE[6F130100]       <1>       mov   esi, nextline
23551 0000715F E8F9F1FFFF         <1>       call  print_msg
23552 00007164 C3                 <1>       retn
23553                             <1>
23554                             <1> loc_set_date_stc_0:
23555                             <1>       ;xor   bh, bh ; video page 0
23556 00007165 E828ACFFFF         <1>       call  beeper ; BEEP !
23557 0000716A E925FEFFFF         <1>         jmp     loc_enter_day_1
23558                             <1> loc_set_date_stc_1:
23559 0000716F E8CA000000         <1>       call  check_for_backspace
23560 00007174 740A               <1>       je    short loc_set_date_bs_1
23561                             <1>       ;xor   bh, bh ; video page 0
23562 00007176 E817ACFFFF         <1>       call  beeper ; BEEP !
23563 0000717B E947FEFFFF         <1>         jmp     loc_enter_day_2
23564                             <1> loc_set_date_bs_1:
23565 00007180 E8A7000000         <1>       call  write_backspace
23566 00007185 E90AFEFFFF         <1>         jmp     loc_enter_day_1
23567                             <1> loc_set_date_stc_2:
23568 0000718A E8AF000000         <1>       call  check_for_backspace
23569 0000718F 740A               <1>       je    short loc_set_date_bs_2
23570                             <1>       ;xor   bh, bh ; video page 0
23571 00007191 E8FCABFFFF         <1>       call  beeper ; BEEP !
23572 00007196 E968FEFFFF         <1>         jmp     loc_enter_separator_1
23573                             <1> loc_set_date_bs_2:
23574 0000719B E88C000000         <1>       call  write_backspace
23575 000071A0 E922FEFFFF         <1>         jmp     loc_enter_day_2
23576                             <1> loc_set_date_stc_3:
23577 000071A5 E894000000         <1>       call  check_for_backspace
23578 000071AA 740A               <1>       je short loc_set_date_bs_3
23579                             <1>       ;xor   bh, bh ; video page 0
23580 000071AC E8E1ABFFFF         <1>       call  beeper ; BEEP !
23581 000071B1 E96FFEFFFF         <1>         jmp     loc_enter_month_1
23582                             <1> loc_set_date_bs_3:
```

```
23583 000071B6 E871000000          <1>       call   write_backspace
23584 000071BB E943FEFFFF          <1>         jmp     loc_enter_separator_1
23585                              <1> loc_set_date_stc_4:
23586 000071C0 E879000000          <1>       call   check_for_backspace
23587 000071C5 740A                <1>       je     short loc_set_date_bs_4
23588                              <1>       ;xor   bh, bh ; video page 0
23589 000071C7 E8C6ABFFFF          <1>       call   beeper ; BEEP !
23590 000071CC E97FFEFFFF          <1>         jmp     loc_enter_month_2
23591                              <1> loc_set_date_bs_4:
23592 000071D1 E856000000          <1>       call   write_backspace
23593 000071D6 E94AFEFFFF          <1>         jmp     loc_enter_month_1
23594                              <1> loc_set_date_stc_5:
23595 000071DB E85E000000          <1>       call   check_for_backspace
23596 000071E0 740A                <1>       je     short loc_set_date_bs_5
23597                              <1>       ;xor   bh, bh ; video page 0
23598 000071E2 E8ABABFFFF          <1>       call   beeper ; BEEP !
23599 000071E7 E9A0FEFFFF          <1>         jmp     loc_enter_separator_2
23600                              <1> loc_set_date_bs_5:
23601 000071EC E83B000000          <1>       call   write_backspace
23602 000071F1 E95AFEFFFF          <1>         jmp     loc_enter_month_2
23603                              <1> loc_set_date_stc_6:
23604 000071F6 E843000000          <1>       call   check_for_backspace
23605 000071FB 740A                <1>         je      short  loc_set_date_bs_6
23606                              <1>       ;xor   bh, bh ; video page 0
23607 000071FD E890ABFFFF          <1>       call   beeper ; BEEP !
23608 00007202 E9A7FEFFFF          <1>         jmp     loc_enter_year_1
23609                              <1> loc_set_date_bs_6:
23610 00007207 E820000000          <1>       call   write_backspace
23611 0000720C E97BFEFFFF          <1>         jmp     loc_enter_separator_2
23612                              <1> loc_set_date_stc_7:
23613 00007211 E828000000          <1>       call   check_for_backspace
23614 00007216 740A                <1>       je     short loc_set_date_bs_7
23615                              <1>       ;xor   bh, bh ; video page 0
23616 00007218 E875ABFFFF          <1>       call   beeper ; BEEP !
23617 0000721D E9B7FEFFFF          <1>         jmp     loc_enter_year_2
23618                              <1> loc_set_date_bs_7:
23619 00007222 E805000000          <1>       call   write_backspace
23620 00007227 E982FEFFFF          <1>         jmp     loc_enter_year_1
23621                              <1>
23622                              <1> write_backspace:
23623                              <1>       ; 18/01/2016 (TRDOS 386 = TRDOS v2.0)
23624 0000722C B008                <1>       mov    al, 08h ; BACKSPACE
23625                              <1>       ; 13/05/2016
23626 0000722E 66BB0700            <1>       mov    bx, 7 ; bl = attribute/color
23627                              <1>                    ; bh = video page = 0
23628 00007232 E87BAAFFFF          <1>       call   _write_tty
23629 00007237 B020                <1>       mov    al, 20h ; BLANK/SPACE char
23630                              <1>       ;mov   bx, 7 ; attribute/color
23631                              <1>       ;call  _write_c_current
23632                              <1>       ;retn
23633 00007239 E9FBA9FFFF          <1>       jmp    _write_c_current
23634                              <1>
23635                              <1> check_for_backspace:
23636                              <1>       ; 18/01/2016 (TRDOS 386 = TRDOS v2.0)
23637 0000723E 663D080E            <1>       cmp    ax, 0E08h
23638 00007242 7410                <1>       je     short cfbs_retn
23639 00007244 663DE04B            <1>       cmp    ax, 4BE0h
23640 00007248 740A                <1>       je     short cfbs_retn
23641 0000724A 663D004B            <1>       cmp    ax, 4B00h
23642 0000724E 7404                <1>       je     short cfbs_retn
23643 00007250 663DE053            <1>       cmp    ax, 53E0h
23644                              <1> cfbs_retn:
23645 00007254 C3                  <1>       retn
23646                              <1>
23647                              <1> show_time:
23648                              <1>       ; 18/01/2016 (TRDOS 386 = TRDOS v2.0)
23649                              <1>         ; 2004-2005
23650                              <1>
23651                              <1>       ;mov   ah, 02h
23652                              <1>       ;call  int1Ah
23653 00007255 E894E7FFFF          <1>       call   RTC_20 ; GET RTC TIME
23654                              <1>
23655 0000725A 88E8                <1>       mov    al, ch
23656 0000725C E8AC99FFFF          <1>       call   bcd_to_ascii
23657 00007261 66A3[40080100]      <1>       mov    [Hour], ax
23658                              <1>
23659 00007267 88C8                <1>       mov    al, cl
23660 00007269 E89F99FFFF          <1>       call   bcd_to_ascii
23661 0000726E 66A3[43080100]      <1>       mov    [Minute], ax
23662                              <1>
23663 00007274 88F0                <1>       mov    al, dh
23664 00007276 E89299FFFF          <1>       call   bcd_to_ascii
23665 0000727B 66A3[46080100]      <1>       mov    [Second], ax
23666                              <1>
23667 00007281 BE[30080100]        <1>       mov    esi, Msg_Show_Time
23668 00007286 E8D2F0FFFF          <1>       call   print_msg
23669 0000728B C3                  <1>       retn
23670                              <1>
23671                              <1> set_time:
23672                              <1>       ; 13/05/2016
23673                              <1>       ; 18/01/2016 (TRDOS 386 = TRDOS v2.0)
23674                              <1>         ; 2004-2005
23675                              <1>
23676 0000728C BE[1F080100]        <1>       mov    esi, Msg_Enter_Time
23677 00007291 E8C7F0FFFF          <1>       call   print_msg
23678                              <1>
23679                              <1> loc_enter_hour_1:
23680 00007296 30E4                <1>       xor     ah, ah
23681 00007298 E87999FFFF          <1>       call   int16h
23682                              <1>       ; AL = ASCII Code of the Character
23683 0000729D 3C0D                <1>       cmp    al, 13 ; ENTER key
23684 0000729F 0F84AE010000        <1>         je      loc_set_time_retn
23685 000072A5 3C1B                <1>       cmp    al, 27 ; ESC key
```

```
23686 000072A7 0F84A6010000        <1>         je      loc_set_time_retn
23687 000072AD A2[40080100]        <1>       mov   [Hour], al
23688 000072B2 3C30                <1>       cmp   al, '0'
23689 000072B4 0F82A4010000        <1>         jb      loc_set_time_stc_0
23690 000072BA 3C32                <1>       cmp   al, '2'
23691 000072BC 0F879C010000        <1>         ja      loc_set_time_stc_0
23692                              <1>       ; 13/05/2016
23693                              <1>       ;mov   bx, 7 ; attribute/color (bl)
23694                              <1>                    ; video page 0 (bh)
23695 000072C2 B307                <1>       mov   bl, 7
23696 000072C4 E8E9A9FFFF          <1>       call  _write_tty
23697                              <1> loc_enter_hour_2:
23698 000072C9 30E4                <1>       xor     ah, ah
23699 000072CB E84699FFFF          <1>       call  int16h
23700                              <1>       ; AL = ASCII Code of the Character
23701 000072D0 3C1B                <1>       cmp   al, 27
23702 000072D2 0F847B010000        <1>         je      loc_set_time_retn
23703 000072D8 A2[41080100]        <1>       mov   [Hour+1], al
23704 000072DD 3C30                <1>       cmp   al, '0'
23705 000072DF 0F8283010000        <1>         jb      loc_set_time_stc_1
23706 000072E5 3C39                <1>       cmp   al, '9'
23707 000072E7 0F877B010000        <1>       ja    loc_set_time_stc_1
23708 000072ED 803D[40080100]32    <1>         cmp     byte [Hour], '2'
23709 000072F4 7208                <1>       jb    short pass_set_time_24
23710 000072F6 3C34                <1>       cmp   al, '4'
23711 000072F8 0F876A010000        <1>         ja      loc_set_time_stc_1
23712                              <1> pass_set_time_24:
23713                              <1>       ; 13/05/2016
23714                              <1>       ;mov   bx, 7 ; attribute/color (bl)
23715                              <1>                    ; video page 0 (bh)
23716 000072FE B307                <1>       mov   bl, 7
23717 00007300 E8ADA9FFFF          <1>       call  _write_tty
23718                              <1> loc_enter_time_separator_1:
23719 00007305 28E4                <1>       sub    ah, ah ; 0
23720 00007307 E80A99FFFF          <1>       call  int16h
23721                              <1>       ; AL = ASCII Code of the Character
23722 0000730C 3C1B                <1>       cmp   al, 27
23723 0000730E 0F843F010000        <1>         je      loc_set_time_retn
23724 00007314 3C3A                <1>       cmp   al, ':'
23725 00007316 0F8567010000        <1>         jne     loc_set_time_stc_2
23726                              <1>       ; 13/05/2016
23727                              <1>       ;mov   bx, 7 ; attribute/color (bl)
23728                              <1>                    ; video page 0 (bh)
23729 0000731C B307                <1>       mov   bl, 7
23730 0000731E E88FA9FFFF          <1>       call  _write_tty
23731                              <1> loc_enter_minute_1:
23732 00007323 30E4                <1>       xor     ah, ah
23733 00007325 E8EC98FFFF          <1>       call  int16h
23734                              <1>       ; AL = ASCII Code of the Character
23735 0000732A 3C1B                <1>       cmp   al, 27
23736 0000732C 0F8421010000        <1>         je      loc_set_time_retn
23737 00007332 A2[43080100]        <1>       mov   [Minute], al
23738 00007337 3C30                <1>       cmp   al, '0'
23739 00007339 0F825F010000        <1>         jb      loc_set_time_stc_3
23740 0000733F 3C35                <1>       cmp   al, '5'
23741 00007341 0F8757010000        <1>         ja      loc_set_time_stc_3
23742                              <1>       ; 13/05/2016
23743                              <1>       ;mov   bx, 7 ; attribute/color (bl)
23744                              <1>                    ; video page 0 (bh)
23745 00007347 B307                <1>       mov   bl, 7
23746 00007349 E864A9FFFF          <1>       call  _write_tty
23747                              <1> loc_enter_minute_2:
23748 0000734E 30E4                <1>       xor     ah, ah
23749 00007350 E8C198FFFF          <1>       call  int16h
23750                              <1>       ; AL = ASCII Code of the Character
23751 00007355 3C1B                <1>       cmp   al, 27
23752 00007357 0F84F6000000        <1>         je      loc_set_time_retn
23753 0000735D A2[44080100]        <1>       mov   [Minute+1], al
23754 00007362 3C30                <1>       cmp   al, '0'
23755 00007364 0F824F010000        <1>         jb      loc_set_time_stc_4
23756 0000736A 3C39                <1>       cmp   al, '9'
23757 0000736C 0F8747010000        <1>         ja      loc_set_time_stc_4
23758                              <1>       ; 13/05/2016
23759                              <1>       ;mov   bx, 7 ; attribute/color (bl)
23760                              <1>                    ; video page 0 (bh)
23761 00007372 B307                <1>       mov   bl, 7
23762 00007374 E839A9FFFF          <1>       call  _write_tty
23763                              <1> loc_enter_time_separator_2:
23764 00007379 66C705[46080100]30- <1>       mov   word [Second], 3030h
23765 00007381 30                  <1>
23766 00007382 28E4                <1>       sub     ah, ah
23767 00007384 E88D98FFFF          <1>       call  int16h
23768                              <1>       ; AL = ASCII Code of the Character
23769 00007389 3C0D                <1>       cmp   al, 13
23770 0000738B 0F8485000000        <1>         je      loc_set_time_progress
23771 00007391 3C1B                <1>       cmp   al, 27
23772 00007393 0F84BA000000        <1>         je      loc_set_time_retn
23773 00007399 3C3A                <1>       cmp   al, ':'
23774 0000739B 0F8533010000        <1>         jne     loc_set_time_stc_5
23775                              <1>       ; 13/05/2016
23776                              <1>       ;mov   bx, 7 ; attribute/color (bl)
23777                              <1>                    ; video page 0 (bh)
23778 000073A1 B307                <1>       mov   bl, 7
23779 000073A3 E80AA9FFFF          <1>       call  _write_tty
23780                              <1> loc_enter_second_1:
23781 000073A8 30E4                <1>       xor     ah, ah
23782 000073AA E86798FFFF          <1>       call  int16h
23783                              <1>       ; AL = ASCII Code of the Character
23784 000073AF 3C0D                <1>       cmp   al, 13
23785 000073B1 7463                <1>       je    short loc_set_time_progress
23786 000073B3 3C1B                <1>       cmp   al, 27
23787 000073B5 0F8498000000        <1>         je      loc_set_time_retn
23788 000073BB A2[46080100]        <1>       mov   [Second], al
```

```
23789 000073C0 3C30              <1>       cmp    al, '0'
23790 000073C2 0F8227010000      <1>         jb      loc_set_time_stc_6
23791 000073C8 3C35              <1>       cmp    al, '5'
23792 000073CA 0F871F010000      <1>         ja      loc_set_time_stc_6
23793                            <1>       ; 13/05/2016
23794                            <1>       ;mov   bx, 7 ; attribute/color (bl)
23795                            <1>                    ; video page 0 (bh)
23796 000073D0 B307              <1>       mov    bl, 7
23797 000073D2 E8DBA8FFFF        <1>       call   _write_tty
23798                            <1> loc_enter_second_2:
23799 000073D7 30E4              <1>       xor     ah, ah
23800 000073D9 E83898FFFF        <1>       call   int16h
23801                            <1>       ; AL = ASCII Code of the Character
23802 000073DE 3C1B              <1>       cmp    al, 27
23803 000073E0 7471              <1>       je     short loc_set_time_retn
23804 000073E2 3C30              <1>       cmp    al, '0'
23805 000073E4 0F8229010000      <1>         jb      loc_set_time_stc_7
23806 000073EA 3C39              <1>       cmp    al, '9'
23807 000073EC 0F8721010000      <1>         ja      loc_set_time_stc_7
23808                            <1>       ; 13/05/2016
23809                            <1>       ;mov   bx, 7 ; attribute/color (bl)
23810                            <1>                    ; video page 0 (bh)
23811 000073F2 B307              <1>       mov    bl, 7
23812 000073F4 E8B9A8FFFF        <1>       call   _write_tty
23813                            <1> loc_set_time_get_lchar_again:
23814 000073F9 28E4              <1>       sub    ah, ah ; 0
23815 000073FB E81698FFFF        <1>       call   int16h
23816                            <1>       ; AL = ASCII Code of the Character
23817 00007400 3C0D              <1>       cmp    al, 13
23818 00007402 7412              <1>       je     short loc_set_time_progress
23819 00007404 3C1B              <1>       cmp    al, 27
23820 00007406 744B              <1>       je     short loc_set_time_retn
23821                            <1>       ;
23822 00007408 E831FEFFFF        <1>       call   check_for_backspace
23823 0000740D 75EA              <1>       jne    short loc_set_time_get_lchar_again
23824                            <1>
23825                            <1> loc_set_time_bs_8:
23826 0000740F E818FEFFFF        <1>       call   write_backspace
23827 00007414 EBC1              <1>       jmp    short loc_enter_second_2
23828                            <1>
23829                            <1> loc_set_time_progress:
23830                            <1>       ; Get Current Time
23831                            <1>       ;mov   ah, 02h
23832                            <1>       ;call  int1Ah
23833 00007416 E8D3E5FFFF        <1>       call   RTC_20 ; GET RTC TIME
23834                            <1>       ;DL = Daylight Savings Enable option (0-1)
23835                            <1>
23836 0000741B 66A1[40080100]    <1>       mov    ax, [Hour]
23837 00007421 662D3030          <1>       sub    ax, '00'
23838 00007425 C0E004            <1>       shl    al, 4 ; * 16
23839 00007428 88C5              <1>       mov    ch, al
23840 0000742A 00E5              <1>       add    ch, ah
23841 0000742C 66A1[43080100]    <1>       mov    ax, [Minute]
23842 00007432 662D3030          <1>       sub    ax, '00'
23843 00007436 C0E004            <1>       shl    al, 4 ; * 16
23844 00007439 88C1              <1>       mov    cl, al
23845 0000743B 00E1              <1>       add    cl, ah
23846 0000743D 66A1[46080100]    <1>       mov    ax, [Second]
23847 00007443 662D3030          <1>       sub    ax, '00'
23848 00007447 C0E004            <1>       shl    al, 4 ; * 16
23849 0000744A 88C6              <1>       mov    dh, al
23850 0000744C 00E6              <1>       add    dh, ah
23851                            <1>
23852                            <1>       ;mov   ah, 03h
23853                            <1>       ;call  int1Ah
23854 0000744E E8CAE5FFFF        <1>       call   RTC_30 ; SET RTC TIME
23855                            <1>
23856                            <1> loc_set_time_retn:
23857 00007453 BE[6F130100]      <1>       mov    esi, nextline
23858 00007458 E800EFFFFF        <1>       call   print_msg
23859 0000745D C3                <1>       retn
23860                            <1>
23861                            <1> loc_set_time_stc_0:
23862                            <1>       ;xor   bh, bh ; video page 0
23863 0000745E E82FA9FFFF        <1>       call   beeper ; BEEP !
23864 00007463 E92EFEFFFF        <1>         jmp     loc_enter_hour_1
23865                            <1> loc_set_time_stc_1:
23866 00007468 E8D1FDFFFF        <1>       call   check_for_backspace
23867 0000746D 740A              <1>       je     short loc_set_time_bs_1
23868                            <1>       ;xor   bh, bh ; video page 0
23869 0000746F E81EA9FFFF        <1>       call   beeper ; BEEP !
23870 00007474 E950FEFFFF        <1>         jmp     loc_enter_hour_2
23871                            <1> loc_set_time_bs_1:
23872 00007479 E8AEFDFFFF        <1>       call   write_backspace
23873 0000747E E913FEFFFF        <1>         jmp     loc_enter_hour_1
23874                            <1> loc_set_time_stc_2:
23875 00007483 E8B6FDFFFF        <1>       call   check_for_backspace
23876 00007488 740A              <1>       je     short loc_set_time_bs_2
23877                            <1>       ;xor   bh, bh ; video page 0
23878 0000748A E803A9FFFF        <1>       call   beeper ; BEEP !
23879 0000748F E971FEFFFF        <1>         jmp     loc_enter_time_separator_1
23880                            <1> loc_set_time_bs_2:
23881 00007494 E893FDFFFF        <1>       call   write_backspace
23882 00007499 E92BFEFFFF        <1>         jmp     loc_enter_hour_2
23883                            <1> loc_set_time_stc_3:
23884 0000749E E89BFDFFFF        <1>       call   check_for_backspace
23885 000074A3 740A              <1>       je     short loc_set_time_bs_3
23886                            <1>       ;xor   bh, bh ; video page 0
23887 000074A5 E8E8A8FFFF        <1>       call   beeper ; BEEP !6
23888 000074AA E974FEFFFF        <1>         jmp     loc_enter_minute_1
23889                            <1> loc_set_time_bs_3:
23890 000074AF E878FDFFFF        <1>       call   write_backspace
23891 000074B4 E94CFEFFFF        <1>         jmp     loc_enter_time_separator_1
```

```
23892                                  <1> loc_set_time_stc_4:
23893 000074B9 E880FDFFFF              <1>       call   check_for_backspace
23894 000074BE 740A                    <1>       je     short loc_set_time_bs_4
23895                                  <1>       ;xor   bh, bh ; video page 0
23896 000074C0 E8CDA8FFFF              <1>       call   beeper ; BEEP !
23897 000074C5 E984FEFFFF              <1>         jmp     loc_enter_minute_2
23898                                  <1> loc_set_time_bs_4:
23899 000074CA E85DFDFFFF              <1>       call   write_backspace
23900 000074CF E94FFEFFFF              <1>         jmp     loc_enter_minute_1
23901                                  <1> loc_set_time_stc_5:
23902 000074D4 E865FDFFFF              <1>       call   check_for_backspace
23903 000074D9 740A                    <1>       je     short loc_set_time_bs_5
23904                                  <1>       ;xor   bh, bh ; video page 0
23905 000074DB E8B2A8FFFF              <1>       call   beeper ; BEEP !
23906 000074E0 E994FEFFFF              <1>         jmp     loc_enter_time_separator_2
23907                                  <1> loc_set_time_bs_5:
23908 000074E5 E842FDFFFF              <1>       call   write_backspace
23909 000074EA E95FFEFFFF              <1>         jmp     loc_enter_minute_2
23910                                  <1> loc_set_time_stc_6:
23911 000074EF E84AFDFFFF              <1>       call   check_for_backspace
23912 000074F4 7413                    <1>       je     short loc_set_time_bs_6
23913                                  <1>       ;xor   bh, bh ; video page 0
23914 000074F6 E897A8FFFF              <1>       call   beeper ; BEEP !
23915 000074FB 66C705[46080100]30-     <1>       mov    word [Second], 3030h
23916 00007503 30                      <1>
23917 00007504 E99FFEFFFF              <1>         jmp     loc_enter_second_1
23918                                  <1> loc_set_time_bs_6:
23919 00007509 E81EFDFFFF              <1>       call   write_backspace
23920 0000750E E966FEFFFF              <1>         jmp     loc_enter_time_separator_2
23921                                  <1> loc_set_time_stc_7:
23922 00007513 E826FDFFFF              <1>       call   check_for_backspace
23923 00007518 740A                    <1>       je     short loc_set_time_bs_7
23924                                  <1>       ;xor   bh, bh ; video page 0
23925 0000751A E873A8FFFF              <1>       call   beeper ; BEEP !
23926 0000751F E9B3FEFFFF              <1>         jmp     loc_enter_second_2
23927                                  <1> loc_set_time_bs_7:
23928 00007524 E803FDFFFF              <1>       call   write_backspace
23929 00007529 E97AFEFFFF              <1>         jmp     loc_enter_second_1
23930                                  <1>
23931                                  <1> print_volume_info:
23932                                  <1>       ; 01/03/2016
23933                                  <1>       ; 08/02/2016
23934                                  <1>       ; 06/02/2016
23935                                  <1>       ; 04/02/2016
23936                                  <1>       ; 18/01/2016 (TRDOS 386 = TRDOS v2.0)
23937                                  <1>       ; 25/10/2009
23938                                  <1>       ;
23939                                  <1>       ; "Volume Serial No: "
23940                                  <1>       ;
23941                                  <1>       ; INPUT  : AL = DOS Drive Number
23942                                  <1>       ; OUTPUT : AH = FS Type
23943                                  <1>       ;          AL = DOS Drive Name
23944                                  <1>       ; CF = 0 -> OK
23945                                  <1>       ; CF = 1 -> Drive not ready
23946                                  <1>
23947 0000752E 88C4                    <1>       mov    ah, al
23948 00007530 28C0                    <1>       sub    al, al
23949 00007532 0FB7F0                  <1>       movzx  esi, ax
23950 00007535 81C600010900            <1>       add    esi, Logical_DOSDisks
23951 0000753B 8A06                    <1>       mov    al, [esi]
23952 0000753D 3C41                    <1>       cmp    al, 'A'
23953 0000753F 7304                    <1>       jnb    short loc_pvi_set_vol_name
23954 00007541 8A6604                  <1>       mov    ah, [esi+LD_FSType]
23955 00007544 C3                      <1>       retn
23956                                  <1>
23957                                  <1> loc_pvi_set_vol_name:
23958 00007545 A2[7A080100]            <1>       mov    [Vol_Drv_Name], al
23959 0000754A 56                      <1>       push   esi
23960 0000754B E858010000              <1>       call   move_volume_name_and_serial_no ;;;
23961 00007550 7302                    <1>       jnc    short loc_pvi_mvn_ok
23962 00007552 5E                      <1>       pop    esi
23963 00007553 C3                      <1>       retn
23964                                  <1>
23965                                  <1> loc_pvi_mvn_ok:
23966 00007554 8B3424                  <1>       mov    esi, [esp]
23967 00007557 807E04A1                <1>       cmp    byte [esi+LD_FSType], 0A1h
23968 0000755B 7509                    <1>       jne    short loc_pvi_fat_vol_size
23969 0000755D 8B4670                  <1>       mov    eax, [esi+LD_FS_VolumeSize]
23970 00007560 0FB75E11                <1>       movzx  ebx, word [esi+LD_FS_BytesPerSec]
23971 00007564 EB07                    <1>       jmp    short loc_vol_size_mul32
23972                                  <1> loc_pvi_fat_vol_size:
23973 00007566 8B4670                  <1>       mov    eax, [esi+LD_TotalSectors]
23974 00007569 0FB75E11                <1>       movzx  ebx, word [esi+LD_BPB+BPB_BytsPerSec]
23975                                  <1> loc_vol_size_mul32:
23976 0000756D F7E3                    <1>       mul    ebx
23977 0000756F 09D2                    <1>       or     edx, edx
23978 00007571 7507                    <1>       jnz    short loc_vol_size_in_kbytes
23979                                  <1> loc_vol_size_in_bytes:
23980 00007573 B9[58080100]            <1>       mov    ecx, VolSize_Bytes
23981 00007578 EB0D                    <1>       jmp    short loc_write_vol_size_str
23982                                  <1> loc_vol_size_in_kbytes:
23983 0000757A 66BB0004                <1>       mov    bx, 1024
23984 0000757E F7F3                    <1>       div    ebx
23985 00007580 B9[4B080100]            <1>       mov    ecx, VolSize_KiloBytes
23986 00007585 31D2                    <1>       xor    edx, edx ; 0
23987                                  <1> loc_write_vol_size_str:
23988 00007587 890D[1F5B0100]          <1>       mov    [VolSize_Unit1], ecx
23989                                  <1>       ;
23990 0000758D BF[355B0100]            <1>       mov    edi, Vol_Tot_Sec_Str_End
23991                                  <1>         ;mov byte [edi], 0
23992 00007592 B90A000000              <1>       mov    ecx, 10
23993                                  <1> loc_write_vol_size_chr:
23994 00007597 F7F1                    <1>       div    ecx
```

```
23995 00007599 80C230              <1>       add    dl, '0'
23996 0000759C 4F                  <1>       dec    edi
23997 0000759D 8817                <1>       mov    [edi], dl
23998 0000759F 85C0                <1>       test   eax, eax
23999 000075A1 7404                <1>       jz     short loc_write_vol_size_str_ok
24000 000075A3 28D2                <1>       sub    dl, dl ; 0
24001 000075A5 EBF0                <1>       jmp    short loc_write_vol_size_chr
24002                              <1>
24003                              <1> loc_write_vol_size_str_ok:
24004 000075A7 893D[275B0100]      <1>       mov    [Vol_Tot_Sec_Str_Start], edi
24005                              <1>       ;
24006 000075AD BF[63080100]        <1>       mov    edi, Vol_FS_Name
24007 000075B2 8A4E03              <1>       mov    cl, [esi+LD_FATType]
24008 000075B5 20C9                <1>       and    cl, cl ; 0 ?
24009 000075B7 7515                <1>       jnz    short loc_write_vol_FAT_str_1
24010 000075B9 66C7075452          <1>       mov    word [edi], 'TR'
24011 000075BE C7470420465331      <1>       mov    dword [edi+4], ' FS1'
24012                              <1>       ;movzx ebx, word [esi+LD_FS_BytesPerSec]
24013 000075C5 668B5E11            <1>       mov    bx, [esi+LD_FS_BytesPerSec]
24014 000075C9 8B4674              <1>       mov    eax, [esi+LD_FS_FreeSectors]
24015 000075CC EB36                <1>       jmp    short loc_vol_freespace_mul32
24016                              <1>
24017                              <1> loc_write_vol_FAT_str_1:
24018 000075CE 66B83332            <1>       mov    ax, '32' ; FAT32
24019 000075D2 80F902              <1>       cmp    cl, 2 ; [esi+LD_FATType]
24020 000075D5 7708                <1>       ja     short loc_write_vol_FAT_str_2
24021 000075D7 66B83132            <1>       mov    ax, '12' ; FAT12
24022 000075DB 7202                <1>       jb     short loc_write_vol_FAT_str_2
24023 000075DD B436                <1>       mov    ah, '6'  ; FAT16
24024                              <1> loc_write_vol_FAT_str_2:
24025 000075DF C70746415420        <1>       mov    dword [edi], 'FAT '
24026 000075E5 66894704            <1>       mov    word [edi+4], ax
24027                              <1>       ;
24028                              <1>       ;movzx ebx, word [esi+LD_BPB+BPB_BytsPerSec]
24029 000075E9 668B5E11            <1>       mov    bx, [esi+LD_BPB+BPB_BytsPerSec]
24030 000075ED 8B4674              <1>       mov    eax, [esi+LD_FreeSectors]
24031                              <1>
24032                              <1> loc_vol_freespace_recalc0:
24033                              <1>       ; 01/03/2016
24034 000075F0 83F8FF              <1>       cmp    eax, 0FFFFFFFFh
24035 000075F3 720F                <1>       jb     short loc_vol_freespace_mul32
24036                              <1>       ;inc   eax ; 0
24037 000075F5 20C9                <1>       and    cl, cl ; byte [esi+LD_FATType]
24038 000075F7 740B                <1>       jz     short loc_vol_freespace_mul32
24039 000075F9 53                  <1>       push   ebx
24040 000075FA 66BB00FF            <1>       mov    bx, 0FF00h ; recalculate free sectors
24041 000075FE E8C0490000          <1>       call   calculate_fat_freespace
24042 00007603 5B                  <1>       pop    ebx
24043                              <1>
24044                              <1> loc_vol_freespace_mul32:
24045 00007604 F7E3                <1>       mul    ebx
24046 00007606 09D2                <1>       or     edx, edx
24047 00007608 7507                <1>       jnz    short loc_vol_fspace_in_kbytes
24048                              <1> loc_vol_fspace_in_bytes:
24049 0000760A B9[58080100]        <1>       mov    ecx, VolSize_Bytes
24050 0000760F EB0D                <1>       jmp    short loc_write_vol_fspace_str
24051                              <1> loc_vol_fspace_in_kbytes:
24052 00007611 66BB0004            <1>       mov    bx, 1024
24053 00007615 F7F3                <1>       div    ebx
24054 00007617 B9[4B080100]        <1>       mov    ecx, VolSize_KiloBytes
24055 0000761C 31D2                <1>       xor    edx, edx ; 0
24056                              <1> loc_write_vol_fspace_str:
24057 0000761E 890D[235B0100]      <1>       mov    [VolSize_Unit2], ecx
24058                              <1>       ;
24059 00007624 BF[455B0100]        <1>       mov    edi, Vol_Free_Sectors_Str_End
24060                              <1>         ;mov byte [edi], 0
24061 00007629 B90A000000          <1>       mov    ecx, 10
24062                              <1> loc_write_vol_fspace_chr:
24063 0000762E F7F1                <1>       div    ecx
24064 00007630 80C230              <1>       add    dl, '0'
24065 00007633 4F                  <1>       dec    edi
24066 00007634 8817                <1>       mov    [edi], dl
24067 00007636 85C0                <1>       test   eax, eax
24068 00007638 7404                <1>       jz     short loc_write_vol_fspace_str_ok
24069 0000763A 28D2                <1>       sub    dl, dl ; 0
24070 0000763C EBF0                <1>       jmp    short loc_write_vol_fspace_chr
24071                              <1>
24072                              <1> loc_write_vol_fspace_str_ok:
24073 0000763E 893D[375B0100]      <1>       mov    [Vol_Free_Sectors_Str_Start], edi
24074                              <1>       ;
24075 00007644 BE[61080100]        <1>       mov    esi, Volume_in_drive
24076 00007649 E80FEDFFFF          <1>       call   print_msg
24077 0000764E BE[A1080100]        <1>       mov    esi, Vol_Name
24078 00007653 E805EDFFFF          <1>       call   print_msg
24079 00007658 BE[6F130100]        <1>       mov    esi, nextline
24080 0000765D E8FBECFFFF          <1>       call   print_msg
24081                              <1>       ;
24082 00007662 BE[02090100]        <1>       mov    esi, Vol_Total_Sector_Header
24083 00007667 E8F1ECFFFF          <1>       call   print_msg
24084 0000766C 8B35[275B0100]      <1>       mov    esi, [Vol_Tot_Sec_Str_Start]
24085 00007672 E8E6ECFFFF          <1>       call   print_msg
24086 00007677 8B35[1F5B0100]      <1>       mov    esi, [VolSize_Unit1]
24087 0000767D E8DBECFFFF          <1>       call   print_msg
24088                              <1>       ;
24089 00007682 BE[13090100]        <1>       mov    esi, Vol_Free_Sectors_Header
24090 00007687 E8D1ECFFFF          <1>       call   print_msg
24091 0000768C 8B35[375B0100]      <1>       mov    esi, [Vol_Free_Sectors_Str_Start]
24092 00007692 E8C6ECFFFF          <1>       call   print_msg
24093 00007697 8B35[235B0100]      <1>       mov    esi, [VolSize_Unit2]
24094 0000769D E8BBECFFFF          <1>       call   print_msg
24095                              <1>       ;
24096 000076A2 5E                  <1>       pop    esi
24097                              <1>
```

```
24098                                   <1>       ;mov   ah, [esi+LD_FSType]
24099                                   <1>       ;mov   al, [esi+LD_FATType]
24100 000076A3 668B4603                 <1>       mov    ax, [esi+LD_FATType]
24101                                   <1>
24102 000076A7 C3                       <1>       retn
24103                                   <1>
24104                                   <1> move_volume_name_and_serial_no:
24105                                   <1>       ; 08/02/2016  (TRDOS 386 = TRDOS v2.0)
24106                                   <1>       ; this routine will be called by
24107                                   <1>       ; "print_volume_info" and "print_directory"
24108                                   <1>       ; INPUT ->
24109                                   <1>       ;     ESI = Logical DOS drv descripton table address
24110                                   <1>       ; OUTPUT ->
24111                                   <1>       ;     *Volume name will be moved to text area
24112                                   <1>       ;     *Volume serial number will be converted to
24113                                   <1>       ;      text and will be moved to text area
24114                                   <1>       ;   cf = 1 -> invalid/unknown dos drive
24115                                   <1>       ;   cf = 0 -> ecx = 0
24116                                   <1>       ;
24117                                   <1>       ; (eax, edx, ecx, esi, edi will be changed)
24118                                   <1>
24119 000076A8 BF[A1080100]            <1>       mov    edi, Vol_Name
24120                                   <1>
24121                                   <1>       ;mov   ah, [esi+LD_FSType]
24122                                   <1>       ;mov   al, [esi+LD_FATType]
24123 000076AD 668B4603                <1>       mov    ax, [esi+LD_FATType]
24124 000076B1 80FCA1                  <1>       cmp    ah, 0A1h
24125 000076B4 7418                    <1>       je     short mvn_2
24126 000076B6 08E4                    <1>       or     ah, ah
24127 000076B8 7404                    <1>       jz     short mvn_0
24128 000076BA 08C0                    <1>       or     al, al
24129 000076BC 7504                    <1>       jnz    short mvn_1
24130                                   <1> mvn_0:
24131 000076BE 8A06                    <1>       mov    al, [esi]
24132 000076C0 F9                      <1>       stc
24133 000076C1 C3                      <1>       retn
24134                                   <1> mvn_1:
24135 000076C2 3C02                    <1>       cmp    al, 2
24136 000076C4 7717                    <1>       ja     short mvn_3
24137                                   <1>       ;or    al, al
24138                                   <1>       ;jz    short mvn_2
24139 000076C6 8B462D                  <1>       mov    eax, [esi+LD_BPB+VolumeID]
24140 000076C9 83C631                  <1>       add    esi, LD_BPB+VolumeLabel
24141 000076CC EB15                    <1>       jmp    short mvn_4
24142                                   <1> mvn_2:
24143 000076CE 8B4628                  <1>       mov    eax, [esi+LD_FS_VolumeSerial]
24144 000076D1 83C62C                  <1>       add    esi, LD_FS_VolumeName
24145 000076D4 B910000000              <1>       mov    ecx, 16
24146 000076D9 F3A5                    <1>       rep    movsd
24147 000076DB EB10                    <1>       jmp    short mvn_5
24148                                   <1> mvn_3:
24149 000076DD 8B4649                  <1>       mov    eax, [esi+LD_BPB+FAT32_VolID]
24150 000076E0 83C64D                  <1>       add    esi, LD_BPB+FAT32_VolLab
24151                                   <1> mvn_4:
24152 000076E3 B90B000000              <1>       mov    ecx, 11
24153 000076E8 F3A4                    <1>       rep    movsb
24154 000076EA C60700                  <1>       mov    byte [edi], 0
24155                                   <1> mvn_5:
24156                                   <1>       ;mov   [Current_VolSerial], eax
24157 000076ED E817BCFFFF              <1>       call   dwordtohex
24158 000076F2 8915[F6080100]          <1>       mov    [Vol_Serial1], edx
24159 000076F8 A3[FB080100]            <1>       mov    [Vol_Serial2], eax
24160                                   <1>       ; ecx = 0
24161 000076FD C3                      <1>       retn
24162                                   <1>
24163                                   <1> get_volume_serial_number:
24164                                   <1>       ; 19/01/2016 (TRDOS 386 = TRDOS v2.0)
24165                                   <1>       ; 08/08/2010
24166                                   <1>       ;
24167                                   <1>       ; INPUT -> DL = Logical DOS Drive number
24168                                   <1>       ; OUTPUT -> EAX = Volume serial number
24169                                   <1>       ;          BL= FAT Type
24170                                   <1>       ;          BH = Logical DOS drv Number (DL input)
24171                                   <1>       ; cf = 1 -> Drive not ready
24172                                   <1>
24173 000076FE 31DB                    <1>       xor    ebx, ebx
24174 00007700 88D7                    <1>       mov    bh, dl
24175 00007702 3815[D2060100]          <1>       cmp    [Last_DOS_DiskNo], dl
24176 00007708 7304                    <1>       jnb    short loc_gvsn_start
24177                                   <1> loc_gvsn_stc_retn:
24178 0000770A 31C0                    <1>       xor    eax, eax
24179 0000770C F9                      <1>       stc
24180 0000770D C3                      <1>         retn
24181                                   <1> loc_gvsn_start:
24182 0000770E 56                      <1>       push   esi
24183 0000770F BE00010900              <1>       mov    esi, Logical_DOSDisks
24184 00007714 01DE                    <1>       add    esi, ebx
24185 00007716 8A5E03                  <1>       mov    bl, [esi+LD_FATType]
24186 00007719 20DB                    <1>       and    bl, bl
24187 0000771B 740F                    <1>       jz     short loc_gvsn_fs
24188 0000771D 80FB02                  <1>       cmp    bl, 2
24189 00007720 7705                    <1>       ja     short loc_gvsn_fat32
24190                                   <1> loc_gvsn_fat:
24191 00007722 83C62D                  <1>       add    esi, LD_BPB + VolumeID
24192 00007725 EB0E                    <1>       jmp    short loc_gvsn_return
24193                                   <1> loc_gvsn_fat32:
24194 00007727 83C649                  <1>       add    esi, LD_BPB + FAT32_VolID
24195 0000772A EB09                    <1>       jmp    short loc_gvsn_return
24196                                   <1> loc_gvsn_fs:
24197 0000772C 807E04A1                <1>       cmp    byte [esi+LD_FSType], 0A1h
24198 00007730 75D8                    <1>       jne    short loc_gvsn_stc_retn
24199 00007732 83C628                  <1>       add    esi, LD_FS_VolumeSerial
24200                                   <1> loc_gvsn_return:
```

```
24201 00007735 8B06              <1>       mov    eax, [esi]
24202 00007737 5E                <1>       pop    esi
24203 00007738 C3                <1>       retn
24204                            <1>
24205                            <1> ; CMD_INTR.ASM [ TRDOS Command Interpreter Procedure ]
24206                            <1> ; 09/11/2011
24207                            <1> ; 29/01/2005
24208                            <1>
24209                            <1> command_interpreter:
24210                            <1>       ; 16/10/2016
24211                            <1>       ; 12/10/2016
24212                            <1>       ; 13/05/2016
24213                            <1>       ; 07/05/2016
24214                            <1>       ; 04/03/2016
24215                            <1>       ; 04/02/2016
24216                            <1>       ; 03/02/2016
24217                            <1>       ; 30/01/2016
24218                            <1>       ; 29/01/2016 (TRDOS 386 = TRDOS 2.0)
24219                            <1>       ; 15/09/2011
24220                            <1>       ; 29/01/2005
24221                            <1>
24222                            <1>       ; Input: ecx = command word length (CL)
24223                            <1>       ;      CommandBuffer = Command string offset
24224                            <1>
24225 00007739 C605[D85B0100]00  <1>       mov    byte [Program_Exit],0
24226 00007740 80F904            <1>       cmp    cl, 4
24227 00007743 0F87B7020000      <1>         ja     c_6
24228 00007749 0F8237010000      <1>         jb     c_2
24229                            <1> c_4:
24230                            <1>
24231                            <1> cmp_cmd_exit:
24232 0000774F BF[40070100]      <1>       mov    edi, Cmd_Exit
24233 00007754 E8C3030000        <1>       call   cmp_cmd
24234 00007759 7208              <1>       jc     short cmp_cmd_date
24235                            <1>
24236 0000775B C605[D85B0100]01  <1>         mov     byte [Program_Exit], 1
24237 00007762 C3                <1>         retn
24238                            <1>
24239                            <1> cmp_cmd_date:
24240 00007763 B104              <1>       mov    cl, 4
24241 00007765 BF[5C070100]      <1>       mov    edi, Cmd_Date
24242 0000776A E8AD030000        <1>       call   cmp_cmd
24243 0000776F 720B              <1>         jc   short cmp_cmd_time
24244                            <1>
24245 00007771 E8D0F7FFFF        <1>       call   show_date
24246 00007776 E80FF8FFFF        <1>       call   set_date
24247 0000777B C3                <1>       retn
24248                            <1>
24249                            <1> cmp_cmd_time:
24250 0000777C B104              <1>       mov    cl, 4
24251 0000777E BF[61070100]      <1>       mov    edi, Cmd_Time
24252 00007783 E894030000        <1>       call   cmp_cmd
24253 00007788 720B              <1>       jc     short cmp_cmd_show
24254                            <1>
24255 0000778A E8C6FAFFFF        <1>       call   show_time
24256 0000778F E8F8FAFFFF        <1>       call   set_time
24257 00007794 C3                <1>       retn
24258                            <1>
24259                            <1> cmp_cmd_show:
24260 00007795 B104              <1>       mov    cl, 4
24261 00007797 BF[72070100]      <1>       mov    edi, Cmd_Show
24262 0000779C E87B030000        <1>       call   cmp_cmd
24263 000077A1 0F83F8090000      <1>         jnc     show_file
24264                            <1>
24265                            <1> cmp_cmd_echo:
24266 000077A7 B104              <1>       mov    cl, 4
24267 000077A9 BF[AE070100]      <1>       mov    edi, Cmd_Echo
24268 000077AE E869030000        <1>       call   cmp_cmd
24269 000077B3 7224              <1>       jc     short cmp_cmd_copy
24270                            <1>
24271                            <1>       ; 22/11/2017
24272                            <1>       ; AL = 0
24273 000077B5 803E20            <1>       cmp    byte [esi], 20h
24274 000077B8 7215              <1>       jb     short cmd_echo_nextline
24275                            <1>       ; 14/04/2016
24276 000077BA 56                <1>       push   esi
24277                            <1> cmd_echo_asciiz:
24278                            <1>       ;inc   esi
24279                            <1>       ;mov   al, [esi]
24280                            <1>       ; 22/11/2017
24281 000077BB AC                <1>       lodsb
24282 000077BC 3C20              <1>       cmp    al, 20h
24283 000077BE 73FB              <1>       jnb    short cmd_echo_asciiz
24284 000077C0 4E                <1>       dec    esi
24285 000077C1 C60600            <1>       mov    byte [esi], 0
24286 000077C4 5E                <1>       pop    esi
24287 000077C5 89F7              <1>       mov    edi, esi
24288 000077C7 E891EBFFFF        <1>       call   print_msg
24289 000077CC C60700            <1>       mov    byte [edi], 0
24290                            <1> cmd_echo_nextline:
24291 000077CF BE[B8130100]      <1>       mov    esi, NextLine
24292                            <1>       ;call  print_msg
24293                            <1>       ;retn
24294 000077D4 E984EBFFFF        <1>       jmp    print_msg
24295                            <1>
24296                            <1> cmp_cmd_copy:
24297 000077D9 B104              <1>       mov    cl, 4
24298 000077DB BF[95070100]      <1>       mov    edi, Cmd_Copy
24299 000077E0 E837030000        <1>       call   cmp_cmd
24300 000077E5 0F8304180000      <1>       jnc    copy_file
24301                            <1>
24302                            <1> cmp_cmd_move:
24303 000077EB B104              <1>       mov    cl, 4
```

```
24304 000077ED BF[9A070100]       <1>       mov    edi, Cmd_Move
24305 000077F2 E825030000         <1>       call   cmp_cmd
24306 000077F7 0F8398160000       <1>       jnc    move_file
24307                             <1>
24308                             <1> cmp_cmd_path:
24309 000077FD B104               <1>       mov    cl, 4
24310 000077FF BF[9F070100]       <1>       mov    edi, Cmd_Path
24311 00007804 E813030000         <1>       call   cmp_cmd
24312 00007809 0F83381A0000       <1>       jnc    set_get_path
24313                             <1>
24314                             <1> cmp_cmd_beep:
24315 0000780F B104               <1>       mov    cl, 4
24316 00007811 BF[CC070100]       <1>       mov    edi, Cmd_Beep
24317 00007816 E801030000         <1>       call   cmp_cmd
24318 0000781B 720B               <1>       jc     short cmp_cmd_find
24319                             <1>       ; 13/05/2016
24320 0000781D 8A3D[4E520100]     <1>       mov    bh, [ptty] ; [ACTIVE_PAGE]
24321 00007823 E96AA5FFFF         <1>       jmp    beeper
24322                             <1>
24323                             <1> cmp_cmd_find:
24324 00007828 B104               <1>       mov    cl, 4
24325 0000782A BF[A9070100]       <1>       mov    edi, Cmd_Find
24326 0000782F E8E8020000         <1>       call   cmp_cmd
24327 00007834 0F82C5020000       <1>         jc     cmp_cmd_external
24328                             <1>
24329                             <1>       ;call  find_and_list_files
24330 0000783A E9F7220000         <1>       jmp    find_and_list_files
24331                             <1>       ;retn
24332                             <1>
24333                             <1> c_1:
24334 0000783F AD                 <1>       lodsd
24335                             <1> cmp_cmd_help:
24336 00007840 3C3F               <1>       cmp    al, '?'
24337 00007842 751D               <1>         jne    short cmp_cmd_remark
24338                             <1>
24339 00007844 BE[32070100]       <1>       mov    esi, Command_List
24340                             <1> cmd_help_next_w:
24341 00007849 E80FEBFFFF         <1>       call   print_msg
24342                             <1>
24343 0000784E 803E20             <1>       cmp    byte [esi], 20h ; 0
24344 00007851 7232               <1>       jb     short cmd_help_retn
24345                             <1>
24346 00007853 56                 <1>       push   esi
24347 00007854 BE[6F130100]       <1>       mov    esi, nextline
24348 00007859 E8FFEAFFFF         <1>       call   print_msg
24349 0000785E 5E                 <1>       pop    esi
24350 0000785F EBE8               <1>       jmp    short cmd_help_next_w
24351                             <1>
24352                             <1> cmp_cmd_remark:
24353 00007861 3C2A               <1>       cmp    al, '*'
24354 00007863 0F8596020000       <1>         jne    cmp_cmd_external
24355 00007869 46                 <1>       inc    esi
24356 0000786A BF[48530100]       <1>       mov    edi, Remark
24357 0000786F 8A06               <1>       mov    al, [esi]
24358 00007871 3C20               <1>       cmp    al, 20h
24359 00007873 7707               <1>       ja     short cmd_remark_write
24360 00007875 89FE               <1>       mov    esi, edi ; Remark
24361 00007877 E9E1EAFFFF         <1>       jmp    print_msg
24362                             <1>
24363                             <1> cmd_remark_write:
24364 0000787C AA                 <1>       stosb
24365 0000787D AC                 <1>       lodsb
24366 0000787E 3C20               <1>       cmp    al, 20h
24367 00007880 73FA               <1>       jnb    short cmd_remark_write
24368 00007882 C60700             <1>       mov    byte [edi], 0
24369                             <1>
24370                             <1> cmd_help_retn:
24371                             <1> cmd_remark_retn:
24372                             <1> cd_retn:
24373 00007885 C3                 <1>       retn
24374                             <1>
24375                             <1> c_2:
24376 00007886 80F902             <1>       cmp    cl, 2
24377 00007889 0F87B1000000       <1>         ja     c_3
24378 0000788F BE[96530100]       <1>       mov    esi, CommandBuffer
24379 00007894 72A9               <1>       jb     short c_1
24380                             <1>
24381                             <1> cmp_cmd_cd:
24382 00007896 66AD               <1>       lodsw
24383 00007898 663D4344           <1>       cmp    ax, 'CD'
24384 0000789C 7553               <1>       jne    short cmp_cmd_drive
24385 0000789E 46                 <1>         inc  esi
24386                             <1> cd_0:
24387 0000789F 668B06             <1>       mov    ax, [esi]
24388 000078A2 3C20               <1>       cmp    al, 20h
24389 000078A4 76DF               <1>       jna    short cd_retn
24390                             <1>       ; 10/02/2016
24391 000078A6 80FC3A             <1>       cmp    ah, ':'
24392 000078A9 7504               <1>       jne    short cd_1
24393 000078AB 46                 <1>       inc    esi
24394 000078AC 46                 <1>       inc    esi
24395 000078AD EB4B               <1>       jmp    short cd_2
24396                             <1>
24397                             <1> cd_1: ; change current directory
24398                             <1>       ; 29/11/2009
24399                             <1>       ; AH = CDh   ; to separate 'CD' command from others
24400                             <1>                    ; for restoring current directory
24401                             <1>                    ; 0CDh sign is for saving cdir into
24402                             <1>                    ; DOS drv description table cdir area
24403                             <1>
24404 000078AF B4CD               <1>       mov    ah, 0CDh ; mov byte [CD_COMMAND], 0CDh
24405                             <1>
24406 000078B1 E865230000         <1>       call   change_current_directory
```

```
24407 000078B6 0F837F220000        <1>         jnc     change_prompt_dir_string
24408                              <1>
24409                              <1> cd_error_messages:
24410 000078BC 3C03                <1>         cmp     al, 3
24411 000078BE 740C                <1>         je      short cd_path_not_found
24412                              <1>         ; 16/10/2016 (15h -> 15)
24413 000078C0 3C0F                <1>         cmp     al, 15 ; drive not ready error
24414 000078C2 745B                <1>         je      short cd_drive_not_ready
24415 000078C4 3C11                <1>         cmp     al, 17 ; read error
24416 000078C6 7457                <1>         je      short cd_drive_not_ready
24417 000078C8 3C13                <1>         cmp     al, 19 ; ; Bad directory/path name
24418 000078CA 7468                <1>         je      short cd_command_failed
24419                              <1>
24420                              <1> cd_path_not_found:
24421 000078CC 6650                <1>         push    ax
24422 000078CE BE[D5090100]        <1>         mov     esi, Msg_Dir_Not_Found
24423 000078D3 E885EAFFFF          <1>         call    print_msg
24424 000078D8 6658                <1>         pop     ax
24425 000078DA 3A25[E4520100]      <1>         cmp     ah, [Current_Dir_Level]
24426 000078E0 0F8355220000        <1>           jnb     change_prompt_dir_string
24427 000078E6 8825[E4520100]      <1>         mov     [Current_Dir_Level], ah
24428 000078EC E94A220000          <1>           jmp     change_prompt_dir_string
24429                              <1>
24430                              <1> cmp_cmd_drive: ; change current drive
24431                              <1>         ; C:, D:, E: etc.
24432 000078F1 80FC3A              <1>         cmp     ah, ':'
24433 000078F4 0F8505020000        <1>           jne     cmp_cmd_external
24434                              <1>
24435                              <1> cd_2: ; 'CD C:', 'CD D:' ...
24436 000078FA 803E20              <1>         cmp     byte [esi], 20h
24437 000078FD 0F8706020000        <1>           ja      loc_cmd_failed
24438                              <1>
24439 00007903 24DF                <1>         and     al, 0DFh
24440 00007905 2C41                <1>         sub     al, 'A'
24441 00007907 0F82FC010000        <1>           jc      loc_cmd_failed
24442                              <1>
24443 0000790D 3A05[D2060100]      <1>           cmp     al, [Last_DOS_DiskNo]
24444 00007913 770A                <1>           ja   short cd_drive_not_ready
24445                              <1>
24446 00007915 88C2                <1>         mov     dl, al
24447 00007917 E854F3FFFF          <1>         call    change_current_drive
24448 0000791C 7201                <1>         jc      short cd_drive_not_ready
24449 0000791E C3                  <1>         retn
24450                              <1>
24451                              <1> cd_drive_not_ready:
24452 0000791F BE[92090100]        <1>         mov     esi, Msg_Not_Ready_Read_Err
24453 00007924 E834EAFFFF          <1>         call    print_msg
24454                              <1>
24455                              <1> cd_fail_drive_restart:
24456 00007929 8A15[E6520100]      <1>         mov     dl, [Current_Drv]
24457                              <1>         ;call  change_current_drive
24458 0000792F E93CF3FFFF          <1>           jmp     change_current_drive
24459                              <1>         ;retn
24460                              <1>
24461                              <1> cd_command_failed:
24462 00007934 BE[73090100]        <1>         mov     esi, Msg_Bad_Command
24463 00007939 E81FEAFFFF          <1>         call    print_msg
24464 0000793E EBE9                <1>         jmp     short cd_fail_drive_restart
24465                              <1>
24466                              <1> c_3:
24467                              <1> cmp_cmd_dir:
24468 00007940 BF[32070100]        <1>         mov     edi, Cmd_Dir
24469 00007945 E8D2010000          <1>         call    cmp_cmd
24470 0000794A 0F8371020000        <1>         jnc     print_directory_list
24471                              <1>
24472                              <1> cmp_cmd_cls:
24473 00007950 B103                <1>         mov     cl, 3
24474 00007952 BF[6E070100]        <1>         mov     edi, Cmd_Cls
24475 00007957 E8C0010000          <1>         call    cmp_cmd
24476 0000795C 0F8311EAFFFF        <1>           jnc   clear_screen
24477                              <1>
24478                              <1> cmp_cmd_ver:
24479 00007962 B103                <1>         mov     cl, 3
24480 00007964 BF[3C070100]        <1>         mov     edi, Cmd_Ver
24481 00007969 E8AE010000          <1>         call    cmp_cmd
24482 0000796E 720A                <1>         jc      short cmp_cmd_mem
24483                              <1>
24484 00007970 BE[DA060100]        <1>         mov     esi, mainprog_Version
24485                              <1>         ;call  print_msg
24486 00007975 E9E3E9FFFF          <1>         jmp     print_msg
24487                              <1>         ;retn
24488                              <1>
24489                              <1> cmp_cmd_mem:
24490 0000797A B103                <1>         mov     cl, 3
24491 0000797C BF[A4070100]        <1>         mov     edi, Cmd_Mem
24492 00007981 E896010000          <1>         call    cmp_cmd
24493 00007986 0F83C8B8FFFF        <1>         jnc     memory_info
24494                              <1>
24495                              <1> cmp_cmd_del:
24496 0000798C B103                <1>         mov     cl, 3
24497 0000798E BF[77070100]        <1>         mov     edi, Cmd_Del
24498 00007993 E884010000          <1>         call    cmp_cmd
24499 00007998 0F832D0F0000        <1>           jnc     delete_file
24500                              <1>
24501                              <1> cmp_cmd_set:
24502 0000799E B103                <1>         mov     cl, 3
24503 000079A0 BF[6A070100]        <1>         mov     edi, Cmd_Set
24504 000079A5 E872010000          <1>         call    cmp_cmd
24505 000079AA 0F830F180000        <1>           jnc     set_get_env
24506                              <1>
24507                              <1> cmp_cmd_run:
24508 000079B0 B103                <1>         mov     cl, 3
24509 000079B2 BF[66070100]        <1>         mov     edi, Cmd_Run
```

```
24510 000079B7 E860010000          <1>       call   cmp_cmd
24511                              <1>       ; 07/05/2016
24512 000079BC 0F823D010000        <1>         jc      cmp_cmd_external
24513 000079C2 E9551E0000          <1>       jmp    load_and_execute_file
24514                              <1> c_5:
24515                              <1> cmp_cmd_mkdir:
24516 000079C7 BF[8F070100]        <1>       mov    edi, Cmd_Mkdir
24517 000079CC E84B010000          <1>       call   cmp_cmd
24518 000079D1 0F838C0A0000        <1>         jnc     make_directory
24519                              <1>
24520                              <1> cmp_cmd_rmdir:
24521 000079D7 B105                <1>       mov    cl, 5
24522 000079D9 BF[89070100]        <1>       mov    edi, Cmd_Rmdir
24523 000079DE E839010000          <1>       call   cmp_cmd
24524 000079E3 0F83990B0000        <1>         jnc     delete_directory
24525                              <1>
24526                              <1> cmp_cmd_chdir:
24527 000079E9 B105                <1>       mov    cl, 5
24528 000079EB BF[C6070100]        <1>       mov    edi, Cmd_Chdir
24529 000079F0 E827010000          <1>       call   cmp_cmd
24530 000079F5 0F8204010000        <1>         jc      cmp_cmd_external
24531                              <1>
24532 000079FB E99FFEFFFF          <1>       jmp    cd_0
24533                              <1>
24534                              <1> c_6:
24535 00007A00 80F906              <1>       cmp    cl, 6
24536 00007A03 0F87DF000000        <1>         ja      c_8
24537 00007A09 72BC                <1>       jb     short c_5
24538                              <1> cmp_cmd_prompt:
24539 00007A0B BF[45070100]        <1>       mov    edi, Cmd_Prompt
24540 00007A10 E807010000          <1>       call   cmp_cmd
24541 00007A15 722E                <1>         jc   short cmp_cmd_volume
24542                              <1> get_prompt_name_fchar:
24543 00007A17 AC                  <1>       lodsb
24544 00007A18 3C20                <1>       cmp    al, 20h
24545 00007A1A 74FB                <1>       je     short get_prompt_name_fchar
24546 00007A1C 7712                <1>       ja     short loc_change_prompt_label
24547 00007A1E BE[26070100]        <1>       mov    esi, TRDOSPromptLabel
24548 00007A23 C7065452444F        <1>       mov    dword [esi], "TRDO"
24549 00007A29 66C746045300        <1>              mov    word [esi+4], "S"
24550                              <1> loc_cmd_prompt_return:
24551 00007A2F C3                  <1>       retn
24552                              <1> loc_change_prompt_label:
24553 00007A30 66B90B00            <1>       mov    cx, 11
24554 00007A34 BF[26070100]        <1>       mov    edi, TRDOSPromptLabel
24555                              <1> put_char_new_prompt_label:
24556 00007A39 AA                  <1>       stosb
24557 00007A3A AC                  <1>       lodsb
24558 00007A3B 3C20                <1>       cmp    al, 20h
24559 00007A3D 7202                <1>       jb     short pass_put_new_prompt_label
24560 00007A3F E2F8                <1>       loop   put_char_new_prompt_label
24561                              <1> pass_put_new_prompt_label:
24562 00007A41 C60700              <1>       mov    byte [edi], 0
24563 00007A44 C3                  <1>       retn
24564                              <1>
24565                              <1> cmp_cmd_volume:
24566 00007A45 B106                <1>       mov    cl, 6
24567 00007A47 BF[4C070100]        <1>       mov    edi, Cmd_Volume
24568 00007A4C E8CB000000          <1>       call   cmp_cmd
24569 00007A51 7255                <1>         jc   short cmp_cmd_attrib
24570                              <1>
24571                              <1> cmd_vol1:
24572 00007A53 AC                  <1>       lodsb
24573 00007A54 3C20                <1>       cmp    al, 20h
24574 00007A56 7707                <1>       ja     short cmd_vol2
24575 00007A58 A0[E6520100]        <1>       mov    al, [Current_Drv]
24576 00007A5D EB3D                <1>       jmp    short cmd_vol4
24577                              <1> cmd_vol2:
24578 00007A5F 3C41                <1>       cmp    al, 'A'
24579 00007A61 0F82A2000000        <1>         jb      loc_cmd_failed
24580 00007A67 3C7A                <1>       cmp    al, 'z'
24581 00007A69 0F879A000000        <1>         ja      loc_cmd_failed
24582 00007A6F 3C5A                <1>       cmp    al, 'Z'
24583 00007A71 760A                <1>       jna    short cmd_vol3
24584 00007A73 3C61                <1>       cmp    al, 'a'
24585 00007A75 0F828E000000        <1>         jb      loc_cmd_failed
24586 00007A7B 24DF                <1>       and    al, 0DFh
24587                              <1> cmd_vol3:
24588 00007A7D 8A26                <1>       mov    ah, [esi]
24589 00007A7F 80FC3A              <1>       cmp    ah, ':'
24590 00007A82 0F8581000000        <1>         jne     loc_cmd_failed
24591 00007A88 2C41                <1>       sub    al, 'A'
24592 00007A8A 3A05[D2060100]      <1>         cmp     al, [Last_DOS_DiskNo]
24593 00007A90 760A                <1>       jna    short cmd_vol4
24594                              <1>
24595 00007A92 BE[92090100]        <1>       mov    esi, Msg_Not_Ready_Read_Err
24596 00007A97 E9C1E8FFFF          <1>       jmp    print_msg
24597                              <1>
24598                              <1> cmd_vol4:
24599 00007A9C E88DFAFFFF          <1>       call   print_volume_info
24600 00007AA1 0F8278FEFFFF        <1>         jc      cd_drive_not_ready
24601 00007AA7 C3                  <1>       retn
24602                              <1>
24603                              <1> cmp_cmd_attrib:
24604 00007AA8 B106                <1>       mov    cl, 6
24605 00007AAA BF[7B070100]        <1>       mov    edi, Cmd_Attrib
24606 00007AAF E868000000          <1>       call   cmp_cmd
24607 00007AB4 0F83310F0000        <1>         jnc     set_file_attributes
24608                              <1>
24609                              <1> cmp_cmd_rename:
24610 00007ABA B106                <1>       mov    cl, 6
24611 00007ABC BF[82070100]        <1>       mov    edi, Cmd_Rename
24612 00007AC1 E856000000          <1>       call   cmp_cmd
```

```
24613 00007AC6 0F8367110000        <1>         jnc     rename_file
24614                              <1>
24615                              <1> cmp_cmd_device:
24616 00007ACC B106                <1>         mov     cl, 6
24617 00007ACE BF[B7070100]        <1>         mov     edi, Cmd_Device
24618 00007AD3 E844000000          <1>         call    cmp_cmd
24619 00007AD8 7225                <1>           jc    short cmp_cmd_external
24620                              <1>
24621 00007ADA C3                  <1>         retn
24622                              <1>
24623                              <1> c_7:
24624                              <1> cmp_cmd_devlist:
24625 00007ADB BF[BE070100]        <1>         mov     edi, Cmd_DevList
24626 00007AE0 E837000000          <1>         call    cmp_cmd
24627 00007AE5 7218                <1>           jc    short cmp_cmd_external
24628                              <1>
24629                              <1> loc_cmd_return:
24630 00007AE7 C3                  <1>         retn
24631                              <1>
24632                              <1> c_8:
24633 00007AE8 80F908              <1>           cmp  cl, 8
24634 00007AEB 7712                <1>         ja      short cmp_cmd_external
24635 00007AED 72EC                <1>         jb      short c_7
24636                              <1>
24637                              <1> cmp_cmd_longname:
24638 00007AEF BF[53070100]        <1>         mov     edi, Cmd_LongName
24639 00007AF4 E823000000          <1>         call    cmp_cmd
24640 00007AF9 0F8342060000        <1>           jnc     get_and_print_longname
24641                              <1>
24642                              <1> cmp_cmd_external:
24643                              <1>         ; 07/05/2016
24644                              <1>         ; 22/04/2016
24645 00007AFF BE[96530100]        <1>         mov     esi, CommandBuffer
24646 00007B04 E9131D0000          <1>         jmp     loc_run_check_filename
24647                              <1>
24648                              <1> loc_cmd_failed:
24649 00007B09 803D[96530100]20    <1>         cmp     byte [CommandBuffer], 20h
24650 00007B10 76D5                <1>         jna     short loc_cmd_return
24651 00007B12 BE[73090100]        <1>         mov     esi, Msg_Bad_Command
24652                              <1> ;       call    print_msg
24653                              <1> ;loc_cmd_return:
24654                              <1> ;       retn
24655 00007B17 E941E8FFFF          <1>         jmp     print_msg
24656                              <1>
24657                              <1> cmp_cmd:
24658                              <1>          ; 29/01/2016 (TRDOS 386 = TRDOS v2.0)
24659 00007B1C BE[96530100]        <1>           mov esi, CommandBuffer
24660                              <1>           ; edi = internal command word (ASCIIZ)
24661                              <1>          ; ecx = command length (<=8)
24662                              <1> cmp_cmd_1:
24663 00007B21 AC                  <1>         lodsb
24664 00007B22 AE                  <1>         scasb
24665 00007B23 750D                <1>         jne     short cmp_cmd_3
24666 00007B25 E2FA                <1>         loop    cmp_cmd_1
24667 00007B27 AC                  <1>         lodsb
24668 00007B28 3C20                <1>         cmp     al, 20h
24669 00007B2A 7703                <1>         ja      short cmp_cmd_2
24670 00007B2C 30C0                <1>         xor     al, al
24671                              <1>         ; ZF = 1 -> internal command word matches
24672 00007B2E C3                  <1>         retn
24673                              <1> cmp_cmd_2:
24674                              <1>         ; ZF = 0 (CF = 0) -> external command word
24675 00007B2F 58                  <1>         pop     eax ; no return to the caller from here
24676 00007B30 EBCD                <1>         jmp     cmp_cmd_external
24677                              <1> cmp_cmd_3:
24678 00007B32 F9                  <1>         stc
24679                              <1>         ; CF = 1 -> internal command word does not match
24680 00007B33 C3                  <1>         retn
24681                              <1>
24682                              <1> loc_run_cmd_failed:
24683                              <1>         ; 15/03/2016
24684                              <1>         ; 15/02/2016 (TRDOS 386 =  TRDOS v2.0)
24685                              <1>         ; 07/12/2009 (CMD_INTR.ASM)
24686                              <1>         ; 29/11/2009
24687                              <1>
24688 00007B34 E855000000          <1>         call    restore_cdir_after_cmd_fail
24689                              <1>
24690                              <1> loc_run_cmd_failed_cmp_al:
24691                              <1>         ; End of Restore_CDIR code (29/11/2009)
24692                              <1>
24693 00007B39 3C01                <1>         cmp     al, 1 ; Bad command or file name
24694 00007B3B 74CC                <1>         je      loc_cmd_failed
24695                              <1> loc_run_dir_not_found:
24696 00007B3D 3C03                <1>         cmp     al, 3
24697 00007B3F 750A                <1>         jne     short loc_run_file_notfound_msg
24698                              <1>         ; Path not found (MS-DOS Error Code = 3)
24699 00007B41 BE[D5090100]        <1>         mov     esi, Msg_Dir_Not_Found
24700 00007B46 E912E8FFFF          <1>         jmp     print_msg
24701                              <1>
24702                              <1> loc_run_file_notfound_msg:
24703 00007B4B 3C02                <1>         cmp     al, 2 ; File not found
24704 00007B4D 750A                <1>         jne     short loc_run_file_drv_read_err
24705                              <1>
24706                              <1> loc_print_file_notfound_msg:
24707 00007B4F BE[EC090100]        <1>           mov     esi, Msg_File_Not_Found
24708                              <1>         ;call   proc_printmsg
24709                              <1>         ;retn
24710 00007B54 E904E8FFFF          <1>         jmp     print_msg
24711                              <1>
24712                              <1> loc_run_file_drv_read_err:
24713                              <1>         ; Err: 17 (Read fault)
24714 00007B59 3C11                <1>         cmp     al, 17 ; Drive not ready or read error
24715 00007B5B 7404                <1>         je      short loc_run_file_print_drv_read_err
```

```
24716                                  <1>       ;
24717 00007B5D 3C0F                    <1>       cmp    al, 15 ; Drive not ready (or read error)
24718 00007B5F 750A                    <1>       jne    short loc_run_file_toobig
24719                                  <1>
24720                                  <1> loc_run_file_print_drv_read_err:
24721 00007B61 BE[92090100]            <1>       mov    esi, Msg_Not_Ready_Read_Err
24722 00007B66 E9F2E7FFFF              <1>       jmp    print_msg
24723                                  <1>
24724                                  <1> loc_run_file_toobig:
24725 00007B6B 3C08                    <1>       cmp    al, 8 ; Not enough free memory to load&run file
24726 00007B6D 750A                    <1>       jne    short loc_run_misc_error
24727 00007B6F BE[370A0100]            <1>       mov    esi, Msg_Insufficient_Memory
24728 00007B74 E9E4E7FFFF              <1>       jmp    print_msg
24729                                  <1>
24730                                  <1>       ; 15/03/2016
24731                                  <1> print_misc_error_msg:
24732                                  <1> loc_run_misc_error:
24733                                  <1>       ; AL = Error code
24734 00007B79 E84BB7FFFF              <1>       call   bytetohex
24735 00007B7E 66A3[6B0A0100]          <1>         mov     [error_code_hex], ax
24736                                  <1>
24737 00007B84 BE[4E0A0100]            <1>       mov    esi, Msg_Error_Code
24738                                  <1>       ;call  print_msg
24739                                  <1>       ;retn
24740                                  <1>
24741 00007B89 E9CFE7FFFF              <1>       jmp    print_msg
24742                                  <1>
24743                                  <1> restore_cdir_after_cmd_fail:
24744                                  <1>       ; 15/02/2016 (TRDOS 386 =  TRDOS v2.0)
24745 00007B8E 50                      <1>       push   eax
24746 00007B8F 8A3D[465B0100]          <1>       mov    bh, [RUN_CDRV] ; it is set at the beginning
24747                                  <1>                              ; of the 'run' command.
24748 00007B95 3A3D[E6520100]          <1>       cmp    bh, [Current_Drv]
24749 00007B9B 7409                    <1>       je     short loc_run_restore_cdir
24750 00007B9D 88FA                    <1>       mov    dl, bh
24751 00007B9F E8CCF0FFFF              <1>       call   change_current_drive
24752 00007BA4 EB19                    <1>       jmp    short loc_run_err_pass_restore_cdir
24753                                  <1>
24754                                  <1> loc_run_restore_cdir:
24755 00007BA6 803D[D3060100]00        <1>       cmp    byte [Restore_CDIR], 0
24756 00007BAD 7610                    <1>       jna    short loc_run_err_pass_restore_cdir
24757 00007BAF 30DB                    <1>       xor    bl, bl
24758 00007BB1 0FB7F3                  <1>       movzx  esi, bx
24759 00007BB4 81C600010900            <1>       add    esi, Logical_DOSDisks
24760 00007BBA E868F1FFFF              <1>       call   restore_current_directory
24761                                  <1>
24762                                  <1> loc_run_err_pass_restore_cdir:
24763 00007BBF 58                      <1>       pop    eax
24764 00007BC0 C3                      <1>       retn
24765                                  <1>
24766                                  <1> print_directory_list:
24767                                  <1>       ; 10/02/2016
24768                                  <1>       ; 08/02/2016 (TRDOS 386 =  TRDOS v2.0)
24769                                  <1>       ; 06/12/2009 ('cmp_cmd_dir')
24770                                  <1>       ;
24771 00007BC1 66C705[885C0100]00-     <1>       mov    word [AttributesMask], 0800h ; ..except volume names..
24772 00007BC9 08                      <1>
24773 00007BCA A0[E6520100]            <1>       mov    al, [Current_Drv]
24774 00007BCF A2[465B0100]            <1>       mov    [RUN_CDRV], al
24775                                  <1> get_dfname_fchar:
24776 00007BD4 AC                      <1>       lodsb
24777 00007BD5 3C20                    <1>       cmp    al, 20h
24778 00007BD7 74FB                    <1>       je     short get_dfname_fchar
24779 00007BD9 0F82A4000000            <1>         jb      loc_print_dir_call_all
24780 00007BDF 3C2D                    <1>       cmp    al, '-'
24781 00007BE1 7542                    <1>       jne    short loc_print_dir_call_flt
24782                                  <1> get_next_attr_char:
24783 00007BE3 AC                      <1>       lodsb
24784 00007BE4 3C20                    <1>       cmp    al, 20h
24785 00007BE6 74FB                    <1>       je     short get_next_attr_char
24786 00007BE8 0F821BFFFFFF            <1>         jb      loc_cmd_failed
24787 00007BEE 24DF                    <1>       and    al, 0DFh
24788 00007BF0 3C44                    <1>       cmp    al, 'D' ; directories only ?
24789 00007BF2 7512                    <1>       jne    short pass_only_directories
24790 00007BF4 AC                      <1>       lodsb
24791 00007BF5 3C20                    <1>       cmp    al, 20h
24792 00007BF7 0F870CFFFFFF            <1>         ja      loc_cmd_failed
24793 00007BFD 800D[885C0100]10        <1>       or     byte [AttributesMask], 10h ; ..directory..
24794 00007C04 EB18                    <1>       jmp    short get_dfname_fchar_attr
24795                                  <1> pass_only_directories:
24796 00007C06 3C46                    <1>       cmp    al, 'F'      ; files only ?
24797 00007C08 0F85B0000000            <1>         jne     check_attr_s
24798 00007C0E AC                      <1>       lodsb
24799 00007C0F 3C20                    <1>       cmp    al, 20h
24800 00007C11 0F87F2FEFFFF            <1>         ja      loc_cmd_failed
24801 00007C17 800D[895C0100]10        <1>       or     byte [AttributesMask+1], 10h ; ..except directories..
24802                                  <1> get_dfname_fchar_attr:
24803 00007C1E AC                      <1>       lodsb
24804 00007C1F 3C20                    <1>       cmp    al, 20h
24805 00007C21 74FB                    <1>       je     short get_dfname_fchar_attr
24806 00007C23 725E                    <1>       jb     short loc_print_dir_call_all
24807                                  <1>
24808                                  <1> loc_print_dir_call_flt:
24809 00007C25 4E                      <1>       dec    esi
24810 00007C26 BF[8A5C0100]            <1>       mov    edi, FindFile_Drv
24811 00007C2B E801260000              <1>       call   parse_path_name
24812 00007C30 7308                    <1>       jnc    short loc_print_dir_change_drv_1
24813 00007C32 3C01                    <1>       cmp    al, 1
24814 00007C34 0F87FAFEFFFF            <1>         ja      loc_run_cmd_failed
24815                                  <1>
24816                                  <1> loc_print_dir_change_drv_1:
24817 00007C3A 8A15[8A5C0100]          <1>       mov    dl, [FindFile_Drv]
24818                                  <1> loc_print_dir_change_drv_2:
```

```
24819 00007C40 3A15[465B0100]      <1>       cmp    dl, [RUN_CDRV]
24820 00007C46 740B                <1>       je     short loc_print_dir_change_directory
24821 00007C48 E823F0FFFF          <1>       call   change_current_drive
24822 00007C4D 0F82E1FEFFFF        <1>         jc      loc_run_cmd_failed
24823                              <1> loc_print_dir_change_directory:
24824 00007C53 803D[8B5C0100]20    <1>       cmp    byte [FindFile_Directory], 20h ; 0 or 20h ?
24825 00007C5A 761D                <1>       jna    short pass_print_dir_change_directory
24826                              <1>
24827 00007C5C FE05[D3060100]      <1>       inc    byte [Restore_CDIR]
24828 00007C62 BE[8B5C0100]        <1>       mov    esi, FindFile_Directory
24829 00007C67 30E4                <1>       xor    ah, ah ; CD_COMMAND sign -> 0
24830 00007C69 E8AD1F0000          <1>       call   change_current_directory
24831 00007C6E 0F82C0FEFFFF        <1>         jc      loc_run_cmd_failed
24832                              <1>
24833                              <1> loc_print_dir_change_prompt_dir_string:
24834 00007C74 E8C21E0000          <1>       call   change_prompt_dir_string
24835                              <1>
24836                              <1> pass_print_dir_change_directory:
24837 00007C79 BE[CC5C0100]        <1>       mov    esi, FindFile_Name
24838 00007C7E 803E20              <1>       cmp    byte [esi], 20h ; ; 0 or 20h ?
24839 00007C81 7706                <1>       ja     short loc_print_dir_call
24840                              <1>
24841                              <1> loc_print_dir_call_all:
24842 00007C83 C7062A2E2A00        <1>       mov    dword [esi], '*.*'
24843                              <1> loc_print_dir_call:
24844 00007C89 E87E000000          <1>       call   print_directory
24845                              <1>
24846 00007C8E 8A15[465B0100]      <1>       mov    dl, [RUN_CDRV]  ; it is set at the beginning
24847 00007C94 3A15[E6520100]      <1>       cmp    dl, [Current_Drv]
24848 00007C9A 7406                <1>       je     short loc_print_dir_call_restore_cdir_retn
24849 00007C9C E8CFEFFFFF          <1>       call   change_current_drive
24850 00007CA1 C3                  <1>       retn
24851                              <1>
24852                              <1> loc_print_dir_call_restore_cdir_retn:
24853 00007CA2 803D[D3060100]00    <1>       cmp    byte [Restore_CDIR], 0
24854 00007CA9 7610                <1>       jna    short pass_print_dir_call_restore_cdir_retn
24855                              <1>
24856 00007CAB BE00010900          <1>       mov    esi, Logical_DOSDisks
24857 00007CB0 31C0                <1>       xor    eax, eax
24858 00007CB2 88D4                <1>       mov    ah, dl
24859 00007CB4 01C6                <1>       add    esi, eax
24860                              <1>
24861 00007CB6 E86CF0FFFF          <1>       call   restore_current_directory
24862                              <1>
24863                              <1> pass_print_dir_call_restore_cdir_retn:
24864 00007CBB C3                  <1>       retn
24865                              <1>
24866                              <1> check_attr_s_cap:
24867 00007CBC 24DF                <1>       and    al, 0DFh
24868                              <1> check_attr_s:
24869 00007CBE 3C53                <1>       cmp    al, 'S'
24870 00007CC0 7514                <1>       jne    short pass_attr_s
24871 00007CC2 800D[885C0100]04    <1>       or     byte [AttributesMask], 4 ; system
24872 00007CC9 AC                  <1>       lodsb
24873 00007CCA 3C20                <1>       cmp    al, 20h
24874 00007CCC 0F844CFFFFFF        <1>         je      get_dfname_fchar_attr
24875 00007CD2 72AF                <1>       jb     short loc_print_dir_call_all
24876 00007CD4 24DF                <1>       and    al, 0DFh
24877                              <1> pass_attr_s:
24878 00007CD6 3C48                <1>       cmp    al, 'H'
24879 00007CD8 7514                <1>       jne    short pass_attr_h
24880 00007CDA 800D[885C0100]02    <1>       or     byte [AttributesMask], 2 ; hidden
24881                              <1> pass_attr_shr:
24882 00007CE1 AC                  <1>       lodsb
24883 00007CE2 3C20                <1>       cmp    al, 20h
24884 00007CE4 0F8434FFFFFF        <1>         je      get_dfname_fchar_attr
24885 00007CEA 7297                <1>       jb     short loc_print_dir_call_all
24886 00007CEC EBCE                <1>       jmp    short check_attr_s_cap
24887                              <1>
24888                              <1> pass_attr_h:
24889 00007CEE 3C52                <1>       cmp    al, 'R'
24890 00007CF0 7509                <1>       jne    short pass_attr_r
24891 00007CF2 800D[885C0100]01    <1>       or     byte [AttributesMask], 1 ; read only
24892 00007CF9 EBE6                <1>       jmp    short pass_attr_shr
24893                              <1>
24894                              <1> pass_attr_r:
24895 00007CFB 3C41                <1>       cmp    al, 'A'
24896 00007CFD 0F8506FEFFFF        <1>         jne     loc_cmd_failed
24897 00007D03 800D[885C0100]20    <1>       or     byte [AttributesMask], 20h ; archive
24898 00007D0A EBD5                <1>       jmp    short pass_attr_shr
24899                              <1>
24900                              <1> print_directory:
24901                              <1>       ; 13/05/2016
24902                              <1>       ; 11/02/2016
24903                              <1>       ; 10/02/2016
24904                              <1>       ; 08/02/2016 (TRDOS 386 =  TRDOS v2.0)
24905                              <1>       ; 30/10/2010 ('proc_print_directory')
24906                              <1>       ; 19/09/2009
24907                              <1>       ; 2005
24908                              <1>       ; INPUT ->
24909                              <1>       ;      ESI = Asciiz File/Dir Name Address
24910                              <1>
24911 00007D0C 56                  <1>       push   esi
24912                              <1>
24913 00007D0D 29C0                <1>       sub    eax, eax
24914                              <1>
24915 00007D0F 66A3[145D0100]      <1>       mov    word [Dir_Count], ax ; 0
24916 00007D15 66A3[125D0100]      <1>       mov    word [File_Count], ax ; 0
24917 00007D1B A3[165D0100]        <1>       mov    dword [Total_FSize], eax ; 0
24918                              <1>
24919 00007D20 E84EE6FFFF          <1>       call    clear_screen
24920                              <1>
24921 00007D25 31C9                <1>       xor    ecx, ecx
```

```
24922 00007D27 8A2D[E6520100]      <1>       mov     ch, [Current_Drv] ; DirBuff_Drv - 'A'
24923 00007D2D A0[E7520100]        <1>       mov     al, [Current_Dir_Drv]
24924 00007D32 A2[90080100]        <1>       mov     [Dir_Drive_Name], al
24925 00007D37 BE00010900          <1>       mov    esi, Logical_DOSDisks
24926 00007D3C 01CE                <1>       add    esi, ecx
24927                              <1>
24928 00007D3E E865F9FFFF          <1>       call   move_volume_name_and_serial_no
24929 00007D43 730C                <1>       jnc    short print_dir_strlen_check
24930                              <1>
24931 00007D45 5E                  <1>       pop    esi
24932 00007D46 8A3D[4E520100]      <1>       mov    bh, [ptty] ; [ACTIVE_PAGE]
24933                              <1>       ;call  beeper
24934                              <1>       ;retn
24935 00007D4C E941A0FFFF          <1>       jmp    beeper  ; beep ! and return
24936                              <1>
24937                              <1> print_dir_strlen_check:
24938 00007D51 BE[E9520100]        <1>       mov    esi, Current_Dir_Root
24939 00007D56 BF[2D090100]        <1>       mov    edi, Dir_Str_Root
24940                              <1>
24941                              <1>       ;xor   ecx, ecx
24942 00007D5B 8A0D[45530100]      <1>         mov     cl, [Current_Dir_StrLen]
24943 00007D61 FEC1                <1>       inc    cl
24944 00007D63 80F940              <1>       cmp    cl, 64
24945 00007D66 760D                <1>       jna    short pass_print_dir_strlen_shorting
24946 00007D68 46                  <1>       inc    esi
24947 00007D69 01CE                <1>       add    esi, ecx
24948 00007D6B 83EE40              <1>       sub    esi, 64
24949 00007D6E 47                  <1>       inc    edi
24950 00007D6F B82E2E2E20          <1>       mov    eax, '... '
24951 00007D74 AB                  <1>       stosd
24952                              <1>
24953                              <1> pass_print_dir_strlen_shorting:
24954 00007D75 F3A4                <1>       rep    movsb
24955                              <1>
24956 00007D77 BE[83080100]        <1>       mov    esi, Dir_Drive_Str
24957 00007D7C E8DCE5FFFF          <1>       call   print_msg
24958                              <1>
24959 00007D81 BE[E2080100]        <1>       mov    esi, Vol_Serial_Header
24960 00007D86 E8D2E5FFFF          <1>       call   print_msg
24961                              <1>
24962 00007D8B BE[22090100]        <1>       mov    esi, Dir_Str_Header
24963 00007D90 E8C8E5FFFF          <1>       call   print_msg
24964                              <1>
24965 00007D95 BE[6D130100]        <1>       mov    esi, next2line
24966 00007D9A E8BEE5FFFF          <1>       call   print_msg
24967                              <1>
24968                              <1> loc_print_dir_first_file:
24969 00007D9F C605[295D0100]10    <1>       mov    byte [PrintDir_RowCounter], 16
24970 00007DA6 66A1[885C0100]      <1>       mov    ax, [AttributesMask]
24971 00007DAC 5E                  <1>       pop    esi
24972                              <1>
24973 00007DAD E859020000          <1>       call   find_first_file
24974 00007DB2 0F826F010000        <1>         jc      loc_dir_ok
24975                              <1>
24976                              <1> loc_dfname_use_this:
24977                              <1>       ; bl = File Attributes (bh = Long Name Entry Length)
24978 00007DB8 F6C310              <1>       test   bl, 10h  ; Is it a directory?
24979 00007DBB 741B                <1>       jz     short loc_not_dir
24980                              <1>
24981 00007DBD 66FF05[145D0100]    <1>       inc    word [Dir_Count]
24982 00007DC4 89F2                <1>       mov    edx, esi     ; FindFile_DirEntry address
24983 00007DC6 BE[720A0100]        <1>       mov    esi, Type_Dir; '<DIR>     '
24984 00007DCB BF[890A0100]        <1>       mov    edi, Dir_Or_FileSize
24985                              <1>       ; move 10 bytes
24986 00007DD0 A5                  <1>       movsd
24987 00007DD1 A5                  <1>       movsd
24988 00007DD2 66A5                <1>       movsw
24989 00007DD4 89D6                <1>       mov    esi, edx
24990 00007DD6 EB36                <1>       jmp     short loc_dir_attribute
24991                              <1>
24992                              <1> loc_not_dir:
24993 00007DD8 66FF05[125D0100]    <1>       inc    word [File_Count]
24994 00007DDF 0105[165D0100]      <1>       add    [Total_FSize], eax
24995                              <1>
24996 00007DE5 B90A000000          <1>       mov    ecx, 10  ; 32 bit divisor
24997 00007DEA 89CF                <1>       mov    edi, ecx
24998 00007DEC 81C7[890A0100]      <1>       add    edi, Dir_Or_FileSize
24999                              <1> loc_dir_rdivide:
25000 00007DF2 29D2                <1>       sub    edx, edx
25001 00007DF4 F7F1                <1>       div    ecx     ; remainder in dl (< 10)
25002 00007DF6 80C230              <1>       add     dl, '0'      ; to make visible (ascii)
25003 00007DF9 4F                  <1>       dec    edi
25004 00007DFA 8817                <1>       mov     [edi], dl
25005 00007DFC 21C0                <1>       and    eax, eax
25006 00007DFE 75F2                <1>       jnz    short loc_dir_rdivide
25007                              <1>
25008                              <1> loc_dir_fill_space:
25009 00007E00 81FF[890A0100]      <1>       cmp     edi, Dir_Or_FileSize
25010 00007E06 7606                <1>       jna     short loc_dir_attribute
25011 00007E08 4F                  <1>       dec     edi
25012 00007E09 C60720              <1>       mov     byte [edi], 20h
25013 00007E0C EBF2                <1>       jmp     short loc_dir_fill_space
25014                              <1>
25015                              <1> loc_dir_attribute:
25016 00007E0E C705[940A0100]2020- <1>       mov    dword [File_Attribute], 20202020h
25017 00007E16 2020                <1>
25018                              <1>
25019 00007E18 80FB20              <1>       cmp    bl, 20h  ; Is it an archive file?
25020 00007E1B 7207                <1>       jb     short loc_dir_pass_arch
25021 00007E1D C605[970A0100]41    <1>       mov    byte [File_Attribute+3], 'A'
25022                              <1>
25023                              <1> loc_dir_pass_arch:
25024 00007E24 80E307              <1>       and    bl, 7
```

```
25025 00007E27 7428                <1>       jz     short loc_dir_file_name
25026 00007E29 88DF                <1>       mov    bh, bl
25027 00007E2B 80E303              <1>       and    bl, 3
25028 00007E2E 38DF                <1>       cmp    bh, bl
25029 00007E30 7607                <1>       jna    short loc_dir_pass_s
25030 00007E32 C605[940A0100]53    <1>       mov    byte [File_Attribute], 'S'
25031                              <1>
25032                              <1> loc_dir_pass_s:
25033 00007E39 80E302              <1>       and     bl,2
25034 00007E3C 7407                <1>       jz      short loc_dir_pass_h
25035 00007E3E C605[950A0100]48    <1>       mov     byte [File_Attribute+1], 'H'
25036                              <1> loc_dir_pass_h:
25037 00007E45 80E701              <1>       and     bh,1
25038 00007E48 7407                <1>       jz      short loc_dir_file_name
25039 00007E4A C605[960A0100]52    <1>       mov     byte [File_Attribute+2], 'R'
25040                              <1> loc_dir_file_name:
25041                              <1>       ;mov     bx, [esi+18h] ; Date
25042                              <1>       ;mov     dx, [esi+16h] ; Time
25043 00007E51 8B5E16              <1>       mov    ebx, [esi+16h]
25044 00007E54 89F1                <1>       mov    ecx, esi ; FindFile_DirEntry address
25045 00007E56 BF[7C0A0100]        <1>       mov     edi, File_Name
25046                              <1>       ; move 8 bytes
25047 00007E5B A5                  <1>       movsd
25048 00007E5C A5                  <1>       movsd
25049 00007E5D C60720              <1>       mov    byte [edi], 20h
25050 00007E60 47                  <1>       inc    edi
25051                              <1>       ; move 3 bytes
25052 00007E61 66A5                <1>       movsw
25053 00007E63 A4                  <1>       movsb
25054 00007E64 89CE                <1>       mov    esi, ecx
25055                              <1>
25056                              <1> Dir_Time_start:
25057                              <1>       ;mov   ax, dx       ; Time
25058 00007E66 6689D8              <1>       mov    ax, bx
25059 00007E69 66C1E805            <1>       shr    ax, 5        ; shift right 5 times
25060 00007E6D 6683E03F            <1>       and    ax, 0000111111b     ; Minute Mask
25061 00007E71 D40A                <1>       aam                 ; Q([AL]/10)->AH
25062                              <1>                           ; R([AL]/10)->AL
25063                              <1>                           ; [AL]+[AH]= Minute as BCD
25064 00007E73 660D3030            <1>       or     ax, '00'     ; Convert to ASCII
25065 00007E77 86E0                <1>       xchg   ah, al
25066 00007E79 66A3[A70A0100]      <1>       mov    [File_Minute], ax
25067                              <1>
25068                              <1>       ;mov   al, dh
25069 00007E7F 88F8                <1>       mov    al, bh
25070 00007E81 C0E803              <1>       shr    al, 3        ; shift right 3 times
25071 00007E84 D40A                <1>       aam                 ; [AL]+[AH]= Hours as BCD
25072 00007E86 660D3030            <1>       or     ax, '00'
25073 00007E8A 86E0                <1>       xchg   ah, al
25074 00007E8C 66A3[A40A0100]      <1>       mov     [File_Hour], ax
25075                              <1>
25076 00007E92 C1EB10              <1>       shr    ebx, 16             ; BX = Date
25077                              <1>
25078                              <1> Dir_Date_start:
25079 00007E95 6689D8              <1>       mov    ax, bx       ; Date
25080 00007E98 6683E01F            <1>       and    ax, 00011111b; Day Mask
25081 00007E9C D40A                <1>       aam                 ; Q([AL]/10)->AH
25082                              <1>                           ; R([AL]/10)->AL
25083                              <1>                           ; [AL]+[AH]= Day as BCD
25084 00007E9E 660D3030            <1>       or     ax, '00'     ; Convert to ASCII
25085 00007EA2 86C4                <1>       xchg   al, ah
25086                              <1>
25087 00007EA4 66A3[990A0100]      <1>       mov    [File_Day], ax
25088                              <1>
25089 00007EAA 6689D8              <1>       mov    ax, bx
25090 00007EAD 66C1E805            <1>       shr    ax, 5        ; shift right 5 times
25091 00007EB1 6683E00F            <1>       and    ax, 00001111b; Month Mask
25092 00007EB5 D40A                <1>       aam
25093 00007EB7 660D3030            <1>       or     ax, '00'
25094 00007EBB 86E0                <1>       xchg   ah, al
25095 00007EBD 66A3[9C0A0100]      <1>       mov    [File_Month], ax
25096                              <1>
25097 00007EC3 6689D8              <1>       mov    ax, bx
25098 00007EC6 66C1E809            <1>       shr     ax, 9
25099 00007ECA 6683E07F            <1>       and    ax, 01111111b; Result = Year - 1980
25100 00007ECE 6605BC07            <1>       add    ax, 1980
25101                              <1>
25102 00007ED2 B10A                <1>       mov    cl, 10
25103 00007ED4 F6F1                <1>       div    cl           ; Q -> AL, R -> AH
25104 00007ED6 80CC30              <1>       or     ah, '0'
25105 00007ED9 8825[A20A0100]      <1>       mov    [File_Year+3], ah
25106 00007EDF D40A                <1>       aam
25107 00007EE1 86E0                <1>       xchg   ah, al
25108 00007EE3 80CC30              <1>       or     ah, '0'        ; Convert to ASCII
25109 00007EE6 8825[A10A0100]      <1>       mov    [File_Year+2], ah
25110 00007EEC D40A                <1>       aam
25111 00007EEE 86C4                <1>       xchg   al, ah
25112 00007EF0 660D3030            <1>       or     ax, '00'
25113 00007EF4 66A3[9F0A0100]      <1>       mov    [File_Year], ax
25114                              <1>
25115                              <1> loc_show_line:
25116 00007EFA 56                  <1>       push   esi
25117 00007EFB BE[7C0A0100]        <1>       mov     esi, File_Name
25118 00007F00 E858E4FFFF          <1>       call   print_msg
25119 00007F05 BE[6F130100]        <1>       mov    esi, nextline
25120 00007F0A E84EE4FFFF          <1>       call   print_msg
25121 00007F0F 5E                  <1>       pop    esi
25122                              <1>
25123 00007F10 FE0D[295D0100]      <1>       dec    byte [PrintDir_RowCounter]
25124 00007F16 0F84D4000000        <1>         jz      pause_dir_scroll
25125                              <1>
25126                              <1> loc_next_entry:
25127 00007F1C E899010000          <1>       call   find_next_file
```

```
25128 00007F21 0F8391FEFFFF        <1>         jnc     loc_dfname_use_this
25129                              <1>
25130                              <1> loc_dir_ok:
25131 00007F27 B90A000000          <1>         mov     ecx, 10
25132 00007F2C 66A1[145D0100]      <1>         mov     ax, [Dir_Count]
25133 00007F32 BF[BD0A0100]        <1>         mov     edi, Decimal_Dir_Count
25134 00007F37 6639C8              <1>         cmp     ax, cx ; 10
25135 00007F3A 7216                <1>         jb      short pass_ddc
25136 00007F3C 47                  <1>         inc     edi
25137 00007F3D 6683F864            <1>         cmp     ax, 100
25138 00007F41 720F                <1>         jb      short pass_ddc
25139 00007F43 47                  <1>         inc     edi
25140 00007F44 663DE803            <1>         cmp     ax, 1000
25141 00007F48 7208                <1>         jb      short pass_ddc
25142 00007F4A 47                  <1>         inc     edi
25143 00007F4B 663D1027            <1>         cmp     ax, 10000
25144 00007F4F 7201                <1>         jb      short pass_ddc
25145 00007F51 47                  <1>         inc     edi
25146                              <1> pass_ddc:
25147 00007F52 886F01              <1>         mov     [edi+1], ch ; 0
25148                              <1> loc_ddc_rediv:
25149 00007F55 31D2                <1>         xor     edx, edx
25150 00007F57 66F7F1              <1>         div     cx   ; 10
25151 00007F5A 80C230              <1>         add     dl, '0'
25152 00007F5D 8817                <1>         mov     [edi], dl
25153 00007F5F 4F                  <1>         dec     edi
25154 00007F60 6609C0              <1>         or      ax, ax
25155 00007F63 75F0                <1>         jnz     short loc_ddc_rediv
25156                              <1>
25157 00007F65 66A1[125D0100]      <1>         mov     ax, [File_Count]
25158 00007F6B BF[AC0A0100]        <1>         mov     edi, Decimal_File_Count
25159 00007F70 6639C8              <1>         cmp     ax, cx ; 10
25160 00007F73 7216                <1>         jb      short pass_dfc
25161 00007F75 47                  <1>         inc     edi
25162 00007F76 6683F864            <1>         cmp     ax, 100
25163 00007F7A 720F                <1>         jb      short pass_dfc
25164 00007F7C 47                  <1>         inc     edi
25165 00007F7D 663DE803            <1>         cmp     ax, 1000
25166 00007F81 7208                <1>         jb      short pass_dfc
25167 00007F83 47                  <1>         inc     edi
25168 00007F84 663D1027            <1>         cmp     ax, 10000
25169 00007F88 7201                <1>         jb      short pass_dfc
25170 00007F8A 47                  <1>         inc     edi
25171                              <1> pass_dfc:
25172                              <1>         ;mov    cx, 10
25173 00007F8B 886F01              <1>         mov     [edi+1], ch ; 00
25174                              <1> loc_dfc_rediv:
25175                              <1>         ;xor   dx, dx
25176 00007F8E 30D2                <1>         xor     dl, dl
25177 00007F90 66F7F1              <1>         div     cx
25178 00007F93 80C230              <1>         add     dl, '0'
25179 00007F96 8817                <1>         mov     [edi], dl
25180 00007F98 4F                  <1>         dec     edi
25181 00007F99 6609C0              <1>         or      ax, ax
25182 00007F9C 75F0                <1>         jnz     short loc_dfc_rediv
25183                              <1>
25184 00007F9E BF[285D0100]        <1>         mov     edi, TFS_Dec_End
25185                              <1>           ;mov    byte [edi], 0
25186 00007FA3 A1[165D0100]        <1>         mov     eax, [Total_FSize]
25187                              <1>         ;mov    ecx, 10
25188                              <1> rediv_tfs_hex:
25189                              <1>         ;sub   edx, edx
25190 00007FA8 28D2                <1>         sub     dl, dl
25191 00007FAA F7F1                <1>         div     ecx
25192 00007FAC 80C230              <1>         add     dl, '0'
25193 00007FAF 4F                  <1>         dec     edi
25194 00007FB0 8817                <1>         mov     [edi], dl
25195 00007FB2 21C0                <1>         and     eax, eax
25196 00007FB4 75F2                <1>         jnz     short rediv_tfs_hex
25197                              <1>
25198 00007FB6 893D[1A5D0100]      <1>         mov     [TFS_Dec_Begin], edi
25199 00007FBC BE[AA0A0100]        <1>         mov     esi, Decimal_File_Count_Header
25200 00007FC1 E897E3FFFF          <1>         call    print_msg
25201 00007FC6 BE[B20A0100]        <1>         mov     esi, str_files
25202 00007FCB E88DE3FFFF          <1>         call    print_msg
25203 00007FD0 BE[C30A0100]        <1>         mov     esi, str_dirs
25204 00007FD5 E883E3FFFF          <1>         call    print_msg
25205 00007FDA 8B35[1A5D0100]      <1>         mov     esi, [TFS_Dec_Begin]
25206 00007FE0 E878E3FFFF          <1>         call    print_msg
25207 00007FE5 BE[D40A0100]        <1>         mov     esi, str_bytes
25208 00007FEA E86EE3FFFF          <1>         call    print_msg
25209                              <1>
25210 00007FEF C3                  <1>         retn
25211                              <1>
25212                              <1> pause_dir_scroll:
25213 00007FF0 28E4                <1>         sub     ah, ah
25214 00007FF2 E81F8CFFFF          <1>         call    int16h
25215 00007FF7 3C1B                <1>         cmp     al, 1Bh
25216 00007FF9 0F8428FFFFFF        <1>           je      loc_dir_ok
25217 00007FFF C605[295D0100]10    <1>         mov     byte [PrintDir_RowCounter], 16 ; Reset counter
25218 00008006 E911FFFFFF          <1>           jmp     loc_next_entry
25219                              <1>
25220                              <1> find_first_file:
25221                              <1>         ; 11/02/2016
25222                              <1>         ; 10/02/2016
25223                              <1>         ; 08/02/2016 (TRDOS 386 =  TRDOS v2.0)
25224                              <1>         ; 09/10/2011
25225                              <1>         ; 17/09/2009
25226                              <1>         ; 2005
25227                              <1>         ; INPUT ->
25228                              <1>         ;      ESI = ASCIIZ File/Dir Name Address (in Current Directory)
25229                              <1>         ;      AL = Attributes AND mask (The AND result must be equal to AL)
25230                              <1>         ;            bit 0 = Read Only
```

```
25231                                  <1>       ;           bir 1 = Hidden
25232                                  <1>       ;           bit 2 = System
25233                                  <1>       ;           bit 3 = Volume Label
25234                                  <1>       ;           bit 4 = Directory
25235                                  <1>       ;           bit 5 = Archive
25236                                  <1>       ;           bit 6 = Reserved, must be 0
25237                                  <1>       ;           bit 7 = Reserved, must be 0
25238                                  <1>       ;      AH = Attributes Negative AND mask (The AND result must be ZERO)
25239                                  <1>       ;
25240                                  <1>       ; OUTPUT ->
25241                                  <1>       ;     CF = 1 -> Error, Error Code in EAX (AL)
25242                                  <1>       ;     CF = 0 ->
25243                                  <1>       ;          ESI = Directory Entry (FindFile_DirEntry) Location
25244                                  <1>       ;          EDI = Directory Buffer Directory Entry Location
25245                                  <1>       ;          EAX = File Size
25246                                  <1>       ;           BL = Attributes of The File/Directory
25247                                  <1>       ;           BH = Long Name Yes/No Status (>0 is YES)
25248                                  <1>       ;             DX > 0 : Ambiguous filename chars are used
25249                                  <1>       ;
25250                                  <1>       ; (EAX, EBX, ECX, EDX, ESI, EDI will be changed)
25251                                  <1>
25252 0000800B 66A3[DA5C0100]          <1>       mov   [FindFile_AttributesMask], ax
25253 00008011 BF[DC5C0100]            <1>       mov   edi, FindFile_DirEntry ; TR-DOS Fullfilename formatted buffer
25254 00008016 31C0                    <1>       xor   eax, eax
25255 00008018 B90B000000              <1>       mov   ecx, 11
25256 0000801D F3AB                    <1>       rep   stosd  ; 44 bytes
25257                                  <1>       ;stosw       ; +2 bytes
25258                                  <1>
25259 0000801F BF[CC5C0100]            <1>       mov   edi, FindFile_Name ; FFF structure, offset 66
25260 00008024 39FE                    <1>       cmp   esi, edi
25261 00008026 7408                    <1>       je    short loc_fff_mfn_ok
25262 00008028 89FA                    <1>       mov   edx, edi
25263                                  <1>       ; move 13 bytes
25264 0000802A A5                      <1>       movsd
25265 0000802B A5                      <1>       movsd
25266 0000802C A5                      <1>       movsd
25267 0000802D AA                      <1>       stosb
25268 0000802E 89D6                    <1>       mov   esi, edx
25269                                  <1> loc_fff_mfn_ok:
25270 00008030 BF[7B5C0100]            <1>       mov   edi, Dir_Entry_Name ; Dir Entry Format File Name
25271 00008035 E82C210000              <1>       call  convert_file_name
25272 0000803A 89FE                    <1>       mov   esi, edi ; offset Dir_Entry_Name
25273                                  <1>
25274 0000803C 66A1[DA5C0100]          <1>       mov   ax, [FindFile_AttributesMask]
25275                                  <1>       ;xor  ecx, ecx
25276 00008042 30C9                    <1>       xor   cl, cl
25277 00008044 E8261E0000              <1>       call  locate_current_dir_file
25278 00008049 726E                    <1>       jc    short loc_fff_retn
25279                                  <1>       ; EDI = Directory Entry
25280                                  <1>       ; EBX = Directory Buffer Entry Index/Number
25281                                  <1>
25282                                  <1> loc_fff_fnf_ln_check:
25283 0000804B 30ED                    <1>       xor   ch, ch
25284 0000804D 80F60F                  <1>       xor   dh, 0Fh
25285 00008050 7408                    <1>       jz    short loc_fff_longname_yes
25286 00008052 882D[D95C0100]          <1>       mov   [FindFile_LongNameYes], ch ; 0
25287 00008058 EB0C                    <1>       jmp   short loc_fff_longname_no
25288                                  <1>
25289                                  <1> loc_fff_longname_yes:
25290                                  <1>       ;inc  byte [FindFile_LongNameYes]
25291 0000805A 8A0D[E65B0100]          <1>       mov   cl, [LFN_EntryLength]
25292 00008060 880D[D95C0100]          <1>       mov   [FindFile_LongNameEntryLength], cl ; FindFile_LongNameYes
25293                                  <1>
25294                                  <1> loc_fff_longname_no:
25295                                  <1>       ;mov  bx, [DirBuff_CurrentEntry]
25296 00008066 66891D[045D0100]        <1>       mov   [FindFile_DirEntryNumber], bx
25297 0000806D 6689C2                  <1>       mov   dx, ax ; Ambiguous Filename chars used sign > 0
25298                                  <1>
25299 00008070 A0[E6520100]            <1>       mov   al, [Current_Drv]
25300 00008075 A2[8A5C0100]            <1>       mov   [FindFile_Drv], al
25301                                  <1>
25302 0000807A A1[E0520100]            <1>       mov   eax, [Current_Dir_FCluster]
25303 0000807F A3[FC5C0100]            <1>       mov   [FindFile_DirFirstCluster], eax
25304                                  <1>
25305 00008084 A1[155B0100]            <1>       mov   eax, [DirBuff_Cluster]
25306 00008089 A3[005D0100]            <1>       mov   [FindFile_DirCluster], eax
25307                                  <1>
25308 0000808E 66FF05[065D0100]        <1>       inc   word [FindFile_MatchCounter]
25309                                  <1>
25310 00008095 89FB                    <1>       mov   ebx, edi
25311 00008097 89FE                    <1>       mov   esi, edi
25312 00008099 BF[DC5C0100]            <1>       mov   edi, FindFile_DirEntry
25313 0000809E 89F8                    <1>       mov   eax, edi
25314 000080A0 B108                    <1>       mov   cl, 8
25315 000080A2 F3A5                    <1>       rep   movsd
25316 000080A4 89C6                    <1>       mov   esi, eax
25317 000080A6 89DF                    <1>       mov   edi, ebx
25318                                  <1>
25319 000080A8 A1[F85C0100]            <1>       mov   eax, [FindFile_DirEntry+28] ; File Size
25320                                  <1>
25321 000080AD 8A1D[E75C0100]          <1>       mov   bl, [FindFile_DirEntry+11] ; File Attributes
25322 000080B3 8A3D[D95C0100]          <1>       mov   bh, [FindFile_LongNameYes]
25323                                  <1>
25324                                  <1>       ;mov  cx, [DirBuff_EntryCounter]
25325                                  <1>       ;mov  [FindFile_DirEntryNumber], cx
25326                                  <1>       ;mov  cx, [FindFile_DirEntryNumber]
25327                                  <1>       ; ecx = 0
25328                                  <1>
25329                                  <1> loc_fff_retn:
25330 000080B9 C3                      <1>       retn
25331                                  <1>
25332                                  <1> find_next_file:
25333                                  <1>       ; 15/10/2016
```

```
25334                                  <1>       ; 10/02/2016
25335                                  <1>       ; 08/02/2016 (TRDOS 386 =  TRDOS v2.0)
25336                                  <1>       ; 06/02/2011
25337                                  <1>       ; 17/09/2009
25338                                  <1>       ; 2005
25339                                  <1>       ; INPUT ->
25340                                  <1>       ;     NONE, Find First File Parameters
25341                                  <1>       ; OUTPUT ->
25342                                  <1>       ;     CF = 1 -> Error, Error Code in EAX (AL)
25343                                  <1>       ;     CF = 0 ->
25344                                  <1>       ;         ESI = Directory Entry (FindFile_DirEntry) Location
25345                                  <1>       ;         EDI = Directory Buffer Directory Entry Location
25346                                  <1>       ;         EAX = File Size
25347                                  <1>       ;           BL = Attributes of The File/Directory
25348                                  <1>       ;           BH = Long Name Yes/No Status (>0 is YES)
25349                                  <1>       ;             DX > 0 : Ambiguous filename chars are used
25350                                  <1>       ;
25351                                  <1>       ; (EAX, EBX, ECX, EDX, ESI, EDI will be changed)
25352                                  <1>
25353 000080BA 66833D[065D0100]00      <1>       cmp    word [FindFile_MatchCounter], 0
25354 000080C2 7707                    <1>       ja     short loc_start_search_next_file
25355                                  <1>
25356                                  <1> loc_fnf_stc_retn:
25357 000080C4 F9                      <1>       stc
25358                                  <1> loc_fnf_ax12h_retn:
25359 000080C5 B80C000000              <1>       mov    eax, 12 ; No More files
25360                                  <1> ;loc_fnf_retn:
25361 000080CA C3                      <1>       retn
25362                                  <1>
25363                                  <1> loc_start_search_next_file:
25364 000080CB 668B1D[045D0100]        <1>       mov    bx, [FindFile_DirEntryNumber]
25365 000080D2 6643                    <1>       inc    bx
25366 000080D4 663B1D[135B0100]        <1>       cmp    bx, [DirBuff_LastEntry]
25367 000080DB 7719                    <1>       ja     short loc_cont_search_next_file
25368                                  <1>
25369                                  <1> loc_fnf_search:
25370 000080DD BE[7B5C0100]            <1>       mov    esi, Dir_Entry_Name
25371 000080E2 66A1[DA5C0100]          <1>       mov    ax, [FindFile_AttributesMask]
25372 000080E8 6631C9                  <1>       xor    cx, cx
25373 000080EB E8831E0000              <1>       call   find_directory_entry
25374 000080F0 0F8355FFFFFF            <1>         jnc     loc_fff_fnf_ln_check
25375                                  <1>
25376                                  <1> loc_cont_search_next_file:
25377 000080F6 31DB                    <1>       xor    ebx, ebx
25378 000080F8 8A3D[E6520100]          <1>       mov    bh, [Current_Drv]
25379 000080FE BE00010900              <1>       mov    esi, Logical_DOSDisks
25380 00008103 01DE                    <1>       add    esi, ebx
25381                                  <1>
25382 00008105 803D[E4520100]00        <1>       cmp    byte [Current_Dir_Level], 0
25383 0000810C 7608                    <1>       jna    short loc_fnf_check_FAT_type
25384 0000810E 807E0301                <1>       cmp    byte [esi+LD_FATType], 1
25385 00008112 72B1                    <1>       jb     short loc_fnf_ax12h_retn
25386 00008114 EB06                    <1>       jmp    short loc_fnf_check_next_cluster
25387                                  <1>
25388                                  <1> loc_fnf_check_FAT_type:
25389 00008116 807E0303                <1>       cmp    byte [esi+LD_FATType], 3
25390 0000811A 72A9                    <1>       jb     short loc_fnf_ax12h_retn
25391                                  <1>
25392                                  <1> loc_fnf_check_next_cluster:
25393 0000811C A1[155B0100]            <1>       mov    eax, [DirBuff_Cluster]
25394 00008121 E821380000              <1>       call   get_next_cluster
25395 00008126 7306                    <1>       jnc    short loc_fnf_load_next_dir_cluster
25396 00008128 09C0                    <1>       or     eax, eax
25397 0000812A 7498                    <1>       jz     short loc_fnf_stc_retn
25398                                  <1>       ;mov   eax, 17 ;Drive not ready or read error
25399 0000812C F5                      <1>       cmc    ;stc
25400                                  <1> loc_fnf_retn:
25401 0000812D C3                      <1>       retn
25402                                  <1>
25403                                  <1> loc_fnf_load_next_dir_cluster:
25404 0000812E E8FA390000              <1>       call   load_FAT_sub_directory
25405 00008133 72F8                    <1>       jc     short loc_fnf_retn
25406 00008135 6631DB                  <1>       xor    bx, bx
25407 00008138 66891D[045D0100]        <1>       mov    [FindFile_DirEntryNumber], bx
25408 0000813F EB9C                    <1>       jmp    short loc_fnf_search
25409                                  <1>
25410                                  <1> get_and_print_longname:
25411                                  <1>       ; 16/10/2016
25412                                  <1>       ; 13/02/2016 (TRDOS 386 =  TRDOS v2.0)
25413                                  <1>       ; 24/01/2010
25414                                  <1>       ; 17/10/2009 (CMD_INTR.ASM, 'cmp_cmd_longname')
25415                                  <1> get_longname_fchar:
25416 00008141 803E20                  <1>       cmp    byte [esi], 20h
25417 00008144 7701                    <1>       ja     short loc_find_longname
25418                                  <1>       ;jb    short loc_longname_retn
25419                                  <1>       ;inc   esi
25420                                  <1>       ;je    short get_longname_fchar
25421                                  <1> ;loc_longname_retn:
25422 00008146 C3                      <1>       retn
25423                                  <1> loc_find_longname:
25424 00008147 E88E210000              <1>       call   find_longname
25425 0000814C 7328                    <1>       jnc    short loc_print_longname
25426                                  <1>
25427 0000814E 08C0                    <1>       or     al, al
25428 00008150 741A                    <1>       jz     short loc_longname_not_found
25429                                  <1>
25430                                  <1>       ; 16/10/2016 (15h -> 15, 17)
25431 00008152 3C0F                    <1>       cmp    al, 15
25432 00008154 0F84C5F7FFFF            <1>       je     cd_drive_not_ready ; drive not ready
25433                                  <1>                                 ; or
25434 0000815A 3C11                    <1>       cmp    al, 17          ; read error
25435 0000815C 0F84BDF7FFFF            <1>       je     cd_drive_not_ready
25436                                  <1>
```

```
25437                                  <1> loc_ln_file_dir_not_found:
25438 00008162 BE[FE090100]            <1>       mov    esi, Msg_File_Directory_Not_Found
25439                                  <1>       ;call  print_msg
25440                                  <1>         ;retn
25441 00008167 E9F1E1FFFF              <1>       jmp    print_msg
25442                                  <1>
25443                                  <1> loc_longname_not_found:
25444 0000816C BE[1D0A0100]            <1>         mov     esi, Msg_LongName_Not_Found
25445                                  <1>       ;call  print_msg
25446                                  <1>         ;retn
25447 00008171 E9E7E1FFFF              <1>       jmp    print_msg
25448                                  <1>
25449                                  <1> loc_print_longname:
25450                                  <1>       ;mov   esi, LongFileName
25451 00008176 BF[E6530100]            <1>       mov    edi, TextBuffer
25452 0000817B 57                      <1>       push   edi
25453 0000817C 3C00                    <1>       cmp    al, 0
25454 0000817E 7708                    <1>       ja     short loc_print_longname_1
25455                                  <1> loc_print_FS_longname: ; Singlix FS (64 byte ASCIIZ file name)
25456 00008180 AC                      <1>       lodsb
25457 00008181 AA                      <1>       stosb
25458 00008182 08C0                    <1>       or     al, al
25459 00008184 75FA                    <1>       jnz    short loc_print_FS_longname
25460 00008186 EB07                    <1>       jmp    short loc_print_longname_2
25461                                  <1>       ;
25462                                  <1> loc_print_longname_1: ; MS Windows long name (UNICODE chars)
25463 00008188 66AD                    <1>       lodsw
25464 0000818A AA                      <1>       stosb
25465 0000818B 08C0                    <1>       or     al, al
25466 0000818D 75F9                    <1>       jnz    short loc_print_longname_1
25467                                  <1>       ;
25468                                  <1> loc_print_longname_2:
25469 0000818F 5E                      <1>       pop    esi
25470 00008190 E8C8E1FFFF              <1>       call   print_msg
25471 00008195 BE[6F130100]            <1>       mov    esi, nextline
25472                                  <1>       ;call  print_msg
25473                                  <1>       ;retn
25474 0000819A E9BEE1FFFF              <1>       jmp    print_msg
25475                                  <1>
25476                                  <1> show_file:
25477                                  <1>       ; 18/02/2016
25478                                  <1>       ; 17/02/2016
25479                                  <1>       ; 15/02/2016 (TRDOS 386 =  TRDOS v2.0)
25480                                  <1>       ; 13/09/2011 (CMD_INTR.ASM, 'cmp_cmd_show')
25481                                  <1>       ; 08/11/2009
25482                                  <1>
25483                                  <1> loc_show_parse_path_name:
25484 0000819F BF[8A5C0100]            <1>       mov    edi, FindFile_Drv
25485 000081A4 E888200000              <1>       call   parse_path_name
25486 000081A9 0F825AF9FFFF            <1>       jc     loc_cmd_failed
25487                                  <1>
25488                                  <1> loc_show_check_filename_exists:
25489 000081AF BE[CC5C0100]            <1>       mov    esi, FindFile_Name
25490 000081B4 803E20                  <1>       cmp    byte [esi], 20h
25491 000081B7 0F864CF9FFFF            <1>       jna    loc_cmd_failed
25492                                  <1>
25493                                  <1>       ; 15/02/2016 (invalid file name check)
25494 000081BD E809020000              <1>       call   check_filename
25495 000081C2 730A                    <1>       jnc    short loc_show_change_drv
25496                                  <1>
25497 000081C4 BE[EA0A0100]            <1>       mov    esi, Msg_invalid_name_chars
25498 000081C9 E98FE1FFFF              <1>       jmp    print_msg
25499                                  <1>
25500                                  <1> loc_show_change_drv:
25501 000081CE 8A35[E6520100]          <1>       mov    dh, [Current_Drv]
25502 000081D4 8835[465B0100]          <1>       mov    [RUN_CDRV], dh
25503 000081DA 8A15[8A5C0100]          <1>       mov    dl, [FindFile_Drv]
25504 000081E0 38F2                    <1>       cmp    dl, dh
25505 000081E2 740B                    <1>       je     short loc_show_change_directory
25506 000081E4 E887EAFFFF              <1>       call   change_current_drive
25507                                  <1>       ;jc    loc_file_rw_cmd_failed
25508 000081E9 0F8245F9FFFF            <1>       jc     loc_run_cmd_failed
25509                                  <1>
25510                                  <1> loc_show_change_directory:
25511 000081EF 803D[8B5C0100]20        <1>       cmp    byte [FindFile_Directory], 20h
25512 000081F6 7618                    <1>       jna    short loc_findload_showfile
25513                                  <1>
25514 000081F8 FE05[D3060100]          <1>       inc    byte [Restore_CDIR]
25515 000081FE BE[8B5C0100]            <1>       mov    esi, FindFile_Directory
25516 00008203 30E4                    <1>       xor    ah, ah ; CD_COMMAND sign -> 0
25517 00008205 E8111A0000              <1>       call   change_current_directory
25518                                  <1>       ;jc    loc_file_rw_cmd_failed
25519 0000820A 0F8224F9FFFF            <1>       jc     loc_run_cmd_failed
25520                                  <1>
25521                                  <1> ;loc_show_change_prompt_dir_string:
25522                                  <1>       ;call  change_prompt_dir_string
25523                                  <1>
25524                                  <1> loc_findload_showfile:
25525                                  <1>       ; 15/02/2016
25526 00008210 BE[CC5C0100]            <1>       mov    esi, FindFile_Name
25527 00008215 BF[7B5C0100]            <1>       mov    edi, Dir_Entry_Name ; Dir Entry Format File Name
25528 0000821A E8471F0000              <1>       call   convert_file_name
25529 0000821F 89FE                    <1>       mov    esi, edi ; offset Dir_Entry_Name
25530                                  <1>
25531 00008221 28C0                    <1>       sub    al, al ; Attrib AND mask = 0
25532                                  <1>       ; Directory attribute : 10h
25533                                  <1>       ; Volume name attribute: 8h
25534 00008223 B418                    <1>       mov    ah, 00011000b ; 18h (Attrib NAND, AND --> zero mask)
25535                                  <1>       ;
25536 00008225 6631C9                  <1>       xor    cx, cx
25537 00008228 E8421C0000              <1>       call   locate_current_dir_file
25538                                  <1>       ;jc    loc_file_rw_cmd_failed
25539 0000822D 0F8201F9FFFF            <1>       jc     loc_run_cmd_failed
```

```
                                <1>
                                <1> loc_show_load_file:
                                <1>       ; EDI = Directory Entry
00008233 668B4714               <1>       mov    ax, [edi+DirEntry_FstClusHI] ; First Cluster High Word
00008237 C1E010                 <1>       shl    eax, 16
0000823A 668B471A               <1>       mov    ax, [edi+DirEntry_FstClusLO] ; First Cluster Low Word
0000823E A3[345D0100]           <1>       mov    [Show_Cluster], eax
00008243 8B471C                 <1>       mov    eax, [edi+DirEntry_FileSize] ; File Size
00008246 21C0                   <1>       and    eax, eax ; Empty file !
00008248 0F8491000000           <1>         jz     end_of_show_file
0000824E A3[385D0100]           <1>       mov    [Show_FileSize], eax
00008253 31C0                   <1>       xor    eax, eax
00008255 A3[3C5D0100]           <1>       mov    [Show_FilePointer], eax ; 0
0000825A 66A3[405D0100]         <1>       mov    [Show_ClusterPointer], ax ; 0
00008260 29DB                   <1>       sub    ebx, ebx
00008262 8A3D[E6520100]         <1>       mov    bh, [Current_Drv]
00008268 BE00010900             <1>       mov    esi, Logical_DOSDisks
0000826D 01DE                   <1>       add    esi, ebx
0000826F 8935[305D0100]         <1>       mov    [Show_LDDDT], esi ; Logical DOS Drv Description Table addr
                                <1>
00008275 807E0300               <1>       cmp    byte [esi+LD_FATType], 0
00008279 7713                   <1>       ja     short loc_show_calculate_cluster_size
                                <1>       ; Singlix FS
                                <1>       ; First Cluster Number is FDT number (in compatibility buffer)
0000827B 8B15[345D0100]         <1>       mov    edx, [Show_Cluster] ; Compatibility dir. buffer value (FDT)
00008281 8915[2C5D0100]         <1>       mov    [Show_FDT], edx
00008287 31C0                   <1>       xor    eax, eax
00008289 A3[345D0100]           <1>       mov    [Show_Cluster], eax ; Sector index  = 0
                                <1>                                  ; (next time it will be 1)
                                <1> loc_show_calculate_cluster_size:
0000828E 668B5E11               <1>       mov    bx, [esi+LD_BPB+BPB_BytsPerSec] ; FAT 12-16-32 (512)
                                <1>       ; BX = 512 = [esi+LD_FS_BytesPerSec] ; Singlix FS
00008292 8A4613                 <1>       mov    al, [esi+LD_BPB+BPB_SecPerClust] ; FAT 12-16-32 (<= 128)
                                <1>       ; AL = 1 = [esi+LD_FS_Reserved2] ; SectPerClust for Singlix FS
00008295 F7E3                   <1>       mul    ebx
                                <1>
                                <1>       ;cmp   eax, 65536 ; non-compatible (very big) cluster size
                                <1>       ;ja    short end_of_show_file
00008297 66A3[425D0100]         <1>       mov    [Show_ClusterSize], ax
                                <1>
                                <1> loc_start_show_file:
0000829D BE[6F130100]           <1>       mov    esi, nextline
000082A2 E8B6E0FFFF             <1>       call   print_msg
                                <1>
000082A7 A1[345D0100]           <1>       mov    eax, [Show_Cluster]
000082AC C605[445D0100]17       <1>       mov    byte [Show_RowCount], 23
                                <1>
                                <1>       ; 17/02/2016
000082B3 8B35[305D0100]         <1>       mov    esi, [Show_LDDDT]
                                <1>
                                <1> loc_show_next_cluster:
                                <1>       ; 15/02/2016
000082B9 BB00000700             <1>       mov    ebx, Cluster_Buffer ; 70000h (for current TRDOS 386 version)
                                <1>       ; ESI = Logical DOS drv description table address
000082BE E8A8380000             <1>       call   read_cluster
                                <1>       ;jc    loc_file_rw_cmd_failed
000082C3 0F826BF8FFFF           <1>       jc     loc_run_cmd_failed
                                <1>
000082C9 31DB                   <1>       xor    ebx, ebx
                                <1> loc_show_next_byte:
000082CB 803D[445D0100]00       <1>       cmp    byte [Show_RowCount], 0
000082D2 7521                   <1>       jne    short pass_show_wait_for_key
000082D4 30E4                   <1>       xor    ah, ah
000082D6 E83B89FFFF             <1>       call   int16h
000082DB 3C1B                   <1>       cmp    al, 1Bh
000082DD 750F                   <1>       jne    short pass_exit_show
                                <1> end_of_show_file:
                                <1> pass_show_file:
000082DF BE[6F130100]           <1>       mov    esi, nextline
000082E4 E874E0FFFF             <1>       call   print_msg
000082E9 E94D010000             <1>       jmp    loc_file_rw_restore_retn
                                <1>
                                <1> pass_exit_show:
000082EE C605[445D0100]14       <1>       mov    byte [Show_RowCount], 20
                                <1> pass_show_wait_for_key:
000082F5 81C300000700           <1>       add    ebx, Cluster_Buffer
000082FB 8A03                   <1>       mov    al, [ebx]
000082FD 3C0D                   <1>       cmp    al, 0Dh
000082FF 0F8590000000           <1>         jne    loc_show_check_tab_space
00008305 FE0D[445D0100]         <1>       dec    byte [Show_RowCount]
                                <1> pass_show_dec_rowcount:
0000830B B307                   <1>       mov    bl, 7 ; (light gray character color, black background)
0000830D 8A3D[4E520100]         <1>       mov    bh, [ACTIVE_PAGE] ; [ptty]
00008313 E89A99FFFF             <1>       call   _write_tty
                                <1> loc_show_check_eof:
00008318 FF05[3C5D0100]         <1>       inc    dword [Show_FilePointer]
0000831E A1[3C5D0100]           <1>       mov    eax, [Show_FilePointer]
00008323 3B05[385D0100]         <1>       cmp    eax, [Show_FileSize]
00008329 73B4                   <1>       jnb    short end_of_show_file
0000832B 66FF05[405D0100]       <1>       inc    word [Show_ClusterPointer]
00008332 0FB71D[405D0100]       <1>       movzx  ebx, word [Show_ClusterPointer]
                                <1>
                                <1>       ; 17/02/2016
                                <1>       ; (sector boundary -9 bits- check, 512 = 0)
00008339 66F7C3FF01             <1>         test   bx, 1FFh ;  1 to 511
0000833E 758B                   <1>       jnz    short loc_show_next_byte
                                <1>
                                <1>       ; 16/02/2016
00008340 8B35[305D0100]         <1>       mov    esi, [Show_LDDDT]
                                <1>       ;
00008346 807E0300               <1>       cmp    byte [esi+LD_FATType], 0
0000834A 7719                   <1>       ja     short loc_show_check_fat_cluster_size
                                <1>
```

```
25643                                  <1>       ; Singlix FS
25644                                  <1>       ; 1 sector, more... (cluster size = 1 sector)
25645 0000834C A1[345D0100]            <1>       mov    eax, [Show_Cluster]
25646 00008351 40                      <1>       inc    eax
25647 00008352 A3[345D0100]            <1>       mov    [Show_Cluster], eax
25648                                  <1>
25649 00008357 6621DB                  <1>       and    bx, bx ; 65536 -> 0
25650 0000835A 0F856BFFFFFF            <1>         jnz  loc_show_next_byte
25651 00008360 E954FFFFFF              <1>       jmp     loc_show_next_cluster
25652                                  <1>
25653                                  <1> loc_show_check_fat_cluster_size:
25654                                  <1>       ; 17/02/2016
25655 00008365 663B1D[425D0100]        <1>       cmp    bx, [Show_ClusterSize] ; cluster size in bytes
25656 0000836C 0F8259FFFFFF            <1>         jb   loc_show_next_byte
25657 00008372 66C705[405D0100]00-     <1>       mov    word [Show_ClusterPointer], 0
25658 0000837A 00                      <1>
25659                                  <1>
25660 0000837B A1[345D0100]            <1>       mov    eax, [Show_Cluster]
25661                                  <1>       ;mov   esi, [Show_LDDDT]
25662                                  <1> loc_show_get_next_cluster:
25663 00008380 E8C2350000              <1>       call   get_next_cluster
25664                                  <1>       ;jc    loc_file_rw_cmd_failed
25665 00008385 0F82A9F7FFFF            <1>       jc     loc_run_cmd_failed
25666                                  <1> loc_show_update_ccluster:
25667 0000838B A3[345D0100]            <1>       mov    [Show_Cluster], eax
25668 00008390 E924FFFFFF              <1>         jmp     loc_show_next_cluster
25669                                  <1>
25670                                  <1> loc_show_check_tab_space:
25671 00008395 3C09                    <1>       cmp    al, 09h
25672 00008397 0F856EFFFFFF            <1>         jne     pass_show_dec_rowcount
25673                                  <1> loc_show_put_tab_space:
25674 0000839D 8A3D[4E520100]          <1>       mov    bh, [ACTIVE_PAGE] ; [ptty]
25675 000083A3 E89995FFFF              <1>       call   get_cpos
25676                                  <1>       ; dl = cursor column
25677 000083A8 80E207                  <1>       and    dl, 7 ; 18/02/2016
25678                                  <1>       ;shr   bh, 1 ; [ACTIVE_PAGE]
25679 000083AB 8A3D[4E520100]          <1>       mov    bh, [ACTIVE_PAGE]
25680 000083B1 B307                    <1>       mov    bl, 7 ; color attribute
25681                                  <1> loc_show_put_space_chars:
25682 000083B3 B020                    <1>       mov    al, 20h ; space
25683                                  <1>       ;mov   bh, [ACTIVE_PAGE] ; [ptty]
25684                                  <1>       ;mov   bl, 7 ; color attribute
25685 000083B5 6652                    <1>       push   dx
25686 000083B7 E8F698FFFF              <1>       call   _write_tty
25687 000083BC 665A                    <1>       pop    dx
25688                                  <1>       ; 18/02/2016
25689 000083BE 80FA07                  <1>       cmp    dl, 7
25690 000083C1 0F8351FFFFFF            <1>       jnb    loc_show_check_eof
25691 000083C7 FEC2                    <1>       inc    dl
25692 000083C9 EBE8                    <1>       jmp    short loc_show_put_space_chars
25693                                  <1>
25694                                  <1> check_filename:
25695                                  <1>       ; 10/10/2016
25696                                  <1>       ; 15/02/2016 (TRDOS 386 =  TRDOS v2.0)
25697                                  <1>       ; 07/08/2010 (FILE.ASM, 'proc_check_filename')
25698                                  <1>       ; 10/07/2010
25699                                  <1>       ; Derived from 'proc_check_filename'
25700                                  <1>       ; in the old TRDOS.ASM (09/02/2005).
25701                                  <1>       ;
25702                                  <1>       ; INPUT ->
25703                                  <1>       ;     ESI = Dot File Name Location
25704                                  <1>       ; OUTPUT ->
25705                                  <1>       ;     cf = 1 -> error code in AL
25706                                  <1>       ;          AL = ERR_INV_FILE_NAME (=26)
25707                                  <1>       ;              Invalid file name chars
25708                                  <1>       ;     cf = 0 -> valid file name
25709                                  <1>       ;
25710                                  <1>       ;(EAX, ECX, EDI will be changed)
25711                                  <1>
25712                                  <1> check_invalid_filename_chars:
25713                                  <1>       ; 15/02/2016 (TRDOS 386 =  TRDOS v2.0)
25714                                  <1>       ; 10/07/2010 (FILE.ASM, 'proc_check_invalid_filename_chars')
25715                                  <1>       ; 10/02/2010
25716                                  <1>       ; Derived from 'proc_check_invalid_filename_chars'
25717                                  <1>       ; in the old TRDOS.ASM (09/02/2005).
25718                                  <1>       ;
25719                                  <1>       ; INPUT ->
25720                                  <1>       ;     ESI = ASCIIZ FileName
25721                                  <1>       ; OUTPUT ->
25722                                  <1>       ;     cf = 1 -> invalid
25723                                  <1>       ;     cf = 0 -> valid
25724                                  <1>       ;
25725                                  <1>       ;(EAX, ECX, EDI will be changed)
25726                                  <1>
25727 000083CB 56                      <1>       push   esi
25728                                  <1>
25729 000083CC BF[D2070100]            <1>         mov     edi, invalid_fname_chars
25730 000083D1 AC                      <1>       lodsb
25731                                  <1> check_filename_next_char:
25732 000083D2 B914000000              <1>       mov    ecx, sizeInvFnChars
25733 000083D7 BF[D2070100]            <1>       mov    edi, invalid_fname_chars
25734                                  <1> loc_scan_invalid_filename_char:
25735 000083DC AE                      <1>       scasb
25736 000083DD 741F                    <1>       je     short loc_invalid_filename_stc
25737 000083DF E2FB                    <1>       loop   loc_scan_invalid_filename_char
25738 000083E1 AC                      <1>       lodsb
25739 000083E2 3C1F                    <1>       cmp    al, 1Fh  ; 20h and above
25740 000083E4 77EC                    <1>       ja     short check_filename_next_char
25741                                  <1>
25742                                  <1> check_filename_dot:
25743 000083E6 8B3424                  <1>       mov    esi, [esp]
25744                                  <1>
25745 000083E9 B421                    <1>       mov    ah, 21h
```

```
25746 000083EB B908000000          <1>       mov    ecx, 8
25747                              <1> loc_check_filename_next_char:
25748 000083F0 AC                  <1>       lodsb
25749 000083F1 3C2E                <1>       cmp    al, 2Eh
25750 000083F3 7511                <1>       jne    short pass_check_fn_dot_check
25751                              <1> loc_check_filename_ext_0:
25752 000083F5 AC                  <1>       lodsb
25753 000083F6 38E0                <1>       cmp    al, ah ; 21h
25754 000083F8 7205                <1>       jb     short loc_invalid_filename
25755 000083FA 3C2E                <1>       cmp    al, 2Eh
25756 000083FC 7519                <1>       jne    short loc_check_filename_ext_1
25757                              <1>
25758                              <1> loc_invalid_filename_stc:
25759                              <1> loc_check_fn_stc_rtn:
25760 000083FE F9                  <1>       stc
25761                              <1> loc_invalid_filename:
25762                              <1>       ; 10/10/2016 (0Bh -> 26)
25763 000083FF B81A000000          <1>       mov    eax, ERR_INV_FILE_NAME ; (=26)
25764                              <1>       ; Invalid file name chars
25765                              <1> loc_check_fn_rtn:
25766 00008404 5E                  <1>       pop    esi
25767 00008405 C3                  <1>       retn
25768                              <1>
25769                              <1> pass_check_fn_dot_check:
25770 00008406 38E0                <1>       cmp    al, ah ; 21h
25771 00008408 7224                <1>       jb     short loc_check_fn_clc_rtn
25772 0000840A E2E4                <1>       loop   loc_check_filename_next_char
25773 0000840C AC                  <1>       lodsb
25774 0000840D 38E0                <1>       cmp    al, ah ; 21h
25775 0000840F 721D                <1>       jb     short loc_check_fn_clc_rtn
25776 00008411 3C2E                <1>       cmp    al, 2Eh
25777 00008413 75E9                <1>       jne    short loc_check_fn_stc_rtn
25778 00008415 EBDE                <1>       jmp    short loc_check_filename_ext_0
25779                              <1>
25780                              <1> loc_check_filename_ext_1:
25781 00008417 AC                  <1>       lodsb
25782 00008418 38E0                <1>       cmp    al, ah ; 21h
25783 0000841A 7212                <1>       jb     short loc_check_fn_clc_rtn
25784 0000841C 3C2E                <1>       cmp    al, 2Eh
25785 0000841E 74DE                <1>       je     short loc_check_fn_stc_rtn
25786 00008420 AC                  <1>       lodsb
25787 00008421 38E0                <1>       cmp    al, ah ; 21h
25788 00008423 7209                <1>       jb     short loc_check_fn_clc_rtn
25789 00008425 3C2E                <1>       cmp    al, 2Eh
25790 00008427 74D5                <1>       je     short loc_check_fn_stc_rtn
25791 00008429 AC                  <1>       lodsb
25792 0000842A 38E0                <1>       cmp    al, ah ; 21h
25793 0000842C 73D0                <1>       jnb    short loc_check_fn_stc_rtn
25794                              <1>
25795                              <1> loc_check_fn_clc_rtn:
25796 0000842E 5E                  <1>       pop    esi
25797 0000842F F8                  <1>       clc
25798 00008430 C3                  <1>       retn
25799                              <1>
25800                              <1> loc_print_deleted_message:
25801 00008431 BE[BF0B0100]        <1>       mov    esi, Msg_Deleted
25802 00008436 E822DFFFFF          <1>       call   print_msg
25803                              <1>
25804                              <1>       ;clc
25805                              <1>
25806                              <1> loc_file_rw_restore_retn:
25807                              <1>       ; 15/02/2016 (TRDOS 386 =  TRDOS v2.0)
25808                              <1>       ; 28/02/2010 (CMD_INTR.ASM)
25809                              <1> loc_file_rw_cmd_failed:
25810 0000843B 9C                  <1>       pushf
25811 0000843C E84DF7FFFF          <1>       call   restore_cdir_after_cmd_fail
25812 00008441 9D                  <1>       popf
25813 00008442 720D                <1>       jc     short loc_file_rw_check_write_fault
25814 00008444 C3                  <1>       retn
25815                              <1>
25816                              <1> loc_permission_denied:
25817                              <1>       ; 27/02/2016
25818 00008445 BE[CC0B0100]        <1>       mov    esi, Msg_Permission_Denied
25819 0000844A E80EDFFFFF          <1>       call   print_msg
25820 0000844F EBEA                <1>       jmp    short loc_file_rw_restore_retn
25821                              <1>
25822                              <1> loc_file_rw_check_write_fault:
25823                              <1>       ;cmp   al, 1Dh ; Write Fault
25824 00008451 3C12                <1>         cmp  al, 18 ; 05/11/2016
25825 00008453 0F85E0F6FFFF        <1>       jne     loc_run_cmd_failed_cmp_al
25826 00008459 BE[B3090100]        <1>       mov    esi, Msg_Not_Ready_Write_Err
25827                              <1>       ;call  print_msg
25828                              <1>       ;retn
25829 0000845E E9FADEFFFF          <1>       jmp    print_msg
25830                              <1>
25831                              <1> make_directory:
25832                              <1>       ; 21/02/2016 (TRDOS 386 =  TRDOS v2.0)
25833                              <1>       ; 12/03/2011 (CMD_INTR.ASM, 'cmp_cmd_mkdir')
25834                              <1>       ; 14/08/2010
25835                              <1>       ; 10/07/2010
25836                              <1>       ; 29/11/2009
25837                              <1>       ;
25838                              <1> get_mkdir_fchar:
25839                              <1>       ; esi = directory name
25840 00008463 803E20              <1>       cmp    byte [esi], 20h
25841 00008466 7701                <1>         ja   short loc_mkdir_parse_path_name
25842                              <1>
25843                              <1> loc_mkdir_nodirname_retn:
25844 00008468 C3                  <1>       retn
25845                              <1>
25846                              <1> loc_mkdir_parse_path_name:
25847 00008469 BF[8A5C0100]        <1>       mov    edi, FindFile_Drv
25848 0000846E E8BE1D0000          <1>         call    parse_path_name
```

```
25849 00008473 0F8290F6FFFF        <1>       jc     loc_cmd_failed
25850                              <1>
25851                              <1> loc_mkdir_check_dirname_exists:
25852 00008479 BE[CC5C0100]        <1>       mov    esi, FindFile_Name
25853 0000847E 803E20              <1>       cmp    byte [esi], 20h
25854 00008481 0F8682F6FFFF        <1>       jna    loc_cmd_failed
25855 00008487 8935[485D0100]      <1>       mov    [DelFile_FNPointer], esi
25856 0000848D E839FFFFFF          <1>       call   check_filename
25857 00008492 7259                <1>       jc     short loc_mkdir_invalid_dir_name_chars
25858                              <1>
25859                              <1> loc_mkdir_drv:
25860 00008494 8A35[E6520100]      <1>       mov    dh, [Current_Drv]
25861 0000849A 8835[465B0100]      <1>       mov    [RUN_CDRV], dh
25862                              <1>
25863 000084A0 8A15[8A5C0100]      <1>       mov    dl, [FindFile_Drv]
25864 000084A6 38F2                <1>       cmp    dl, dh
25865 000084A8 7407                <1>       je     short loc_mkdir_change_directory
25866                              <1>
25867 000084AA E8C1E7FFFF          <1>       call   change_current_drive
25868 000084AF 728A                <1>       jc     loc_file_rw_cmd_failed
25869                              <1>
25870                              <1> loc_mkdir_change_directory:
25871 000084B1 803D[8B5C0100]20    <1>       cmp    byte [FindFile_Directory], 20h
25872 000084B8 7614                <1>       jna    short loc_mkdir_find_directory
25873                              <1>
25874 000084BA FE05[D3060100]      <1>       inc    byte [Restore_CDIR]
25875 000084C0 BE[8B5C0100]        <1>       mov    esi, FindFile_Directory
25876 000084C5 30E4                <1>       xor    ah, ah ; CD_COMMAND sign -> 0
25877 000084C7 E84F170000          <1>       call   change_current_directory
25878 000084CC 722E                <1>       jc     short loc_mkdir_check_error_code
25879                              <1>
25880                              <1> ;loc_mkdir_change_prompt_dir_string:
25881                              <1>       ;call  change_prompt_dir_string
25882                              <1>
25883                              <1> loc_mkdir_find_directory:
25884                              <1>       ;mov   esi, FindFile_Name
25885 000084CE 8B35[485D0100]      <1>       mov    esi, [DelFile_FNPointer]
25886                              <1>       ;xor   eax, eax
25887 000084D4 6631C0              <1>       xor    ax, ax ; any name (dir, file, volume)
25888 000084D7 E82FFBFFFF          <1>       call   find_first_file
25889 000084DC 721E                <1>       jc     short loc_mkdir_check_error_code
25890                              <1>
25891                              <1> loc_mkdir_directory_found:
25892 000084DE BE[170B0100]        <1>       mov    esi, Msg_Name_Exists
25893 000084E3 E875DEFFFF          <1>       call   print_msg
25894                              <1>
25895 000084E8 E94EFFFFFF          <1>         jmp     loc_file_rw_restore_retn
25896                              <1>
25897                              <1> loc_mkdir_invalid_dir_name_chars:
25898 000084ED BE[EA0A0100]        <1>       mov    esi, Msg_invalid_name_chars
25899 000084F2 E866DEFFFF          <1>       call   print_msg
25900                              <1>
25901 000084F7 E93FFFFFFF          <1>         jmp     loc_file_rw_restore_retn
25902                              <1>
25903                              <1> loc_mkdir_check_error_code:
25904 000084FC 3C02                <1>       cmp    al, 2
25905                              <1>       ;je    short loc_mkdir_directory_not_found
25906 000084FE 7406                <1>       je     short loc_mkdir_ask_for_yes_no
25907 00008500 F9                  <1>       stc
25908 00008501 E935FFFFFF          <1>         jmp     loc_file_rw_cmd_failed
25909                              <1>
25910                              <1> loc_mkdir_directory_not_found:
25911                              <1> loc_mkdir_ask_for_yes_no:
25912 00008506 BE[380B0100]        <1>       mov    esi, Msg_DoYouWantMkdir
25913 0000850B E84DDEFFFF          <1>       call   print_msg
25914 00008510 8B35[485D0100]      <1>       mov    esi, [DelFile_FNPointer]
25915 00008516 E842DEFFFF          <1>       call   print_msg
25916 0000851B BE[570B0100]        <1>       mov    esi, Msg_YesNo
25917 00008520 E838DEFFFF          <1>       call   print_msg
25918                              <1>
25919 00008525 C605[610B0100]20    <1>       mov    byte [Y_N_nextline], 20h
25920                              <1>
25921                              <1> loc_mkdir_ask_again:
25922 0000852C 30E4                <1>       xor    ah, ah
25923 0000852E E8E386FFFF          <1>       call   int16h
25924 00008533 3C1B                <1>       cmp    al, 1Bh
25925                              <1>       ;je    short loc_do_not_make_directory
25926 00008535 7447                <1>       je     short loc_mkdir_y_n_escape
25927 00008537 24DF                <1>       and    al, 0DFh ; y -> Y, n -> N
25928 00008539 3C59                <1>       cmp    al, 'Y' ; 'yes'
25929 0000853B 7404                <1>       je     short loc_mkdir_yes_make_directory
25930 0000853D 3C4E                <1>       cmp    al, 'N' ; 'no'
25931 0000853F 75EB                <1>       jne    short loc_mkdir_ask_again
25932                              <1>
25933                              <1> loc_do_not_make_directory:
25934                              <1> loc_mkdir_yes_make_directory:
25935 00008541 A2[610B0100]        <1>       mov    [Y_N_nextline], al
25936 00008546 6650                <1>       push   ax
25937 00008548 BE[610B0100]        <1>       mov    esi, Y_N_nextline
25938 0000854D E80BDEFFFF          <1>       call   print_msg
25939 00008552 6658                <1>       pop    ax
25940                              <1>       ;cmp   al, 'Y' ; 'yes'
25941                              <1>       ;cmc
25942                              <1>         ;jnc loc_file_rw_restore_retn
25943 00008554 3C4E                <1>       cmp    al, 'N' ; 'no'
25944 00008556 0F84DFFEFFFF        <1>         je   loc_file_rw_restore_retn
25945                              <1>
25946                              <1> loc_mkdir_call_make_sub_directory:
25947 0000855C 8B35[485D0100]      <1>       mov    esi, [DelFile_FNPointer]
25948 00008562 B110                <1>       mov    cl, 10h ; Directory attributes
25949 00008564 E8C71D0000          <1>       call   make_sub_directory
25950                              <1> loc_rename_file_ok: ; 06/03/2016
25951 00008569 0F82CCFEFFFF        <1>         jc   loc_file_rw_cmd_failed
```

```
                                <1> move_source_file_to_destination_OK:
0000856F BE[650B0100]           <1>       mov   esi, Msg_OK
00008574 E8E4DDFFFF             <1>       call  print_msg
00008579 E9BDFEFFFF             <1>       jmp   loc_file_rw_restore_retn
                                <1>
                                <1> loc_mkdir_y_n_escape:
0000857E B04E                   <1>       mov   al, 'N' ; 'no'
00008580 EBBF                   <1>       jmp   short loc_do_not_make_directory
                                <1>
                                <1> delete_directory:
                                <1>       ; 15/10/2016
                                <1>       ; 06/03/2016
                                <1>       ; 01/03/2016
                                <1>       ; 29/02/2016
                                <1>       ; 28/02/2016
                                <1>       ; 27/02/2016
                                <1>       ; 26/02/2016 (TRDOS 386 =  TRDOS v2.0)
                                <1>       ; 16/10/2010 (CMD_INTR.ASM, 'cmp_cmd_rmdir')
                                <1>       ; 05/06/2010
                                <1>       ;
                                <1> get_rmdir_fchar:
                                <1>       ; esi = directory name
00008582 803E20                 <1>       cmp   byte [esi], 20h
00008585 7701                   <1>         ja  short loc_rmdir_parse_path_name
                                <1>
                                <1> loc_rmdir_nodirname_retn:
00008587 C3                     <1>       retn
                                <1>
                                <1> loc_rmdir_parse_path_name:
00008588 BF[8A5C0100]           <1>       mov   edi, FindFile_Drv
0000858D E89F1C0000             <1>       call  parse_path_name
00008592 0F8271F5FFFF           <1>       jc    loc_cmd_failed
                                <1>
                                <1> loc_rmdir_check_dirname_exists:
00008598 BE[CC5C0100]           <1>       mov   esi, FindFile_Name
0000859D 803E20                 <1>       cmp   byte [esi], 20h
000085A0 0F8663F5FFFF           <1>       jna   loc_cmd_failed
000085A6 8935[485D0100]         <1>       mov   [DelFile_FNPointer], esi
                                <1>
                                <1> loc_rmdir_drv:
000085AC 8A35[E6520100]         <1>       mov   dh, [Current_Drv]
000085B2 8835[465B0100]         <1>       mov   [RUN_CDRV], dh
                                <1>
000085B8 8A15[8A5C0100]         <1>       mov   dl, [FindFile_Drv]
000085BE 38F2                   <1>       cmp   dl, dh
000085C0 740B                   <1>       je    short loc_rmdir_change_directory
                                <1>
000085C2 E8A9E6FFFF             <1>       call  change_current_drive
000085C7 0F826EFEFFFF           <1>       jc    loc_file_rw_cmd_failed
                                <1>
                                <1> loc_rmdir_change_directory:
000085CD 803D[8B5C0100]20       <1>       cmp   byte [FindFile_Directory], 20h
000085D4 7614                   <1>       jna   short loc_rmdir_find_directory
                                <1>
000085D6 FE05[D3060100]         <1>       inc   byte [Restore_CDIR]
000085DC BE[8B5C0100]           <1>       mov   esi, FindFile_Directory
000085E1 30E4                   <1>       xor   ah, ah ; CD_COMMAND sign -> 0
000085E3 E833160000             <1>       call  change_current_directory
000085E8 7211                   <1>       jc    short loc_rmdir_check_error_code
                                <1>
                                <1> ;loc_rmdir_change_prompt_dir_string:
                                <1>       ;call change_prompt_dir_string
                                <1>
                                <1> loc_rmdir_find_directory:
                                <1>       ;mov  esi, FindFile_Name
000085EA 8B35[485D0100]         <1>       mov   esi, [DelFile_FNPointer]
000085F0 66B81008               <1>       mov   ax, 0810h ; Only directories
000085F4 E812FAFFFF             <1>       call  find_first_file
000085F9 730A                   <1>       jnc   short loc_rmdir_ambgfn_check
                                <1>
                                <1> loc_rmdir_check_error_code:
000085FB 3C02                   <1>       cmp   al, 2
000085FD 740B                   <1>       je    short loc_rmdir_directory_not_found
000085FF F9                     <1>       stc
00008600 E936FEFFFF             <1>       jmp   loc_file_rw_cmd_failed
                                <1>
                                <1> loc_rmdir_ambgfn_check:
00008605 6621D2                 <1>       and   dx, dx ; Ambiguous filename chars used sign (DX>0)
00008608 740F                   <1>       jz    short loc_rmdir_directory_found
                                <1>
                                <1> loc_rmdir_directory_not_found:
0000860A BE[D5090100]           <1>       mov   esi, Msg_Dir_Not_Found
0000860F E849DDFFFF             <1>       call  print_msg
                                <1>
00008614 E922FEFFFF             <1>       jmp   loc_file_rw_restore_retn
                                <1>
                                <1> loc_rmdir_directory_found:
00008619 80E307                 <1>       and   bl, 07h ; Attributes
0000861C 0F8523FEFFFF           <1>       jnz   loc_permission_denied
                                <1>
                                <1> loc_rmdir_save_lnel: ; 28/02/2016
                                <1>        ;mov  bh, [LongName_EntryLength]
00008622 883D[525D0100]         <1>       mov   [DelFile_LNEL], bh ; Long name entry length (if > 0)
                                <1>       ; edi = Directory Entry Offset (DirBuff)
                                <1>       ; esi = Directory Entry (FFF Structure)
                                <1>       ;mov  [DelFile_DirEntryAddr], edi ; not required
                                <1>       ;mov  ax, [edi+20] ; First Cluster High Word
                                <1>         ;shl eax, 16
                                <1>       ;mov  ax, [edi+26] ; First Cluster Low Word
                                <1>       ; ROOT Dir First Cluster = 0
                                <1>         ;cmp eax, 2
                                <1>       ;jb   loc_update_direntry_1
                                <1>
```

```
26055                                  <1> pass_rmdir_fc_check:
26056 00008628 57                      <1>       push   edi ; * (29/02/2016)
26057                                  <1>
26058 00008629 BE[6B0B0100]            <1>       mov    esi, Msg_DoYouWantRmDir
26059 0000862E E82ADDFFFF              <1>       call   print_msg
26060 00008633 8B35[485D0100]          <1>       mov    esi, [DelFile_FNPointer]
26061 00008639 E81FDDFFFF              <1>       call   print_msg
26062 0000863E BE[570B0100]            <1>       mov    esi, Msg_YesNo
26063 00008643 E815DDFFFF              <1>       call   print_msg
26064                                  <1>
26065                                  <1> loc_rmdir_ask_again:
26066 00008648 30E4                    <1>       xor    ah, ah
26067 0000864A E8C785FFFF              <1>       call   int16h
26068 0000864F 3C1B                    <1>       cmp    al, 1Bh
26069                                  <1>       ;je    short loc_do_not_delete_directory
26070 00008651 0F8498000000            <1>         je     loc_rmdir_y_n_escape ; 06/03/2016
26071 00008657 24DF                    <1>       and    al, 0DFh
26072 00008659 A2[610B0100]            <1>       mov    [Y_N_nextline], al
26073 0000865E 3C59                    <1>       cmp    al, 'Y'
26074 00008660 7404                    <1>       je     short loc_rmdir_yes_delete_directory
26075 00008662 3C4E                    <1>       cmp    al, 'N'
26076 00008664 75E2                    <1>       jne    short loc_rmdir_ask_again
26077                                  <1>
26078                                  <1> loc_do_not_delete_directory:
26079                                  <1> loc_rmdir_yes_delete_directory:
26080 00008666 A2[610B0100]            <1>       mov    [Y_N_nextline], al
26081 0000866B 6650                    <1>       push   ax
26082 0000866D BE[610B0100]            <1>       mov    esi, Y_N_nextline
26083 00008672 E8E6DCFFFF              <1>       call   print_msg
26084 00008677 6658                    <1>       pop    ax
26085 00008679 5F                      <1>       pop    edi ; * (29/02/2016)
26086                                  <1>       ;cmp   al, 'Y' ; 'yes'
26087                                  <1>       ;cmc
26088                                  <1>         ;jnc loc_file_rw_restore_retn
26089 0000867A 3C4E                    <1>       cmp    al, 'N' ; 'no'
26090 0000867C 0F84B9FDFFFF            <1>         je   loc_file_rw_restore_retn
26091                                  <1>
26092                                  <1> loc_rmdir_delete_short_name_check_dir_empty:
26093                                  <1>       ; EDI = Directory buffer entry offset/address
26094 00008682 668B4714                <1>       mov    ax, [edi+20] ; First Cluster High Word
26095 00008686 C1E010                  <1>         shl  eax, 16
26096 00008689 668B471A                <1>       mov    ax, [edi+26] ; First Cluster Low Word
26097                                  <1>
26098 0000868D A3[4C5D0100]            <1>       mov    [DelFile_FCluster], eax
26099                                  <1>
26100                                  <1>       ;mov   bx, [DirBuff_EntryCounter]
26101 00008692 668B1D[045D0100]        <1>       mov    bx, [FindFile_DirEntryNumber] ; 27/02/2016
26102 00008699 66891D[505D0100]        <1>       mov    [DelFile_EntryCounter], bx
26103                                  <1>
26104 000086A0 29DB                    <1>       sub    ebx, ebx
26105 000086A2 8A3D[8A5C0100]          <1>       mov    bh, [FindFile_Drv]
26106 000086A8 BE00010900              <1>       mov    esi, Logical_DOSDisks
26107 000086AD 01DE                    <1>       add    esi, ebx
26108                                  <1>
26109 000086AF 66817F0CA101            <1>       cmp    word [edi+DirEntry_NTRes], 01A1h
26110 000086B5 743F                    <1>       je     short loc_rmdir_delete_fs_directory
26111                                  <1>
26112                                  <1>       ;cmp   byte [esi+LD_FATType], 1
26113                                  <1>       ;jnb   short loc_rmdir_get__last_cluster_0
26114                                  <1>       ;mov   eax, 0Bh ; Invalid Format
26115                                  <1>       ;jmp   loc_file_rw_cmd_failed
26116                                  <1>
26117                                  <1> ;loc_rmdir_get_last_cluster_0:
26118 000086B7 8B15[155B0100]          <1>       mov    edx, [DirBuff_Cluster]
26119 000086BD 8915[7C5D0100]          <1>       mov    [RmDir_ParentDirCluster], edx
26120                                  <1>
26121 000086C3 893D[785D0100]          <1>       mov    [RmDir_DirEntryOffset], edi
26122                                  <1>
26123                                  <1>       ; 01/03/2016
26124 000086C9 C705[065B0100]0000-     <1>       mov    dword [FAT_ClusterCounter], 0 ; Reset
26125 000086D1 0000                    <1>
26126                                  <1>
26127                                  <1> loc_rmdir_get_last_cluster:
26128 000086D3 E86A3A0000              <1>       call   get_last_cluster
26129 000086D8 0F82B8000000            <1>         jc     loc_rmdir_cmd_failed
26130                                  <1>
26131 000086DE 3B05[4C5D0100]          <1>       cmp    eax, [DelFile_FCluster]
26132 000086E4 752F                    <1>       jne    short loc_rmdir_multi_dir_clusters
26133                                  <1>
26134 000086E6 C605[775D0100]00        <1>       mov    byte [RmDir_MultiClusters], 0
26135 000086ED EB2D                    <1>       jmp    short pass_rmdir_multi_dir_clusters
26136                                  <1>
26137                                  <1> loc_rmdir_y_n_escape:
26138 000086EF B04E                    <1>       mov    al, 'N' ; 'no'
26139 000086F1 E970FFFFFF              <1>         jmp    loc_do_not_delete_directory
26140                                  <1>
26141                                  <1> loc_rmdir_delete_fs_directory:
26142 000086F6 807E04A1                <1>       cmp    byte [esi+LD_FSType], 0A1h
26143 000086FA 0F8545FDFFFF            <1>       jne    loc_permission_denied
26144                                  <1>
26145 00008700 E833140000              <1>       call   delete_fs_directory
26146 00008705 0F8326FDFFFF            <1>       jnc    loc_print_deleted_message
26147                                  <1>
26148 0000870B 09C0                    <1>       or     eax, eax
26149 0000870D 745D                    <1>       jz     loc_rmdir_directory_not_empty_2
26150 0000870F F9                      <1>       stc
26151 00008710 E926FDFFFF              <1>       jmp    loc_file_rw_cmd_failed
26152                                  <1>
26153                                  <1> loc_rmdir_multi_dir_clusters:
26154 00008715 C605[775D0100]01        <1>       mov    byte [RmDir_MultiClusters], 1
26155                                  <1>
26156                                  <1> pass_rmdir_multi_dir_clusters:
26157 0000871C A3[805D0100]            <1>       mov    [RmDir_DirLastCluster], eax
```

```
26158 00008721 890D[845D0100]      <1>       mov    [RmDir_PreviousCluster], ecx
26159                              <1>
26160                              <1> loc_rmdir_load_fat_sub_directory:
26161 00008727 E801340000          <1>       call   load_FAT_sub_directory
26162 0000872C 7268                <1>       jc     loc_rmdir_cmd_failed
26163                              <1>
26164                              <1> loc_rmdir_find_last_dir_entry:
26165 0000872E 56                  <1>       push   esi
26166 0000872F BE[6E5C0100]        <1>       mov    esi, Dir_File_Name
26167 00008734 C6062A              <1>       mov    byte [esi], '*'
26168 00008737 C646082A            <1>       mov    byte [esi+8], '*'
26169 0000873B 31DB                <1>       xor    ebx, ebx ; Entry offset  = 0
26170                              <1> loc_rmdir_find_last_dir_entry_next:
26171 0000873D 66B80008            <1>       mov    ax, 0800h ; Except volume/long names
26172 00008741 6631C9              <1>       xor    cx, cx ; 0 = Find a valid file or dir name
26173 00008744 E82A180000          <1>       call   find_directory_entry
26174 00008749 7271                <1>       jc     short loc_rmdir_empty_dir_cluster
26175 0000874B 83FB01              <1>       cmp    ebx, 1
26176 0000874E 771B                <1>       ja     short loc_rmdir_directory_not_empty_1
26177                              <1> loc_rmdir_dot_entry_check:
26178 00008750 80FD2E              <1>       cmp    ch, '.' ; The first char of the dir entry
26179 00008753 7516                <1>       jne    short loc_rmdir_directory_not_empty_1
26180 00008755 08DB                <1>       or     bl, bl
26181 00008757 7506                <1>       jnz    short loc_rmdir_dotdot_entry_check
26182 00008759 807F0120            <1>       cmp    byte [edi+1], 20h
26183 0000875D EB06                <1>       jmp    short pass_rmdir_dot_entry_check
26184                              <1>
26185                              <1> loc_rmdir_dotdot_entry_check:
26186 0000875F 66817F012E20        <1>       cmp    word [edi+1], '. '
26187                              <1> pass_rmdir_dot_entry_check:
26188 00008765 7504                <1>       jne    short loc_rmdir_directory_not_empty_1
26189 00008767 FEC3                <1>       inc    bl
26190 00008769 EBD2                <1>       jmp    short loc_rmdir_find_last_dir_entry_next
26191                              <1>
26192                              <1>
26193                              <1> loc_rmdir_directory_not_empty_1:
26194 0000876B 58                  <1>       pop    eax ; pushed esi
26195                              <1>
26196                              <1> loc_rmdir_directory_not_empty_2:
26197 0000876C BE[8C0B0100]        <1>       mov    esi, Msg_Dir_Not_Empty
26198 00008771 E8E7DBFFFF          <1>       call   print_msg
26199                              <1>       ; 01/03/2016
26200 00008776 A1[065B0100]        <1>       mov    eax, [FAT_ClusterCounter]
26201 0000877B 09C0                <1>       or     eax, eax ; 0 ?
26202 0000877D 0F84B8FCFFFF        <1>       jz     loc_file_rw_restore_retn
26203                              <1>       ; ESI = Logical DOS Drive Description Table address
26204                              <1>
26205 00008783 66BB01FF            <1>       mov    bx, 0FF01h ; BH = FFh -> use ESI for Drive parameters
26206                              <1>                   ; BL = 1 -> add free clusters
26207 00008787 E837380000          <1>       call   calculate_fat_freespace
26208 0000878C 09C9                <1>       or     ecx, ecx
26209 0000878E 0F84A7FCFFFF        <1>         jz      loc_file_rw_restore_retn ; ecx = 0 -> OK
26210                              <1>       ; ecx > 0 -> Error (Recalculation is neeeded)
26211 00008794 EB0E                <1>       jmp    short loc_rmdir_cmd_return
26212                              <1>
26213                              <1>
26214                              <1> loc_rmdir_cmd_failed:
26215 00008796 833D[065B0100]01    <1>       cmp    dword [FAT_ClusterCounter], 1
26216 0000879D 0F8298FCFFFF        <1>       jb     loc_file_rw_cmd_failed
26217 000087A3 F9                  <1>       stc
26218                              <1> loc_rmdir_cmd_return:
26219                              <1>       ; 01/03/2016
26220 000087A4 9C                  <1>       pushf
26221                              <1>       ; ESI = Logical DOS Drive Description Table address
26222 000087A5 66BB00FF            <1>       mov    bx, 0FF00h ; BH = FFh -> use ESI for Drive parameters
26223                              <1>                   ; BL = 0 -> Recalculate free cluster count
26224 000087A9 50                  <1>       push   eax
26225 000087AA E814380000          <1>       call   calculate_fat_freespace
26226 000087AF 58                  <1>       pop    eax
26227 000087B0 9D                  <1>       popf
26228 000087B1 0F8284FCFFFF        <1>       jc     loc_file_rw_cmd_failed
26229 000087B7 E97FFCFFFF          <1>       jmp    loc_file_rw_restore_retn
26230                              <1>
26231                              <1>
26232                              <1> loc_rmdir_empty_dir_cluster:
26233 000087BC 5E                  <1>       pop    esi
26234                              <1>
26235                              <1> loc_rmdir_set_prev_cluster_dir_last_cluster:
26236 000087BD 803D[775D0100]00    <1>       cmp    byte [RmDir_MultiClusters], 0
26237 000087C4 761D                <1>       jna    short loc_rmdir_unlink_dir_last_cluster
26238                              <1>
26239 000087C6 A1[845D0100]        <1>       mov    eax, [RmDir_PreviousCluster]
26240                              <1>       ;xor   ecx, ecx
26241 000087CB 49                  <1>       dec    ecx ; FFFFFFFFh
26242 000087CC E8A0340000          <1>       call   update_cluster
26243 000087D1 7310                <1>       jnc    short loc_rmdir_unlink_dir_last_cluster
26244                              <1>
26245                              <1> loc_rmdir_unlink_stc_retn:
26246                              <1>       ; 01/03/2016
26247 000087D3 83F801              <1>       cmp    eax, 1  ; eax = 0 -> end of cluster chain
26248 000087D6 F5                  <1>       cmc
26249 000087D7 72BD                <1>       jc     short loc_rmdir_cmd_failed
26250 000087D9 EB1D                <1>       jmp    short loc_rmdir_save_fat_buffer
26251                              <1>
26252                              <1> loc_rmdir_unlink_stc_retn_0Bh:
26253 000087DB B81C000000          <1>       mov    eax, 28 ; Invalid format ; 15/10/2016
26254 000087E0 F9                  <1>       stc
26255 000087E1 EBB3                <1>       jmp    short loc_rmdir_cmd_failed
26256                              <1>
26257                              <1> loc_rmdir_unlink_dir_last_cluster:
26258 000087E3 A1[805D0100]        <1>       mov    eax, [RmDir_DirLastCluster]
26259 000087E8 31C9                <1>       xor    ecx, ecx ; 0
26260 000087EA E882340000          <1>       call   update_cluster
```

```
26261 000087EF 73EA               <1>       jnc    short loc_rmdir_unlink_stc_retn_0Bh
26262                             <1>       ; Because of it is the last cluster
26263                             <1>       ; 'update_cluster' must return with eocc error
26264 000087F1 09C0               <1>       or     eax, eax
26265 000087F3 7403               <1>       jz     short loc_rmdir_save_fat_buffer ; eocc
26266 000087F5 F9                 <1>       stc
26267 000087F6 EB9E               <1>         jmp     short loc_rmdir_cmd_failed
26268                             <1>
26269                             <1> loc_rmdir_save_fat_buffer:
26270 000087F8 803D[FE5A0100]02   <1>       cmp    byte [FAT_BuffValidData], 2
26271 000087FF 7525               <1>       jne    short loc_rmdir_calculate_FAT_freespace
26272 00008801 E828370000         <1>       call   save_fat_buffer
26273 00008806 728E               <1>       jc     short loc_rmdir_cmd_failed
26274                             <1>
26275                             <1>       ; 01/03/2016
26276 00008808 803D[775D0100]00   <1>       cmp    byte [RmDir_MultiClusters], 0
26277 0000880F 7615               <1>       jna    short loc_rmdir_calculate_FAT_freespace
26278                             <1>
26279 00008811 A1[4C5D0100]       <1>       mov    eax, [DelFile_FCluster]
26280 00008816 E9B8FEFFFF         <1>         jmp     loc_rmdir_get_last_cluster
26281                             <1>
26282                             <1> loc_rmdir_delete_short_name_invalid_data:
26283 0000881B B81D000000         <1>       mov    eax, 29 ; Invalid data (15/10/2016)
26284 00008820 F9                 <1>       stc
26285 00008821 E970FFFFFF         <1>         jmp     loc_rmdir_cmd_failed
26286                             <1>
26287                             <1> loc_rmdir_calculate_FAT_freespace:
26288 00008826 A1[065B0100]       <1>       mov    eax, [FAT_ClusterCounter]
26289 0000882B 66BB01FF           <1>       mov    bx, 0FF01h
26290                             <1>       ; BL = 1 -> Add EAX to free space count
26291                             <1>       ; BH = FFh ->
26292                             <1>       ; ESI = Logical DOS Drive Description Table address
26293 0000882F E88F370000         <1>       call   calculate_fat_freespace
26294                             <1>
26295 00008834 21C9               <1>       and    ecx, ecx ; ecx = 0 -> valid free sector count
26296 00008836 7409               <1>       jz     short loc_rmdir_delete_short_name_continue
26297                             <1>
26298                             <1> loc_rmdir_recalculate_FAT_freespace:
26299 00008838 66BB00FF           <1>         mov     bx, 0FF00h ; BL = 0 -> Recalculate free space
26300 0000883C E882370000         <1>       call   calculate_fat_freespace
26301                             <1>
26302                             <1> loc_rmdir_delete_short_name_continue:
26303 00008841 A1[7C5D0100]       <1>       mov    eax, [RmDir_ParentDirCluster]
26304 00008846 83F802             <1>       cmp    eax, 2
26305 00008849 730D               <1>       jnb    short loc_rmdir_del_short_name_load_sub_dir
26306 0000884B E852320000         <1>       call   load_FAT_root_directory
26307 00008850 0F82E5FBFFFF       <1>       jc     loc_file_rw_cmd_failed
26308 00008856 EB0B               <1>       jmp    short loc_rmdir_del_short_name_ld_chk_fclust
26309                             <1>
26310                             <1> loc_rmdir_del_short_name_load_sub_dir:
26311 00008858 E8D0320000         <1>       call   load_FAT_sub_directory
26312 0000885D 0F82D8FBFFFF       <1>       jc     loc_file_rw_cmd_failed
26313                             <1>
26314                             <1> loc_rmdir_del_short_name_ld_chk_fclust:
26315 00008863 0FB73D[785D0100]   <1>       movzx  edi, word [RmDir_DirEntryOffset]
26316 0000886A 81C700000800       <1>       add    edi, Directory_Buffer
26317                             <1>
26318 00008870 668B4714           <1>       mov    ax, [edi+20] ; First Cluster High Word
26319 00008874 C1E010             <1>       shl    eax, 16
26320 00008877 668B471A           <1>       mov    ax, [edi+26] ; First Cluster Low Word
26321                             <1>         ; Not necessary...
26322 0000887B 3B05[4C5D0100]     <1>       cmp    eax, [DelFile_FCluster]
26323 00008881 7598               <1>       jne    short loc_rmdir_delete_short_name_invalid_data
26324                             <1>       ;
26325 00008883 C607E5             <1>       mov    byte [edi], 0E5h ; 'Deleted' sign
26326                             <1>       ; 27/02/2016
26327                             <1>       ; TRDOS v1 has a bug here! it does not set
26328                             <1>       ; 'DirBuff_ValidData' to 2; as result of this bug,
26329                             <1>       ; 'save_directory_buffer' would not save the change !
26330 00008886 C605[105B0100]02    <1>       mov    byte [DirBuff_ValidData], 2 ; change sign
26331                             <1>       ;
26332 0000888D E8031E0000         <1>       call   save_directory_buffer
26333 00008892 0F82A3FBFFFF       <1>       jc     loc_file_rw_cmd_failed
26334                             <1>
26335                             <1> loc_rmdir_del_long_name:
26336 00008898 0FB615[525D0100]   <1>       movzx  edx, byte [DelFile_LNEL]
26337 0000889F 08D2               <1>       or     dl, dl
26338 000088A1 7414               <1>       jz     short loc_rmdir_update_parent_dir_lmdt
26339                             <1>
26340 000088A3 0FB705[505D0100]   <1>       movzx  eax, word [DelFile_EntryCounter]
26341 000088AA 29D0               <1>       sub    eax, edx
26342 000088AC 0F8289FBFFFF       <1>       jc     loc_file_rw_cmd_failed
26343                             <1>
26344                             <1>       ; EAX = Directory Entry Number of the long name last entry
26345 000088B2 E83E1F0000         <1>       call   delete_longname
26346                             <1>       ;jc    short loc_file_rw_cmd_failed
26347                             <1>
26348                             <1> loc_rmdir_update_parent_dir_lmdt:
26349 000088B7 E8741E0000         <1>       call   update_parent_dir_lmdt
26350                             <1>       ;jc    short loc_file_rw_cmd_failed
26351                             <1>
26352                             <1> loc_rmdir_ok:
26353 000088BC BE[650B0100]       <1>       mov    esi, Msg_OK
26354 000088C1 E897DAFFFF         <1>       call   print_msg
26355 000088C6 E970FBFFFF         <1>       jmp    loc_file_rw_restore_retn
26356                             <1>
26357                             <1>
26358                             <1> delete_file:
26359                             <1>       ; 29/02/2016
26360                             <1>       ; 28/02/2016 (TRDOS 386 =  TRDOS v2.0)
26361                             <1>       ; 09/08/2010 (CMD_INTR.ASM, 'cmp_cmd_del')
26362                             <1>       ; 28/02/2010
26363                             <1>
```

```
                            <1> get_delfile_fchar:
                            <1>       ; esi = file name
000088CB 803E20             <1>       cmp    byte [esi], 20h
000088CE 7701               <1>         ja   short loc_delfile_parse_path_name
                            <1>
                            <1> loc_delfile_nofilename_retn:
000088D0 C3                 <1>       retn
                            <1>
                            <1> loc_delfile_parse_path_name:
000088D1 BF[8A5C0100]       <1>       mov    edi, FindFile_Drv
000088D6 E856190000         <1>       call   parse_path_name
000088DB 0F8228F2FFFF       <1>       jc     loc_cmd_failed
                            <1>
                            <1> loc_delfile_check_filename_exists:
000088E1 BE[CC5C0100]       <1>       mov    esi, FindFile_Name
000088E6 803E20             <1>       cmp    byte [esi], 20h
000088E9 0F861AF2FFFF       <1>       jna    loc_cmd_failed
000088EF 8935[485D0100]     <1>       mov    [DelFile_FNPointer], esi
                            <1>
                            <1> loc_delfile_drv:
000088F5 8A15[8A5C0100]     <1>       mov    dl, [FindFile_Drv]
000088FB 8A35[E6520100]     <1>       mov    dh, [Current_Drv]
00008901 8835[465B0100]     <1>       mov    [RUN_CDRV], dh
00008907 38F2               <1>       cmp    dl, dh
00008909 740B               <1>       je     short loc_delfile_change_directory
                            <1>
0000890B E860E3FFFF         <1>       call   change_current_drive
00008910 0F8225FBFFFF       <1>       jc     loc_file_rw_cmd_failed
                            <1>
                            <1> loc_delfile_change_directory:
00008916 803D[8B5C0100]20   <1>       cmp    byte [FindFile_Directory], 20h
0000891D 7618               <1>       jna    short loc_delfile_find
                            <1>
0000891F FE05[D3060100]     <1>       inc    byte [Restore_CDIR]
00008925 BE[8B5C0100]       <1>       mov    esi, FindFile_Directory
0000892A 30E4               <1>       xor    ah, ah ; CD_COMMAND sign -> 0
0000892C E8EA120000         <1>       call   change_current_directory
00008931 0F8204FBFFFF       <1>       jc     loc_file_rw_cmd_failed
                            <1>
                            <1> ;loc_delfile_change_prompt_dir_string:
                            <1>       ;call  change_prompt_dir_string
                            <1>
                            <1> loc_delfile_find:
                            <1>       ;mov   esi, FindFile_Name
00008937 8B35[485D0100]     <1>       mov    esi, [DelFile_FNPointer]
0000893D 66B80018           <1>       mov    ax, 1800h ; Except volume label and dirs
00008941 E8C5F6FFFF         <1>       call   find_first_file
00008946 0F82EFFAFFFF       <1>       jc     loc_file_rw_cmd_failed
                            <1>
                            <1> loc_delfile_ambgfn_check:
0000894C 6621D2             <1>       and    dx, dx ; Ambiguous filename chars used sign (DX>0)
0000894F 740B               <1>       jz     short loc_delfile_found
                            <1>
                            <1> loc_file_not_found:
00008951 B802000000         <1>       mov    eax, 2 ; File not found sign
00008956 F9                 <1>       stc
00008957 E9DFFAFFFF         <1>       jmp    loc_file_rw_cmd_failed
                            <1>
                            <1> loc_delfile_found:
0000895C 80E307             <1>       and    bl, 07h ; Attributes
0000895F 0F85E0FAFFFF       <1>         jnz     loc_permission_denied
                            <1>
                            <1> ;loc_delfile_found_save_lnel:
                            <1> ;     mov    [DelFile_LNEL], bh ; Long name entry length (if > 0)
                            <1>
                            <1> loc_delfile_ask_for_delete:
00008965 57                 <1>       push   edi ; * (29/02/2016)
                            <1>
00008966 BE[A30B0100]       <1>       mov    esi, Msg_DoYouWantDelete
0000896B E8EDD9FFFF         <1>       call   print_msg
00008970 8B35[485D0100]     <1>       mov    esi, [DelFile_FNPointer]
00008976 E8E2D9FFFF         <1>       call   print_msg
0000897B BE[570B0100]       <1>       mov    esi, Msg_YesNo
00008980 E8D8D9FFFF         <1>       call   print_msg
                            <1>
                            <1> loc_delfile_ask_again:
00008985 30E4               <1>       xor    ah, ah
00008987 E88A82FFFF         <1>       call   int16h
0000898C 3C1B               <1>       cmp    al, 1Bh
                            <1>       ;je    short loc_do_not_delete_file
0000898E 7457               <1>       je     short loc_delfile_y_n_escape ; 06/03/2016
00008990 24DF               <1>       and    al, 0DFh
00008992 A2[610B0100]       <1>       mov    [Y_N_nextline], al
00008997 3C59               <1>       cmp    al, 'Y'
00008999 7404               <1>       je     short loc_yes_delete_file
0000899B 3C4E               <1>       cmp    al, 'N'
0000899D 75E6               <1>       jne    short loc_delfile_ask_again
                            <1>
                            <1> loc_do_not_delete_file:
                            <1> loc_yes_delete_file:
0000899F A2[610B0100]       <1>       mov    [Y_N_nextline], al
000089A4 6650               <1>       push   ax
000089A6 BE[610B0100]       <1>       mov    esi, Y_N_nextline
000089AB E8ADD9FFFF         <1>       call   print_msg
000089B0 6658               <1>       pop    ax
000089B2 5F                 <1>       pop    edi ; * (29/02/2016)
                            <1>       ;cmp   al, 'Y' ; 'yes'
                            <1>       ;cmc
                            <1>         ;jnc loc_file_rw_restore_retn
000089B3 3C4E               <1>       cmp    al, 'N' ; 'no'
000089B5 0F8480FAFFFF       <1>         je   loc_file_rw_restore_retn
                            <1>
                            <1> loc_delete_file:
```

```
26467 000089BB 8A3D[8A5C0100]      <1>       mov    bh, [FindFile_Drv]
26468                              <1>       ;mov   bl, [DelFile_LNEL]
26469 000089C1 8A1D[D95C0100]      <1>       mov    bl, [FindFile_LongNameEntryLength]
26470                              <1>       ;mov   cx, [DirBuff_EntryCounter]
26471 000089C7 668B0D[045D0100]    <1>       mov    cx, [FindFile_DirEntryNumber]
26472                              <1>       ; (*) EDI = Directory buffer entry offset/address
26473 000089CE E80C200000          <1>       call   remove_file ; (FILE.ASM, 'proc_delete_file')
26474 000089D3 0F8358FAFFFF        <1>       jnc    loc_print_deleted_message
26475                              <1>
26476 000089D9 3C05                <1>       cmp    al, 05h
26477 000089DB 0F8464FAFFFF        <1>       je     loc_permission_denied
26478 000089E1 F9                  <1>       stc
26479 000089E2 E954FAFFFF          <1>       jmp    loc_file_rw_cmd_failed
26480                              <1>
26481                              <1> loc_delfile_y_n_escape:
26482 000089E7 B04E                <1>       mov    al, 'N' ; 'no'
26483 000089E9 EBB4                <1>       jmp    short loc_do_not_delete_file
26484                              <1>
26485                              <1> set_file_attributes:
26486                              <1>       ; 06/03/2016
26487                              <1>       ; 04/03/2016 (TRDOS 386 =  TRDOS v2.0)
26488                              <1>       ; 10/07/2010 (TRDOS v1, CMD_INTR.ASM, 'cmp_cmd_attrib')
26489                              <1>       ; 23/05/2010
26490                              <1>       ; 17/12/2000 (P2000.ASM)
26491                              <1>
26492                              <1>       ; esi = file or directory name
26493 000089EB 6631C0             <1>       xor    ax, ax
26494 000089EE 66A3[F40B0100]      <1>       mov    [Attr_Chars], ax
26495 000089F4 A2[A05D0100]        <1>       mov    [Attributes], al
26496                              <1>
26497                              <1> get_attrib_fchar:
26498                              <1>       ; esi = file name
26499 000089F9 8A06                <1>       mov    al, [esi]
26500 000089FB 3C20                <1>       cmp    al, 20h
26501 000089FD 7623                <1>       jna    short loc_attr_file_nofilename_retn
26502                              <1>
26503                              <1> loc_scan_attrib_params:
26504 000089FF 3C2D                <1>       cmp    al, '-'
26505 00008A01 0F871C010000        <1>       ja     loc_attr_file_parse_path_name
26506 00008A07 7408                <1>       je     short loc_attr_space
26507                              <1>
26508 00008A09 3C2B                <1>       cmp    al, '+'
26509 00008A0B 0F85F8F0FFFF        <1>       jne    loc_cmd_failed
26510                              <1>
26511                              <1> loc_attr_space:
26512 00008A11 8A6601              <1>       mov    ah, [esi+1]
26513 00008A14 80FC20              <1>       cmp    ah, 20h
26514 00008A17 770A                <1>       ja     short pass_attr_space
26515 00008A19 0F82EAF0FFFF        <1>       jb     loc_cmd_failed
26516 00008A1F 46                  <1>       inc    esi
26517 00008A20 EBEF                <1>       jmp    short loc_attr_space
26518                              <1>
26519                              <1> loc_attr_file_nofilename_retn:
26520 00008A22 C3                  <1>       retn
26521                              <1>
26522                              <1> pass_attr_space:
26523 00008A23 80E4DF              <1>       and    ah, 0DFh
26524 00008A26 80FC53              <1>       cmp    ah, 'S'
26525 00008A29 0F87DAF0FFFF        <1>       ja     loc_cmd_failed
26526 00008A2F 7204                <1>       jb     short pass_attr_system
26527 00008A31 B404                <1>       mov    ah, 04h   ; System
26528 00008A33 EB21                <1>       jmp    short pass_attr_archive
26529                              <1>
26530                              <1> pass_attr_system:
26531 00008A35 80FC48              <1>       cmp    ah, 'H'
26532 00008A38 7706                <1>       ja     short pass_attr_hidden
26533 00008A3A 7213                <1>       jb     short pass_attr_read_only
26534 00008A3C B402                <1>       mov    ah, 02h    ; Hidden
26535 00008A3E EB16                <1>       jmp    short pass_attr_archive
26536                              <1>
26537                              <1> pass_attr_hidden:
26538 00008A40 80FC52              <1>       cmp    ah, 'R'
26539 00008A43 0F87C0F0FFFF        <1>       ja     loc_cmd_failed
26540 00008A49 7204                <1>       jb     short pass_attr_read_only ; Read only
26541 00008A4B B401                <1>       mov    ah, 01h
26542 00008A4D EB07                <1>       jmp    short pass_attr_archive
26543                              <1>
26544                              <1> pass_attr_read_only:
26545 00008A4F 80FC41              <1>       cmp    ah, 'A'
26546 00008A52 753B                <1>       jne    short loc_chk_attr_enter
26547 00008A54 B420                <1>       mov    ah, 20h    ; Archive
26548                              <1>
26549                              <1> pass_attr_archive:
26550 00008A56 3C2D                <1>       cmp    al, '-'
26551 00008A58 7508                <1>       jne    short pass_reducing_attributes
26552 00008A5A 0825[F40B0100]      <1>       or     [Attr_Chars], ah
26553 00008A60 EB06                <1>       jmp    short loc_change_attributes_inc
26554                              <1>
26555                              <1> pass_reducing_attributes:
26556 00008A62 0825[F50B0100]      <1>       or     [Attr_Chars+1], ah
26557                              <1>
26558                              <1> loc_change_attributes_inc:
26559 00008A68 46                  <1>       inc    esi
26560 00008A69 8A6601              <1>       mov    ah, [esi+1]
26561 00008A6C 80FC20              <1>       cmp    ah, 20h
26562 00008A6F 7227                <1>       jb     short pass_change_attr
26563 00008A71 74F5                <1>       je     short loc_change_attributes_inc
26564 00008A73 80FC2D              <1>       cmp    ah, '-'
26565 00008A76 770D                <1>       ja     short loc_chk_next_attr_char1
26566 00008A78 7405                <1>       je     short loc_chk_next_attr_char0
26567 00008A7A 80FC2B              <1>       cmp    ah, '+'
26568 00008A7D 7506                <1>       jne    short loc_chk_next_attr_char1
26569                              <1>
```

```
26570                                  <1> loc_chk_next_attr_char0:
26571 00008A7F 46                      <1>         inc    esi
26572 00008A80 668B06                  <1>         mov    ax, [esi]
26573 00008A83 EB9E                    <1>         jmp    short pass_attr_space
26574                                  <1>
26575                                  <1> loc_chk_next_attr_char1:
26576 00008A85 803E2D                  <1>         cmp    byte [esi], '-'
26577 00008A88 7799                    <1>         ja     short pass_attr_space
26578 00008A8A E988000000              <1>           jmp     loc_attr_file_check_fname_fchar
26579                                  <1>
26580                                  <1> loc_chk_attr_enter:
26581 00008A8F 80FC0D                  <1>         cmp    ah, 0Dh
26582 00008A92 0F8571F0FFFF            <1>         jne    loc_cmd_failed
26583                                  <1>
26584                                  <1> pass_change_attr:
26585 00008A98 A0[F40B0100]            <1>         mov    al, [Attr_Chars]
26586 00008A9D F6D0                    <1>         not    al
26587 00008A9F 2005[A05D0100]          <1>         and    [Attributes], al
26588 00008AA5 A0[F50B0100]            <1>         mov    al, [Attr_Chars+1]
26589 00008AAA 0805[A05D0100]          <1>         or     [Attributes], al
26590                                  <1>
26591                                  <1> loc_show_attributes:
26592 00008AB0 BE[6F130100]            <1>         mov    esi, nextline
26593 00008AB5 E8A3D8FFFF              <1>         call   print_msg
26594                                  <1>
26595                                  <1> loc_show_attributes_no_nextline:
26596 00008ABA C705[F40B0100]4E4F-      <1>         mov    dword [Attr_Chars], 'NORM'
26597 00008AC2 524D                    <1>
26598 00008AC4 66C705[F80B0100]41-      <1>         mov    word [Attr_Chars+4], 'AL'
26599 00008ACC 4C                      <1>
26600 00008ACD BE[F40B0100]            <1>         mov    esi, Attr_Chars
26601 00008AD2 A0[A05D0100]            <1>         mov    al, [Attributes]
26602 00008AD7 A804                    <1>         test   al, 04h
26603 00008AD9 7406                    <1>         jz     short pass_put_attr_s
26604 00008ADB 66C7065300              <1>         mov    word [esi], 0053h     ; S
26605 00008AE0 46                      <1>         inc    esi
26606                                  <1>
26607                                  <1> pass_put_attr_s:
26608 00008AE1 A802                    <1>         test   al, 02h
26609 00008AE3 7406                    <1>         jz     short pass_put_attr_h
26610 00008AE5 66C7064800              <1>         mov    word [esi], 0048h     ; H
26611 00008AEA 46                      <1>         inc    esi
26612                                  <1>
26613                                  <1> pass_put_attr_h:
26614 00008AEB A801                    <1>         test   al, 01h
26615 00008AED 7406                    <1>         jz     short pass_put_attr_r
26616 00008AEF 66C7065200              <1>         mov    word [esi], 0052h     ; R
26617 00008AF4 46                      <1>         inc    esi
26618                                  <1>
26619                                  <1> pass_put_attr_r:
26620 00008AF5 3C20                    <1>         cmp    al, 20h
26621 00008AF7 7205                    <1>         jb     short pass_put_attr_a
26622 00008AF9 66C7064100              <1>         mov    word [esi], 0041h     ; A
26623                                  <1>
26624                                  <1> pass_put_attr_a:
26625 00008AFE BE[E70B0100]            <1>         mov    esi, Str_Attributes
26626 00008B03 E855D8FFFF              <1>         call   print_msg
26627 00008B08 BE[6F130100]            <1>         mov    esi, nextline
26628 00008B0D E84BD8FFFF              <1>         call   print_msg
26629 00008B12 E924F9FFFF              <1>         jmp    loc_file_rw_restore_retn
26630                                  <1>
26631                                  <1> loc_attr_file_check_fname_fchar:
26632 00008B17 46                      <1>         inc    esi
26633 00008B18 803E20                  <1>         cmp    byte [esi], 20h
26634 00008B1B 74FA                    <1>         je     short loc_attr_file_check_fname_fchar
26635 00008B1D 0F8275FFFFFF            <1>           jb      pass_change_attr
26636                                  <1>
26637                                  <1> loc_attr_file_parse_path_name:
26638 00008B23 BF[8A5C0100]            <1>         mov    edi, FindFile_Drv
26639 00008B28 E804170000              <1>         call   parse_path_name
26640 00008B2D 0F82D6EFFFFF            <1>         jc     loc_cmd_failed
26641                                  <1>
26642                                  <1> loc_attr_file_check_filename_exists:
26643 00008B33 BE[CC5C0100]            <1>         mov    esi, FindFile_Name
26644 00008B38 803E20                  <1>         cmp    byte [esi], 20h
26645 00008B3B 0F86C8EFFFFF            <1>         jna    loc_cmd_failed
26646 00008B41 8935[485D0100]          <1>         mov    [DelFile_FNPointer], esi
26647                                  <1>
26648                                  <1> loc_attr_file_drv:
26649 00008B47 8A35[E6520100]          <1>         mov    dh, [Current_Drv]
26650 00008B4D 8835[465B0100]          <1>         mov    [RUN_CDRV], dh
26651                                  <1>
26652 00008B53 8A15[8A5C0100]          <1>         mov    dl, [FindFile_Drv]
26653 00008B59 38F2                    <1>         cmp    dl, dh
26654 00008B5B 740B                    <1>         je     short loc_attr_file_change_directory
26655                                  <1>
26656 00008B5D E80EE1FFFF              <1>         call   change_current_drive
26657 00008B62 0F82D3F8FFFF            <1>         jc     loc_file_rw_cmd_failed
26658                                  <1>
26659                                  <1> loc_attr_file_change_directory:
26660 00008B68 803D[8B5C0100]20        <1>           cmp     byte [FindFile_Directory], 20h
26661 00008B6F 7618                    <1>         jna    short loc_attr_file_find
26662                                  <1>
26663 00008B71 FE05[D3060100]          <1>         inc    byte [Restore_CDIR]
26664                                  <1>
26665 00008B77 BE[8B5C0100]            <1>         mov    esi, FindFile_Directory
26666 00008B7C 30E4                    <1>         xor    ah, ah ; CD_COMMAND sign -> 0
26667 00008B7E E898100000              <1>         call   change_current_directory
26668 00008B83 0F82B2F8FFFF            <1>         jc     loc_file_rw_cmd_failed
26669                                  <1>
26670                                  <1> ;loc_attr_file_change_prompt_dir_string:
26671                                  <1>         ;call  change_prompt_dir_string
26672                                  <1>
```

```
26673                                 <1> loc_attr_file_find:
26674                                 <1>       ;mov   esi, FindFile_Name
26675 00008B89 8B35[485D0100]         <1>       mov    esi, [DelFile_FNPointer]
26676 00008B8F 66B80008               <1>       mov    ax, 0800h ; Except volume labels
26677 00008B93 E873F4FFFF             <1>       call   find_first_file
26678 00008B98 0F829DF8FFFF           <1>       jc     loc_file_rw_cmd_failed
26679                                 <1>
26680                                 <1> loc_attr_file_ambgfn_check:
26681 00008B9E 6609D2                 <1>       or     dx, dx ; Ambiguous filename chars used sign (DX>0)
26682                                 <1>       ;      (Note: It was BX in TRDOS v1)
26683                                 <1>       ;jz    short loc_attr_file_found
26684 00008BA1 0F85AAFDFFFF           <1>         jnz     loc_file_not_found ; 06/03/2016
26685                                 <1>
26686                                 <1>       ;mov   eax, 2 ; File not found sign
26687                                 <1>       ;stc
26688                                 <1>       ;jmp   loc_file_rw_cmd_failed
26689                                 <1>
26690                                 <1> loc_attr_file_found:
26691                                 <1>       ; EDI = Directory buffer entry offset/address
26692                                 <1>       ; BL = File (or Directory) Attributes
26693                                 <1>       ;      (Note: It was 'CL' in TRDOS v1)
26694                                 <1>       ; mov  bl, [EDI+0Bh]
26695                                 <1>
26696 00008BA7 66833D[F40B0100]00     <1>       cmp    word [Attr_Chars], 0
26697 00008BAF 770B                   <1>       ja     short loc_attr_file_change_attributes
26698 00008BB1 881D[A05D0100]         <1>       mov    [Attributes], bl
26699 00008BB7 E9F4FEFFFF             <1>       jmp    loc_show_attributes
26700                                 <1>
26701                                 <1> loc_attr_file_change_attributes:
26702 00008BBC A0[F40B0100]           <1>       mov    al, [Attr_Chars]
26703 00008BC1 F6D0                   <1>       not    al
26704 00008BC3 20C3                   <1>       and    bl, al
26705 00008BC5 A0[F50B0100]           <1>       mov    al, [Attr_Chars+1]
26706 00008BCA 08C3                   <1>       or     bl, al
26707                                 <1>
26708 00008BCC 66817F0CA101           <1>       cmp    word [edi+DirEntry_NTRes], 01A1h ; Singlix FS
26709 00008BD2 741D                   <1>       je     short loc_attr_file_fs_check
26710                                 <1>
26711 00008BD4 881D[A05D0100]         <1>       mov    [Attributes], bl
26712 00008BDA 885F0B                 <1>       mov    [edi+0Bh], bl    ; Attributes (New!)
26713                                 <1>
26714                                 <1>       ; 04/03/2016
26715                                 <1>       ; TRDOS v1 has a bug here! it does not set
26716                                 <1>       ; 'DirBuff_ValidData' to 2; as result of this bug,
26717                                 <1>       ; 'save_directory_buffer' would not save the new attributes !
26718                                 <1>
26719 00008BDD C605[105B0100]02       <1>       mov    byte [DirBuff_ValidData], 2
26720                                 <1>
26721 00008BE4 E8AC1A0000             <1>       call   save_directory_buffer
26722 00008BE9 0F824CF8FFFF           <1>       jc     loc_file_rw_cmd_failed
26723                                 <1>
26724 00008BEF EB33                   <1>       jmp    short loc_print_attr_changed_message
26725                                 <1>
26726                                 <1> loc_attr_file_fs_check:
26727 00008BF1 29C0                   <1>       sub    eax, eax
26728 00008BF3 8A25[0E5B0100]         <1>         mov     ah, [DirBuff_DRV]
26729 00008BF9 BE00010900             <1>       mov    esi, Logical_DOSDisks
26730 00008BFE 01C6                   <1>         add     esi, eax
26731 00008C00 807E04A1               <1>         cmp     byte [esi+LD_FSType], 0A1h
26732 00008C04 7309                   <1>       jnc    short loc_attr_file_change_fs_file_attributes
26733 00008C06 66B80D00               <1>       mov    ax, 0Dh ; Invalid Data
26734 00008C0A E92CF8FFFF             <1>       jmp    loc_file_rw_cmd_failed
26735                                 <1>
26736                                 <1> loc_attr_file_change_fs_file_attributes:
26737                                 <1>       ; BL = New MS-DOS File Attributes
26738 00008C0F 88D8                   <1>       mov    al, bl ; File/Directory Attributes
26739 00008C11 30E4                   <1>       xor    ah, ah ; Attributes in MS-DOS format sign
26740 00008C13 E8A6050000             <1>       call   change_fs_file_attributes
26741 00008C18 0F821DF8FFFF           <1>       jc     loc_file_rw_cmd_failed
26742                                 <1>
26743 00008C1E 881D[A05D0100]         <1>       mov    [Attributes], bl
26744                                 <1>
26745                                 <1> loc_print_attr_changed_message:
26746 00008C24 BE[E20B0100]           <1>       mov    esi, Msg_New
26747 00008C29 E82FD7FFFF             <1>       call   print_msg
26748 00008C2E E987FEFFFF             <1>       jmp    loc_show_attributes_no_nextline
26749                                 <1>
26750                                 <1> rename_file:
26751                                 <1>       ; 13/11/2017
26752                                 <1>       ; 06/11/2016
26753                                 <1>       ; 05/11/2016
26754                                 <1>       ; 16/10/2016
26755                                 <1>       ; 08/03/2016
26756                                 <1>       ; 06/03/2016 (TRDOS 386 =  TRDOS v2.0)
26757                                 <1>       ; 20/11/2010 (TRDOS v1, CMD_INTR.ASM, 'cmp_cmd_rename')
26758                                 <1>       ; 16/11/2010
26759                                 <1>
26760                                 <1> get_rename_source_fchar:
26761                                 <1>       ; esi = file name
26762 00008C33 803E20                 <1>       cmp    byte [esi], 20h
26763 00008C36 7614                   <1>         jna  short loc_rename_nofilename_retn
26764                                 <1>
26765 00008C38 8935[C85D0100]         <1>       mov    [SourceFilePath], esi
26766                                 <1>
26767                                 <1> rename_scan_source_file:
26768 00008C3E 46                     <1>       inc    esi
26769 00008C3F 803E20                 <1>       cmp    byte [esi], 20h
26770 00008C42 7409                   <1>       je     short rename_scan_destination_file_1
26771                                 <1>       ;jb    short loc_rename_nofilename_retn
26772 00008C44 0F82BFEEFFFF           <1>       jb     loc_cmd_failed
26773 00008C4A EBF2                   <1>       jmp    short rename_scan_source_file
26774                                 <1>
26775                                 <1> loc_rename_nofilename_retn: ; 08/03/2016
```

```
26776 00008C4C C3                <1>       retn
26777                            <1>
26778                            <1> rename_scan_destination_file_1:
26779 00008C4D C60600            <1>       mov    byte [esi], 0
26780                            <1>
26781                            <1> rename_scan_destination_file_2:
26782 00008C50 46                <1>       inc    esi
26783 00008C51 803E20            <1>       cmp    byte [esi], 20h
26784 00008C54 74FA              <1>       je     short rename_scan_destination_file_2
26785                            <1>       ;jb    short loc_rename_nofilename_retn
26786 00008C56 0F82ADEEFFFF      <1>       jb     loc_cmd_failed
26787                            <1>
26788 00008C5C 8935[CC5D0100]    <1>       mov    [DestinationFilePath], esi
26789                            <1>
26790                            <1> rename_scan_destination_file_3:
26791 00008C62 46                <1>       inc    esi
26792 00008C63 803E20            <1>       cmp    byte [esi], 20h
26793 00008C66 77FA              <1>       ja     short rename_scan_destination_file_3
26794                            <1>
26795 00008C68 C60600            <1>       mov    byte [esi], 0
26796                            <1>
26797                            <1> loc_rename_save_current_drive:
26798 00008C6B 8A35[E6520100]    <1>       mov    dh, [Current_Drv]
26799 00008C71 8835[465B0100]    <1>       mov    byte [RUN_CDRV], dh
26800                            <1>
26801                            <1> loc_rename_sf_parse_path_name:
26802 00008C77 8B35[C85D0100]    <1>       mov    esi, [SourceFilePath]
26803 00008C7D BF[8A5C0100]      <1>       mov    edi, FindFile_Drv
26804 00008C82 E8AA150000        <1>       call   parse_path_name
26805 00008C87 0F827CEEFFFF      <1>       jc     loc_cmd_failed
26806                            <1>
26807                            <1> loc_rename_sf_check_filename_exists:
26808 00008C8D BE[CC5C0100]      <1>       mov    esi, FindFile_Name
26809 00008C92 803E20            <1>       cmp    byte [esi], 20h
26810 00008C95 0F866EEEFFFF      <1>       jna    loc_cmd_failed
26811                            <1>
26812                            <1>       ;mov   [DelFile_FNPointer], esi
26813                            <1>
26814                            <1> loc_rename_sf_drv:
26815                            <1>       ;mov   dh, [Current_Drv]
26816                            <1>       ;mov   [RUN_CDRV], dh
26817                            <1>
26818 00008C9B 8A15[8A5C0100]    <1>       mov    dl, [FindFile_Drv]
26819 00008CA1 38F2              <1>       cmp    dl, dh ; dh = [Current_Drv]
26820 00008CA3 740B              <1>       je     short rename_sf_change_directory
26821                            <1>
26822 00008CA5 E8C6DFFFFF        <1>       call   change_current_drive
26823 00008CAA 0F828BF7FFFF      <1>       jc     loc_file_rw_cmd_failed
26824                            <1>
26825                            <1> rename_sf_change_directory:
26826 00008CB0 803D[8B5C0100]20  <1>       cmp    byte [FindFile_Directory], 20h
26827 00008CB7 7618              <1>       jna    short rename_sf_find
26828                            <1>
26829 00008CB9 FE05[D3060100]    <1>       inc    byte [Restore_CDIR]
26830 00008CBF BE[8B5C0100]      <1>       mov    esi, FindFile_Directory
26831 00008CC4 30E4              <1>       xor    ah, ah ; CD_COMMAND sign -> 0
26832 00008CC6 E8500F0000        <1>       call   change_current_directory
26833 00008CCB 0F826AF7FFFF      <1>       jc     loc_file_rw_cmd_failed
26834                            <1>
26835                            <1> ;rename_sf_change_prompt_dir_string:
26836                            <1>       ;call  change_prompt_dir_string
26837                            <1>
26838                            <1> rename_sf_find:
26839                            <1>       ;mov   esi, [DelFile_FNPointer]
26840 00008CD1 BE[CC5C0100]      <1>       mov    esi, FindFile_Name
26841                            <1>
26842 00008CD6 66B80008          <1>       mov    ax, 0800h ; Except volume labels
26843 00008CDA E82CF3FFFF        <1>       call   find_first_file
26844 00008CDF 0F8256F7FFFF      <1>       jc     loc_file_rw_cmd_failed
26845                            <1>
26846                            <1> loc_rename_sf_ambgfn_check:
26847 00008CE5 6621D2            <1>       and    dx, dx ; Ambiguous filename chars used sign (DX>0)
26848                            <1>       ;      (Note: It was BX in TRDOS v1)
26849                            <1>       ;jz    short loc_rename_sf_found
26850 00008CE8 0F8563FCFFFF      <1>       jnz    loc_file_not_found
26851                            <1>
26852                            <1>       ;mov   eax, 2 ; File not found sign
26853                            <1>       ;stc
26854                            <1>       ;jmp   loc_file_rw_cmd_failed
26855                            <1>
26856                            <1> loc_rename_sf_found:
26857                            <1>       ; EDI = Directory buffer entry offset/address
26858                            <1>       ; BL = File (or Directory) Attributes
26859                            <1>       ;      (Note: It was 'CL' in TRDOS v1)
26860                            <1>       ; mov  bl, [EDI+0Bh]
26861                            <1>
26862 00008CEE F6C307            <1>       test   bl, 07h ; Attributes, S-H-R
26863 00008CF1 0F854EF7FFFF      <1>       jnz    loc_permission_denied
26864                            <1>
26865 00008CF7 BE[8A5C0100]      <1>         mov     esi, FindFile_Drv
26866 00008CFC BF[D05D0100]      <1>         mov     edi, SourceFile_Drv
26867 00008D01 B920000000        <1>       mov    ecx, 32
26868 00008D06 F3A5              <1>       rep    movsd
26869                            <1>
26870                            <1> loc_rename_df_parse_path_name:
26871 00008D08 8B35[CC5D0100]    <1>       mov    esi, [DestinationFilePath]
26872 00008D0E BF[8A5C0100]      <1>       mov    edi, FindFile_Drv
26873 00008D13 E819150000        <1>       call   parse_path_name
26874 00008D18 7219              <1>       jc     short loc_rename_df_cmd_failed
26875                            <1>
26876                            <1>       ;mov   dh, [RUN_CDRV]
26877 00008D1A 8A35[E6520100]    <1>       mov    dh, [Current_Drv]
26878                            <1>
```

```
                                <1>       ; 'rename' command is valid only for same dos drive and same dir!
                                <1>       ; ('move' command must be used if source file and destination file
                                <1>       ; directories are not same!)
00008D20 8A15[8A5C0100]         <1>       mov    dl, [FindFile_Drv]
00008D26 38F2                   <1>       cmp    dl, dh ; are source and destination drives different ?!
00008D28 7509                   <1>       jne    short loc_rename_df_cmd_failed ; yes!
                                <1>
                                <1> rename_df_check_dirname_exists:
00008D2A 803D[8B5C0100]00       <1>       cmp    byte [FindFile_Directory], 0
00008D31 760B                   <1>       jna    short rename_df_check_filename_exists
                                <1>
                                <1>       ; different source file and destination file directories !
                                <1> loc_rename_df_cmd_failed:
00008D33 B801000000             <1>       mov    eax, 1 ; TRDOS 'Bad command or file name' error
00008D38 F9                     <1>       stc
00008D39 E9FDF6FFFF             <1>       jmp    loc_file_rw_cmd_failed
                                <1>
                                <1> rename_df_check_filename_exists:
00008D3E BE[CC5C0100]           <1>       mov    esi, FindFile_Name
00008D43 E883F6FFFF             <1>       call   check_filename
00008D48 0F829FF7FFFF           <1>       jc     loc_mkdir_invalid_dir_name_chars
                                <1>
                                <1>       ;mov   [DelFile_FNPointer], esi
                                <1>       ;cmp   byte [esi], 20h
                                <1>       ;ja    short loc_rename_df_find
                                <1>
                                <1>       ;mov   dh, [Current_Drv] ; dh has not been changed
                                <1>
                                <1> rename_df_drv_check_writable:
00008D4E 0FB6F6                 <1>       movzx  esi, dh
                                <1>       ;movzx esi, byte [Current_Drv]
00008D51 81C600010900           <1>       add    esi, Logical_DOSDisks
                                <1>
00008D57 88F2                   <1>       mov    dl, dh ; dl = [Current_Drv]
00008D59 8A7601                 <1>       mov    dh, [esi+LD_DiskType]
                                <1>
00008D5C 80FE01                 <1>       cmp    dh, 1 ; 0 = Invalid
00008D5F 7310                   <1>       jnb    short rename_df_compare_sf_df_name
                                <1>
                                <1>       ; 16/10/2016 (13h -> 30)
00008D61 B81E000000             <1>       mov    eax, 30 ; 'Disk write-protected' error
00008D66 8B1D[CC5D0100]         <1>       mov    ebx, [DestinationFilePath]
00008D6C E9CAF6FFFF             <1>       jmp    loc_file_rw_cmd_failed
                                <1>
                                <1> rename_df_compare_sf_df_name:
00008D71 BE[CC5C0100]           <1>       mov    esi, FindFile_Name
00008D76 BF[125E0100]           <1>       mov    edi, SourceFile_Name
00008D7B B90C000000             <1>       mov    ecx, 12
                                <1> rename_df_compare_sf_df_name_next:
00008D80 AC                     <1>       lodsb
00008D81 AE                     <1>       scasb
00008D82 7506                   <1>       jne    short loc_rename_df_find
00008D84 08C0                   <1>       or     al, al
00008D86 74AB                   <1>       jz     short loc_rename_df_cmd_failed
00008D88 E2F6                   <1>       loop   rename_df_compare_sf_df_name_next
                                <1>
                                <1> loc_rename_df_find:
                                <1>       ;mov   esi, [DelFile_FNPointer]
00008D8A BE[CC5C0100]           <1>       mov    esi, FindFile_Name
                                <1>
00008D8F 6631C0                 <1>       xor    ax, ax ; Any
00008D92 E874F2FFFF             <1>       call   find_first_file
00008D97 730A                   <1>       jnc    short loc_rename_df_found
                                <1>
                                <1> loc_rename_df_check_error_code:
                                <1>       ;cmp   eax, 2
00008D99 3C02                   <1>       cmp    al, 2 ; Not found error
00008D9B 7411                   <1>       je     short rename_df_move_find_struct_to_dest
00008D9D F9                     <1>       stc
00008D9E E998F6FFFF             <1>       jmp loc_file_rw_cmd_failed
                                <1>
                                <1> loc_rename_df_found:
                                <1>       ; 05/11/2016
00008DA3 B805000000             <1>       mov    eax, 05h ; permission denied error
00008DA8 F9                     <1>       stc
00008DA9 E997F6FFFF             <1>       jmp    loc_permission_denied  ; 06/11/2016
                                <1>
                                <1> rename_df_move_find_struct_to_dest:
00008DAE BE[8A5C0100]           <1>         mov     esi, FindFile_Drv
00008DB3 BF[505E0100]           <1>         mov     edi, DestinationFile_Drv
00008DB8 B920000000             <1>       mov    ecx, 32
00008DBD F3A5                   <1>       rep    movsd
                                <1>
                                <1> loc_rename_df_process_q_sf:
                                <1>       ;mov   ecx, 12
00008DBF B10C                   <1>       mov    cl, 12
00008DC1 BE[125E0100]           <1>       mov    esi, SourceFile_Name
00008DC6 BF[230C0100]           <1>       mov    edi, Rename_OldName
                                <1> rename_df_process_q_nml_1_sf:
00008DCB AC                     <1>       lodsb
00008DCC 3C20                   <1>         cmp  al, 20h
00008DCE 7603                   <1>         jna  short rename_df_process_q_nml_2_sf
00008DD0 AA                     <1>       stosb
00008DD1 E2F8                   <1>       loop   rename_df_process_q_nml_1_sf
                                <1>
                                <1> rename_df_process_q_nml_2_sf:
00008DD3 C60700                 <1>       mov    byte [edi], 0
                                <1>
                                <1> loc_rename_df_process_q_df:
                                <1>       ;mov   ecx, 12
00008DD6 B10C                   <1>       mov    cl, 12
00008DD8 BE[925E0100]           <1>       mov    esi, DestinationFile_Name
00008DDD BF[340C0100]           <1>       mov    edi, Rename_NewName
```

```
26982                                  <1> rename_df_process_q_nml_1_df:
26983 00008DE2 AC                      <1>         lodsb
26984 00008DE3 3C20                    <1>         cmp     al, 20h
26985 00008DE5 7603                    <1>         jna     short loc_rename_df_process_q_nml_2_df
26986 00008DE7 AA                      <1>         stosb
26987 00008DE8 E2F8                    <1>         loop    rename_df_process_q_nml_1_df
26988                                  <1>
26989                                  <1> loc_rename_df_process_q_nml_2_df:
26990 00008DEA C60700                  <1>         mov     byte [edi], 0
26991                                  <1>
26992                                  <1> loc_rename_confirmation_question:
26993 00008DED BE[FB0B0100]            <1>         mov     esi, Msg_DoYouWantRename
26994 00008DF2 E866D5FFFF              <1>         call    print_msg
26995                                  <1>
26996 00008DF7 A0[2D5E0100]            <1>         mov     al, [SourceFile_DirEntry+11] ; Attributes
26997 00008DFC 2410                    <1>         and     al, 10h
26998 00008DFE 750C                    <1>         jnz     short rename_confirmation_question_dir
26999                                  <1>
27000                                  <1> rename_confirmation_question_file:
27001 00008E00 BE[120C0100]            <1>         mov     esi, Rename_File
27002 00008E05 E853D5FFFF              <1>         call    print_msg
27003 00008E0A EB0A                    <1>         jmp     short rename_confirmation_question_as
27004                                  <1>
27005                                  <1> rename_confirmation_question_dir:
27006 00008E0C BE[180C0100]            <1>         mov     esi, Rename_Directory
27007 00008E11 E847D5FFFF              <1>         call    print_msg
27008                                  <1>
27009                                  <1> rename_confirmation_question_as:
27010 00008E16 BE[230C0100]            <1>         mov     esi, Rename_OldName
27011 00008E1B E83DD5FFFF              <1>         call    print_msg
27012 00008E20 BE[300C0100]            <1>         mov     esi, Msg_File_rename_as
27013 00008E25 E833D5FFFF              <1>         call    print_msg
27014 00008E2A BE[570B0100]            <1>         mov     esi, Msg_YesNo
27015 00008E2F E829D5FFFF              <1>         call    print_msg
27016                                  <1>
27017                                  <1> loc_rename_ask_again:
27018 00008E34 30E4                    <1>         xor     ah, ah
27019 00008E36 E8DB7DFFFF              <1>         call    int16h
27020 00008E3B 3C1B                    <1>         cmp     al, 1Bh
27021 00008E3D 740F                    <1>         je      short loc_do_not_rename_file
27022 00008E3F 24DF                    <1>         and     al, 0DFh
27023 00008E41 A2[610B0100]            <1>         mov     [Y_N_nextline], al
27024 00008E46 3C59                    <1>         cmp     al, 'Y'
27025 00008E48 7404                    <1>         je      short loc_yes_rename_file
27026 00008E4A 3C4E                    <1>         cmp     al, 'N'
27027 00008E4C 75E6                    <1>         jne     short loc_rename_ask_again
27028                                  <1>
27029                                  <1> loc_do_not_rename_file:
27030                                  <1> loc_yes_rename_file:
27031 00008E4E A2[610B0100]            <1>         mov     [Y_N_nextline], al
27032 00008E53 6650                    <1>         push    ax
27033 00008E55 BE[610B0100]            <1>         mov     esi, Y_N_nextline
27034 00008E5A E8FED4FFFF              <1>         call    print_msg
27035 00008E5F 6658                    <1>         pop     ax
27036                                  <1>         ;cmp    al, 'Y' ; 'yes'
27037                                  <1>         ;cmc
27038                                  <1>           ;jnc loc_file_rw_restore_retn
27039 00008E61 3C4E                    <1>         cmp     al, 'N' ; 'no'
27040 00008E63 0F84D2F5FFFF            <1>           je    loc_file_rw_restore_retn
27041                                  <1>
27042 00008E69 BE[340C0100]            <1>         mov     esi, Rename_NewName
27043 00008E6E 668B0D[4A5E0100]        <1>         mov     cx, [SourceFile_DirEntryNumber]
27044 00008E75 66A1[365E0100]          <1>         mov     ax, [SourceFile_DirEntry+20] ; First Cluster, HW
27045 00008E7B C1E010                  <1>         shl     eax, 16 ; 13/11/2017
27046 00008E7E 66A1[3C5E0100]          <1>         mov     ax, [SourceFile_DirEntry+26] ; First Cluster, LW
27047                                  <1>
27048 00008E84 0FB61D[1F5E0100]        <1>         movzx   ebx, byte [SourceFile_LongNameEntryLength]
27049 00008E8B E8EB1B0000              <1>         call    rename_directory_entry
27050 00008E90 E9D4F6FFFF              <1>         jmp     loc_rename_file_ok
27051                                  <1> ;loc_rename_file_ok:
27052                                  <1> ;       jc      loc_run_cmd_failed
27053                                  <1> ;       mov     esi, Msg_OK
27054                                  <1> ;       call    proc_printmsg
27055                                  <1> ;       jmp     loc_file_rw_restore_retn
27056                                  <1>
27057                                  <1> move_file:
27058                                  <1>         ; 11/03/2016
27059                                  <1>         ; 09/03/2016
27060                                  <1>         ; 08/03/2016 (TRDOS 386 =  TRDOS v2.0)
27061                                  <1>         ; 21/05/2011 (TRDOS v1, CMD_INTR.ASM, 'cmp_cmd_move')
27062                                  <1>         ; 23/04/2011
27063                                  <1>
27064                                  <1> get_move_source_fchar:
27065                                  <1>         ; esi = file name
27066 00008E95 803E20                  <1>         cmp     byte [esi], 20h
27067 00008E98 7614                    <1>           jna   short loc_move_nofilename_retn
27068                                  <1>
27069 00008E9A 8935[C85D0100]          <1>         mov     [SourceFilePath], esi
27070                                  <1>
27071                                  <1> move_scan_source_file:
27072 00008EA0 46                      <1>         inc     esi
27073 00008EA1 803E20                  <1>         cmp     byte [esi], 20h
27074 00008EA4 7409                    <1>           je      short move_scan_destination_1
27075                                  <1>         ;jb     short loc_move_nofilename_retn
27076 00008EA6 0F825DECFFFF            <1>         jb      loc_cmd_failed
27077 00008EAC EBF2                    <1>         jmp     short move_scan_source_file
27078                                  <1>
27079                                  <1> loc_move_nofilename_retn:
27080 00008EAE C3                      <1>         retn
27081                                  <1>
27082                                  <1> move_scan_destination_1:
27083 00008EAF C60600                  <1>         mov     byte [esi], 0
27084                                  <1>
```

```
27085                                  <1> move_scan_destination_2:
27086 00008EB2 46                      <1>         inc    esi
27087 00008EB3 803E20                  <1>         cmp    byte [esi], 20h
27088 00008EB6 74FA                    <1>         je     short move_scan_destination_2
27089                                  <1>         ;jb    short loc_move_nofilename_retn
27090 00008EB8 0F824BECFFFF            <1>         jb     loc_cmd_failed
27091                                  <1>
27092 00008EBE 8935[CC5D0100]          <1>         mov    [DestinationFilePath], esi
27093                                  <1>
27094                                  <1> move_scan_destination_3:
27095 00008EC4 46                      <1>         inc    esi
27096 00008EC5 803E20                  <1>         cmp    byte [esi], 20h
27097 00008EC8 77FA                    <1>         ja     short move_scan_destination_3
27098 00008ECA C60600                  <1>         mov    byte [esi], 0
27099                                  <1>
27100                                  <1> loc_move_scan_destination_OK:
27101 00008ECD 8B35[C85D0100]          <1>         mov    esi, [SourceFilePath]
27102 00008ED3 8B3D[CC5D0100]          <1>         mov    edi, [DestinationFilePath]
27103                                  <1>
27104 00008ED9 B001                    <1>         mov    al, 1  ; move procedure Phase 1
27105 00008EDB E8171C0000              <1>         call   move_source_file_to_destination_file
27106 00008EE0 7328                    <1>         jnc    short move_source_file_to_destination_question
27107                                  <1>
27108                                  <1> loc_move_cmd_failed_1:
27109 00008EE2 08C0                    <1>         or     al, al
27110 00008EE4 0F841FECFFFF            <1>         jz     loc_cmd_failed
27111 00008EEA 3C11                    <1>         cmp    al, 11h
27112 00008EEC 740D                    <1>         je     short loc_msg_not_same_device
27113 00008EEE 3C05                    <1>         cmp    al, 05h
27114 00008EF0 0F853EECFFFF            <1>         jne    loc_run_cmd_failed
27115                                  <1>
27116 00008EF6 E94AF5FFFF              <1>         jmp    loc_permission_denied
27117                                  <1>
27118                                  <1>         ;mov   esi, Msg_Permission_denied
27119                                  <1>         ;call  print_msg
27120                                  <1>         ;jmp   loc_file_rw_restore_retn
27121                                  <1>
27122                                  <1> loc_msg_not_same_device:
27123 00008EFB BE[410C0100]            <1>         mov    esi, msg_not_same_drv
27124 00008F00 E858D4FFFF              <1>         call   print_msg
27125 00008F05 E931F5FFFF              <1>         jmp    loc_file_rw_restore_retn
27126                                  <1>
27127                                  <1> move_source_file_to_destination_question:
27128 00008F0A A0[D05D0100]            <1>           mov     al, [SourceFile_Drv]
27129 00008F0F 0441                    <1>         add    al, 'A'
27130 00008F11 A2[A30C0100]            <1>         mov    [msg_source_file_drv], al
27131 00008F16 A0[505E0100]            <1>           mov     al, [DestinationFile_Drv]
27132 00008F1B 0441                    <1>         add    al, 'A'
27133 00008F1D A2[C20C0100]            <1>         mov    [msg_destination_file_drv], al
27134                                  <1>
27135 00008F22 57                      <1>         push   edi ; *
27136                                  <1>
27137 00008F23 BE[870C0100]            <1>         mov    esi, msg_source_file
27138 00008F28 E830D4FFFF              <1>         call   print_msg
27139 00008F2D BE[D15D0100]            <1>         mov    esi, SourceFile_Directory
27140 00008F32 803E20                  <1>         cmp    byte [esi], 20h
27141 00008F35 7605                    <1>         jna    short msftdfq_sfn
27142 00008F37 E821D4FFFF              <1>         call   print_msg
27143                                  <1> msftdfq_sfn:
27144 00008F3C BE[125E0100]            <1>         mov    esi, SourceFile_Name
27145 00008F41 E817D4FFFF              <1>         call   print_msg
27146 00008F46 BE[A60C0100]            <1>         mov    esi, msg_destination_file
27147 00008F4B E80DD4FFFF              <1>         call   print_msg
27148 00008F50 BE[515E0100]            <1>         mov    esi, DestinationFile_Directory
27149 00008F55 803E20                  <1>         cmp    byte [esi], 20h
27150 00008F58 7605                    <1>         jna    short msftdfq_dfn
27151 00008F5A E8FED3FFFF              <1>         call   print_msg
27152                                  <1> msftdfq_dfn:
27153 00008F5F BE[925E0100]            <1>         mov    esi, DestinationFile_Name
27154 00008F64 E8F4D3FFFF              <1>         call   print_msg
27155 00008F69 BE[C50C0100]            <1>         mov    esi, msg_copy_nextline
27156 00008F6E E8EAD3FFFF              <1>         call   print_msg
27157 00008F73 BE[C50C0100]            <1>         mov    esi, msg_copy_nextline
27158 00008F78 E8E0D3FFFF              <1>         call   print_msg
27159                                  <1>
27160                                  <1> loc_move_ask_for_new_file_yes_no:
27161 00008F7D BE[530C0100]            <1>         mov    esi, Msg_DoYouWantMoveFile
27162 00008F82 E8D6D3FFFF              <1>         call   print_msg
27163 00008F87 BE[570B0100]            <1>         mov    esi, Msg_YesNo
27164 00008F8C E8CCD3FFFF              <1>         call   print_msg
27165                                  <1> loc_move_ask_for_new_file_again:
27166 00008F91 30E4                    <1>         xor    ah, ah
27167 00008F93 E87E7CFFFF              <1>         call   int16h
27168 00008F98 3C1B                    <1>         cmp    al, 1Bh
27169                                  <1>         ;je    short loc_do_not_move_file
27170 00008F9A 744F                    <1>         je     short loc_move_y_n_escape
27171 00008F9C 24DF                    <1>         and    al, 0DFh
27172 00008F9E A2[610B0100]            <1>           mov     [Y_N_nextline], al
27173 00008FA3 3C59                    <1>         cmp    al, 'Y'
27174 00008FA5 7404                    <1>         je     short loc_yes_move_file
27175 00008FA7 3C4E                    <1>         cmp    al, 'N'
27176 00008FA9 75E6                    <1>         jne    short loc_move_ask_for_new_file_again
27177                                  <1>
27178                                  <1> loc_do_not_move_file:
27179                                  <1> loc_yes_move_file:
27180 00008FAB A2[610B0100]            <1>         mov    [Y_N_nextline], al
27181 00008FB0 6650                    <1>         push   ax
27182 00008FB2 BE[610B0100]            <1>         mov    esi, Y_N_nextline
27183 00008FB7 E8A1D3FFFF              <1>         call   print_msg
27184 00008FBC 6658                    <1>         pop    ax
27185 00008FBE 5F                      <1>         pop    edi ; *
27186                                  <1>         ;cmp   al, 'Y' ; 'yes'
27187                                  <1>         ;cmc
```

```
                                <1>         ;jnc loc_file_rw_restore_retn
00008FBF 3C4E                   <1>       cmp   al, 'N' ; 'no'
00008FC1 0F8474F4FFFF           <1>         je  loc_file_rw_restore_retn
                                <1>
                                <1> loc_move_yes_move_file:
00008FC7 B002                   <1>       mov   al, 2 ; move procedure Phase 2
00008FC9 E8291B0000             <1>       call  move_source_file_to_destination_file
                                <1>       ;jc   short loc_move_cmd_failed_2
00008FCE 0F839BF5FFFF           <1>         jnc     move_source_file_to_destination_OK
                                <1>
                                <1> ;move_source_file_to_destination_OK:
                                <1> ;     mov   esi, Msg_OK
                                <1> ;     call  print_msg
                                <1> ;     jmp   loc_file_rw_restore_retn
                                <1>
                                <1> loc_move_cmd_failed_2:
00008FD4 3C27                   <1>       cmp   al, 27h
00008FD6 0F8558EBFFFF           <1>       jne   loc_run_cmd_failed
                                <1>
00008FDC BE[6C0C0100]           <1>       mov   esi, msg_insufficient_disk_space
00008FE1 E877D3FFFF             <1>       call  print_msg
                                <1>
00008FE6 E950F4FFFF             <1>       jmp   loc_file_rw_restore_retn
                                <1>
                                <1> loc_move_y_n_escape:
00008FEB B04E                   <1>       mov   al, 'N' ; 'no'
00008FED EBBC                   <1>       jmp   short loc_do_not_move_file
                                <1>
                                <1> copy_file:
                                <1>       ; 15/10/2016
                                <1>       ; 24/03/2016
                                <1>       ; 21/03/2016
                                <1>       ; 15/03/2016 (TRDOS 386 =  TRDOS v2.0)
                                <1>       ; 21/05/2011 (TRDOS v1, CMD_INTR.ASM, 'cmp_cmd_copy')
                                <1>       ; 01/08/2010
                                <1>
                                <1> get_copy_source_fchar:
                                <1>       ; esi = file name
00008FEF 803E20                 <1>       cmp   byte [esi], 20h
00008FF2 7614                   <1>         jna     short loc_copy_nofilename_retn
                                <1>
00008FF4 8935[C85D0100]         <1>       mov   [SourceFilePath], esi
                                <1>
                                <1> copy_scan_source_file:
00008FFA 46                     <1>       inc   esi
00008FFB 803E20                 <1>       cmp   byte [esi], 20h
00008FFE 7409                   <1>       je    short copy_scan_destination_1
                                <1>       ;jb   short loc_copy_nofilename_retn
00009000 0F8203EBFFFF           <1>       jb    loc_cmd_failed
00009006 EBF2                   <1>       jmp   short copy_scan_source_file
                                <1>
                                <1> loc_copy_nofilename_retn:
00009008 C3                     <1>       retn
                                <1>
                                <1> copy_scan_destination_1:
00009009 C60600                 <1>       mov   byte [esi], 0
                                <1>
                                <1> copy_scan_destination_2:
0000900C 46                     <1>       inc   esi
0000900D 803E20                 <1>       cmp   byte [esi], 20h
00009010 74FA                   <1>       je    short copy_scan_destination_2
                                <1>       ;jb   short loc_copy_nofilename_retn
00009012 0F82F1EAFFFF           <1>       jb    loc_cmd_failed
                                <1>
00009018 8935[CC5D0100]         <1>       mov   [DestinationFilePath], esi
                                <1>
                                <1> copy_scan_destination_3:
0000901E 46                     <1>       inc   esi
0000901F 803E20                 <1>       cmp   byte [esi], 20h
00009022 77FA                   <1>       ja    short copy_scan_destination_3
00009024 C60600                 <1>       mov   byte [esi], 0
                                <1>
                                <1> loc_copy_save_current_drive:
00009027 8A35[E6520100]         <1>       mov   dh, [Current_Drv]
0000902D 8835[465B0100]         <1>       mov   [RUN_CDRV], dh
                                <1>
                                <1> copy_source_file_to_destination_phase_1:
00009033 8B35[C85D0100]         <1>       mov   esi, [SourceFilePath]
00009039 8B3D[CC5D0100]         <1>       mov   edi, [DestinationFilePath]
                                <1>
0000903F B001                   <1>       mov   al, 1  ; copy procedure Phase 1
00009041 E84E1D0000             <1>       call  copy_source_file_to_destination_file
00009046 732B                   <1>       jnc   short copy_source_file_to_destination_question
                                <1>
                                <1> loc_copy_cmd_failed_1:
                                <1>       ; 18/03/2016 (restore current drive and directory)
00009048 08C0                   <1>       or    al, al
0000904A 7507                   <1>       jnz   short loc_copy_cmd_failed_2
                                <1>
0000904C FEC0                   <1>         inc     al ; mov al, 1 ; Bad command or file name !
0000904E E9E1EAFFFF             <1>       jmp   loc_run_cmd_failed
                                <1>
                                <1> loc_copy_cmd_failed_2:
00009053 3C27                   <1>       cmp   al, 27h ; Insufficient disk space
00009055 740D                   <1>       je    short loc_file_write_insuff_disk_space_msg
                                <1>
00009057 3C05                   <1>       cmp   al, 05h
00009059 0F85D5EAFFFF           <1>       jne   loc_run_cmd_failed
                                <1>
0000905F E9E1F3FFFF             <1>       jmp   loc_permission_denied
                                <1>
                                <1> loc_file_write_insuff_disk_space_msg:
00009064 BE[6C0C0100]           <1>       mov   esi, msg_insufficient_disk_space
```

```
27291 00009069 E8EFD2FFFF          <1>       call   print_msg
27292 0000906E E9C8F3FFFF          <1>         jmp     loc_file_rw_restore_retn
27293                              <1>
27294                              <1> copy_source_file_to_destination_question:
27295 00009073 57                  <1>       push   edi ; *
27296                              <1>
27297                              <1>       ; dh = source file attributes
27298                              <1>       ; dl > 0 -> destination file found
27299 00009074 20D2                <1>       and    dl, dl
27300 00009076 7449                <1>       jz     short copy_source_file_to_destination_pass_owrq
27301                              <1>
27302                              <1> loc_copy_ask_for_owr_yes_no:
27303 00009078 BE[C80C0100]        <1>       mov    esi, Msg_DoYouWantOverWriteFile
27304 0000907D E8DBD2FFFF          <1>       call   print_msg
27305 00009082 BE[925E0100]        <1>       mov    esi, DestinationFile_Name
27306 00009087 E8D1D2FFFF          <1>       call   print_msg
27307 0000908C BE[570B0100]        <1>       mov    esi, Msg_YesNo
27308 00009091 E8C7D2FFFF          <1>       call   print_msg
27309                              <1>
27310                              <1> loc_copy_ask_for_owr_again:
27311 00009096 30E4                <1>       xor    ah, ah
27312 00009098 E8797BFFFF          <1>       call   int16h
27313 0000909D 3C1B                <1>       cmp    al, 1Bh
27314                              <1>         ;je     loc_do_not_copy_file
27315 0000909F 7419                <1>         je      short loc_copy_y_n_escape
27316 000090A1 24DF                <1>       and    al, 0DFh
27317 000090A3 A2[610B0100]        <1>         mov     [Y_N_nextline], al
27318 000090A8 3C59                <1>       cmp    al, 'Y'
27319 000090AA 0F84B1000000        <1>         je      loc_yes_copy_file
27320 000090B0 3C4E                <1>       cmp    al, 'N'
27321 000090B2 0F84A9000000        <1>         je      loc_do_not_copy_file
27322 000090B8 EBDC                <1>       jmp    short loc_copy_ask_for_owr_again
27323                              <1>
27324                              <1> loc_copy_y_n_escape:
27325 000090BA B04E                <1>       mov    al, 'N' ; 'no'
27326 000090BC E9A0000000          <1>         jmp     loc_do_not_copy_file
27327                              <1>
27328                              <1> copy_source_file_to_destination_pass_owrq:
27329 000090C1 A0[D05D0100]        <1>       mov     al, [SourceFile_Drv]
27330 000090C6 0441                <1>       add    al, 'A'
27331 000090C8 A2[A30C0100]        <1>       mov    [msg_source_file_drv], al
27332 000090CD A0[505E0100]        <1>         mov     al, [DestinationFile_Drv]
27333 000090D2 0441                <1>       add    al, 'A'
27334 000090D4 A2[C20C0100]        <1>       mov    [msg_destination_file_drv], al
27335                              <1>
27336 000090D9 BE[870C0100]        <1>       mov    esi, msg_source_file
27337 000090DE E87AD2FFFF          <1>       call   print_msg
27338 000090E3 BE[D15D0100]        <1>       mov    esi, SourceFile_Directory
27339 000090E8 803E20              <1>       cmp    byte [esi], 20h
27340 000090EB 7605                <1>       jna    short csftdfq_sfn
27341 000090ED E86BD2FFFF          <1>       call   print_msg
27342                              <1> csftdfq_sfn:
27343 000090F2 BE[125E0100]        <1>       mov    esi, SourceFile_Name
27344 000090F7 E861D2FFFF          <1>       call   print_msg
27345 000090FC BE[A60C0100]        <1>       mov    esi, msg_destination_file
27346 00009101 E857D2FFFF          <1>       call   print_msg
27347 00009106 BE[515E0100]        <1>       mov    esi, DestinationFile_Directory
27348 0000910B 803E20              <1>       cmp    byte [esi], 20h
27349 0000910E 7605                <1>       jna    short csftdfq_dfn
27350 00009110 E848D2FFFF          <1>       call   print_msg
27351                              <1> csftdfq_dfn:
27352 00009115 BE[925E0100]        <1>       mov    esi, DestinationFile_Name
27353 0000911A E83ED2FFFF          <1>       call   print_msg
27354 0000911F BE[C50C0100]        <1>       mov    esi, msg_copy_nextline
27355 00009124 E834D2FFFF          <1>       call   print_msg
27356 00009129 BE[C50C0100]        <1>       mov    esi, msg_copy_nextline
27357 0000912E E82AD2FFFF          <1>       call   print_msg
27358                              <1>
27359                              <1> loc_copy_ask_for_new_file_yes_no:
27360 00009133 BE[E70C0100]        <1>       mov    esi, Msg_DoYouWantCopyFile
27361 00009138 E820D2FFFF          <1>       call   print_msg
27362 0000913D BE[570B0100]        <1>       mov    esi, Msg_YesNo
27363 00009142 E816D2FFFF          <1>       call   print_msg
27364                              <1>
27365                              <1> loc_copy_ask_for_new_file_again:
27366 00009147 30E4                <1>       xor    ah, ah
27367 00009149 E8C87AFFFF          <1>       call   int16h
27368 0000914E 3C1B                <1>       cmp    al, 1Bh
27369 00009150 740F                <1>       je     short loc_do_not_copy_file
27370 00009152 24DF                <1>       and    al, 0DFh
27371 00009154 A2[610B0100]        <1>         mov     [Y_N_nextline], al
27372 00009159 3C59                <1>       cmp    al, 'Y'
27373 0000915B 7404                <1>       je     short loc_yes_copy_file
27374 0000915D 3C4E                <1>       cmp    al, 'N'
27375 0000915F 75E6                <1>       jne    short loc_copy_ask_for_new_file_again
27376                              <1>
27377                              <1> loc_do_not_copy_file:
27378                              <1> loc_yes_copy_file:
27379 00009161 A2[610B0100]        <1>       mov    [Y_N_nextline], al
27380 00009166 6650                <1>       push   ax
27381 00009168 BE[610B0100]        <1>       mov    esi, Y_N_nextline
27382 0000916D E8EBD1FFFF          <1>       call   print_msg
27383 00009172 6658                <1>       pop    ax
27384 00009174 5F                  <1>       pop    edi ; *
27385                              <1>       ;cmp   al, 'Y' ; 'yes'
27386                              <1>       ;cmc
27387                              <1>         ;jnc loc_file_rw_restore_retn
27388 00009175 3C4E                <1>       cmp    al, 'N' ; 'no'
27389 00009177 0F84BEF2FFFF        <1>         je   loc_file_rw_restore_retn
27390                              <1>
27391                              <1> copy_source_file_to_destination_pass_q:
27392 0000917D B002                <1>       mov    al, 2  ; copy procedure Phase 2
27393 0000917F E8101C0000          <1>       call   copy_source_file_to_destination_file
```

```
27394                                   <1>       ;jc    short loc_file_write_check_disk_space_err
27395                                   <1>
27396                                   <1>       ; 24/03/2016
27397 00009184 6651                     <1>       push   cx
27398 00009186 BE[C50C0100]             <1>       mov    esi, msg_copy_nextline
27399 0000918B E8CDD1FFFF               <1>       call   print_msg
27400                                   <1>       ;pop   cx
27401 00009190 6658                     <1>       pop    ax
27402                                   <1>
27403                                   <1>       ;or    cl, cl
27404 00009192 08C0                     <1>       or     al, al
27405 00009194 7419                     <1>       jz     short copy_source_file_to_destination_OK
27406                                   <1>
27407                                   <1>       ; 15/10/2016 (1Dh -> 18)
27408                                   <1>       ; 18/03/2016 (1Dh)
27409                                   <1>       ;cmp   cl, 18 ; write error
27410 00009196 3C12                     <1>       cmp    al, 18
27411 00009198 7506                     <1>       jne    short copy_source_file_to_destination_not_OK
27412                                   <1>       ;
27413                                   <1>       ;mov   al, cl ; error number (write fault!)
27414 0000919A F9                       <1>       stc
27415 0000919B E99BF2FFFF               <1>       jmp    loc_file_rw_cmd_failed
27416                                   <1>
27417                                   <1> copy_source_file_to_destination_not_OK:
27418 000091A0 BE[000D0100]             <1>       mov    esi, Msg_read_file_error_before_EOF
27419 000091A5 E8B3D1FFFF               <1>       call   print_msg
27420 000091AA E98CF2FFFF               <1>       jmp    loc_file_rw_restore_retn
27421                                   <1>
27422                                   <1> copy_source_file_to_destination_OK:
27423 000091AF BE[650B0100]             <1>       mov    esi, Msg_OK
27424 000091B4 E8A4D1FFFF               <1>       call   print_msg
27425                                   <1>
27426 000091B9 E97DF2FFFF               <1>       jmp    loc_file_rw_restore_retn
27427                                   <1>
27428                                   <1> ;loc_file_write_check_disk_space_err:
27429                                   <1>       ;cmp   al, 27h ; Insufficient disk space
27430                                   <1>       ;je    loc_file_write_insuff_disk_space_msg
27431                                   <1>         ;jb  loc_file_rw_cmd_failed
27432                                   <1>
27433                                   <1>       ;call  print_misc_error_msg ; 15/03/2016
27434                                   <1>         ;jmp loc_file_rw_restore_retn
27435                                   <1>
27436                                   <1> change_fs_file_attributes:
27437                                   <1>       ; 04/03/2016 ; Temporary
27438                                   <1>       ; AL = File or directory attributes
27439                                   <1>       ; AH = 0 -> Attributes are in MS-DOS format
27440                                   <1>       ; AH > 0 -> Attributes are in SINGLIX format
27441                                   <1>       ;push  ebx
27442                                   <1>       ; ... do somethings here ...
27443                                   <1>       ;pop   ebx
27444                                   <1>       ; BL = File or directory attributes
27445 000091BE C3                       <1>       retn
27446                                   <1>
27447                                   <1> set_get_env:
27448                                   <1>       ; 11/04/2016 (TRDOS 386 =  TRDOS v2.0)
27449                                   <1>       ; 02/09/2011 (TRDOS v1, CMD_INTR.ASM, 'cmp_cmd_set')
27450                                   <1>       ; 2005 - 28/08/2011
27451                                   <1> get_setenv_fchar:
27452                                   <1>       ; esi = environment variable/string
27453 000091BF 8A06                     <1>       mov    al, [esi]
27454 000091C1 3C20                     <1>       cmp    al, 20h
27455 000091C3 771E                     <1>       ja     short loc_find_env
27456                                   <1>
27457 000091C5 BE00300900               <1>       mov    esi, Env_Page
27458                                   <1> loc_print_setline:
27459 000091CA 803E00                   <1>       cmp    byte [esi], 0
27460 000091CD 7613                     <1>       jna    short loc_setenv_retn
27461 000091CF E889D1FFFF               <1>       call   print_msg
27462 000091D4 56                       <1>       push   esi
27463 000091D5 BE[6F130100]             <1>       mov    esi, nextline
27464 000091DA E87ED1FFFF               <1>       call   print_msg
27465 000091DF 5E                       <1>       pop    esi
27466 000091E0 EBE8                     <1>       jmp    short loc_print_setline
27467                                   <1>
27468                                   <1> loc_setenv_retn:
27469 000091E2 C3                       <1>       retn
27470                                   <1>
27471                                   <1> loc_find_env:
27472 000091E3 3C3D                     <1>       cmp    al, '='
27473 000091E5 0F841EE9FFFF             <1>       je     loc_cmd_failed
27474                                   <1>
27475 000091EB 56                       <1>       push   esi
27476                                   <1> loc_repeat_env_equal_check:
27477 000091EC 46                       <1>       inc    esi
27478 000091ED 803E3D                   <1>       cmp    byte [esi], '='
27479 000091F0 7431                     <1>       je     short pass_env_equal_check
27480 000091F2 803E20                   <1>       cmp    byte [esi], 20h
27481 000091F5 73F5                     <1>       jnb    short loc_repeat_env_equal_check
27482 000091F7 C60600                   <1>       mov    byte [esi], 0
27483 000091FA 5E                       <1>       pop    esi
27484 000091FB BF[E6530100]             <1>       mov    edi, TextBuffer ; out buffer
27485 00009200 B9FF000000               <1>       mov    ecx, 255 ; maximum size (limit)
27486 00009205 30C0                     <1>       xor    al, al ; 0 -> use [ESI]
27487 00009207 E89E000000               <1>       call   get_environment_string
27488 0000920C 72D4                     <1>       jc     short loc_setenv_retn
27489                                   <1>
27490 0000920E BE[E6530100]             <1>       mov    esi, TextBuffer
27491 00009213 E845D1FFFF               <1>       call   print_msg
27492 00009218 BE[6F130100]             <1>       mov    esi, nextline
27493 0000921D E83BD1FFFF               <1>       call   print_msg
27494                                   <1>
27495 00009222 C3                       <1>       retn
27496                                   <1>
```

```
                                <1> pass_env_equal_check:
00009223 46                     <1>         inc    esi
00009224 803E20                 <1>         cmp    byte [esi], 20h
00009227 73FA                   <1>         jnb    short pass_env_equal_check
00009229 C60600                 <1>         mov    byte [esi], 0
                                <1>
                                <1> loc_call_set_env_string:
0000922C 5E                     <1>         pop    esi
0000922D E83B010000             <1>         call   set_environment_string
00009232 73AE                   <1>         jnc    short loc_setenv_retn
                                <1>
                                <1> loc_set_cmd_failed:
00009234 3C08                   <1>         cmp    al, 08h
00009236 0F85CDE8FFFF           <1>         jne    loc_cmd_failed
                                <1>
0000923C BE[400D0100]           <1>         mov    esi, Msg_No_Set_Space
00009241 E817D1FFFF             <1>         call   print_msg
                                <1>
00009246 C3                     <1>         retn
                                <1>
                                <1> set_get_path:
                                <1>         ; 11/04/2016 (TRDOS 386 =  TRDOS v2.0)
                                <1>         ; 03/09/2011 (TRDOS v1, CMD_INTR.ASM, 'cmp_cmd_path')
                                <1>         ; 2005
                                <1> get_path_fchar:
                                <1>         ; esi = path
00009247 803E20                 <1>         cmp    byte [esi], 20h
0000924A 7737                   <1>         ja     short loc_set_path
                                <1>
0000924C BE00300900             <1>         mov    esi, Env_Page
                                <1> loc_print_path:
00009251 803E00                 <1>         cmp    byte [esi], 0
00009254 762C                   <1>         jna    short loc_path_retn
                                <1>
00009256 BE[9F070100]           <1>         mov    esi, Cmd_Path ; 'PATH' address
0000925B BF[E6530100]           <1>         mov    edi, TextBuffer ; oout buffer
00009260 30C0                   <1>         xor    al, al  ; use [ESI]
00009262 B9FF000000             <1>         mov    ecx, 255 ; maximum size (limit)
00009267 E83E000000             <1>         call   get_environment_string
0000926C 7214                   <1>         jc     short loc_path_retn
                                <1>
0000926E BE[E6530100]           <1>         mov    esi, TextBuffer
00009273 E8E5D0FFFF             <1>         call   print_msg
00009278 BE[6F130100]           <1>         mov    esi, nextline
0000927D E8DBD0FFFF             <1>         call   print_msg
                                <1>
                                <1> loc_path_retn:
00009282 C3                     <1>         retn
                                <1>
                                <1> loc_set_path:
00009283 56                     <1>         push   esi
                                <1> loc_set_path_find_end:
00009284 46                     <1>         inc    esi
00009285 803E20                 <1>         cmp    byte [esi], 20h
00009288 73FA                   <1>         jnb    short loc_set_path_find_end
0000928A C60600                 <1>         mov    byte [esi], 0
                                <1> loc_set_path_header:
0000928D 5E                     <1>         pop    esi
0000928E 4E                     <1>         dec    esi
0000928F C6063D                 <1>         mov    byte [esi], '='
00009292 4E                     <1>         dec    esi
00009293 C60648                 <1>         mov    byte [esi], 'H'
00009296 4E                     <1>         dec    esi
00009297 C60654                 <1>         mov    byte [esi], 'T'
0000929A 4E                     <1>         dec    esi
0000929B C60641                 <1>         mov    byte [esi], 'A'
0000929E 4E                     <1>         dec    esi
0000929F C60650                 <1>         mov    byte [esi], 'P'
                                <1>
                                <1> loc_path_call_set_env_string:
000092A2 E8C6000000             <1>         call   set_environment_string
000092A7 728B                   <1>         jc   short loc_set_cmd_failed
                                <1>
000092A9 C3                     <1>       retn
                                <1>
                                <1> get_environment_string:
                                <1>       ; 12/04/2016
                                <1>       ; 11/04/2016
                                <1>       ; 05/04/2016 (TRDOS 386 =  TRDOS v2.0)
                                <1>       ; 02/09/2011 (TRDOS v1, MAINPROG.ASM)
                                <1>       ; 28/08/2011
                                <1>       ; INPUT->
                                <1>       ;     EDI = Output buffer
                                <1>       ;     CX = Buffer length (<= ENV_PAGE_SIZE)
                                <1>       ;
                                <1>       ;     AL > 0 = AL = String sequence number
                                <1>       ;     AL = 0 -> ESI = ASCIIZ Set word
                                <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>
000092AA BA00300900             <1>       mov    edx, Env_Page
000092AF 803A00                 <1>       cmp    byte [edx], 0
000092B2 7474                   <1>       jz     short get_env_string_with_word_stc_retn
                                <1>
000092B4 66890D[545F0100]       <1>       mov    [env_var_length], cx
                                <1>
```

```
27600 000092BB 51                  <1>       push   ecx ; *
27601 000092BC 56                  <1>       push   esi ; **
27602                              <1>
27603 000092BD 08C0                <1>       or     al, al
27604 000092BF 7449                <1>       jz     short get_env_string_with_word
27605                              <1>
27606                              <1> get_env_string_with_seq_number:
27607 000092C1 B101                <1>       mov    cl, 1
27608 000092C3 88C5                <1>       mov    ch, al
27609 000092C5 31C0                <1>       xor    eax, eax
27610 000092C7 89D6                <1>       mov    esi, edx ; Env_Page
27611                              <1>
27612                              <1> get_env_string_seq_number_check:
27613 000092C9 38CD                <1>       cmp    ch, cl
27614 000092CB 7726                <1>       ja     short get_env_string_seq_number_next
27615                              <1>
27616                              <1> get_env_string_move_to_buff:
27617 000092CD 57                  <1>       push   edi ; ***
27618                              <1>
27619 000092CE 29D2                <1>       sub    edx, edx
27620                              <1>
27621                              <1> get_env_string_seq_number_repeat1:
27622 000092D0 42                  <1>       inc    edx
27623 000092D1 AC                  <1>       lodsb
27624 000092D2 AA                  <1>       stosb
27625                              <1>
27626 000092D3 66FF0D[545F0100]    <1>       dec    word [env_var_length]
27627 000092DA 7508                <1>       jnz    short get_env_string_seq_number_repeat3
27628                              <1>
27629                              <1> get_env_string_seq_number_repeat2:
27630 000092DC 20C0                <1>       and    al, al
27631 000092DE 7408                <1>       jz     short get_env_string_seq_number_ok
27632 000092E0 42                  <1>       inc    edx
27633 000092E1 AC                  <1>       lodsb
27634 000092E2 EBF8                <1>       jmp    short get_env_string_seq_number_repeat2
27635                              <1>
27636                              <1> get_env_string_seq_number_repeat3:
27637 000092E4 08C0                <1>       or     al, al
27638 000092E6 75E8                <1>       jnz    short get_env_string_seq_number_repeat1
27639                              <1>
27640                              <1> get_env_string_seq_number_ok:
27641 000092E8 5F                  <1>       pop    edi ; ***
27642 000092E9 89D0                <1>       mov    eax, edx ; Length of the environment string
27643                              <1>                       ; (ASCIIZ, includes ZERO tail)
27644 000092EB BA00300900          <1>       mov    edx, Env_Page
27645                              <1>
27646                              <1> get_env_string_stc_retn:
27647 000092F0 5E                  <1>       pop    esi ; **
27648 000092F1 59                  <1>       pop    ecx ; *
27649 000092F2 C3                  <1>       retn
27650                              <1>
27651                              <1> get_env_string_seq_number_next:
27652 000092F3 AC                  <1>       lodsb
27653 000092F4 08C0                <1>       or     al, al
27654 000092F6 75FB                <1>       jnz    short get_env_string_seq_number_next
27655                              <1>
27656 000092F8 81FE00320900        <1>       cmp    esi, Env_Page + Env_Page_Size ; +512 (+4096)
27657 000092FE F5                  <1>       cmc
27658 000092FF 72EF                <1>       jc     short get_env_string_stc_retn
27659                              <1>
27660 00009301 AC                  <1>       lodsb
27661 00009302 3C01                <1>       cmp    al, 1
27662 00009304 72EA                <1>       jb     short get_env_string_stc_retn
27663 00009306 FEC1                <1>       inc    cl
27664 00009308 EBBF                <1>       jmp    short get_env_string_seq_number_check
27665                              <1>
27666                              <1> get_env_string_with_word:
27667 0000930A 31C9                <1>       xor    ecx, ecx
27668                              <1>
27669                              <1> get_env_string_calc_word_length:
27670 0000930C AC                  <1>       lodsb
27671 0000930D 3C20                <1>       cmp    al, 20h
27672 0000930F 7211                <1>       jb     short get_env_string_calc_word_length_ok
27673                              <1>       ;inc   cx
27674 00009311 FEC1                <1>       inc    cl
27675                              <1>
27676 00009313 3C61                <1>       cmp    al, 'a'
27677 00009315 72F5                <1>       jb     short get_env_string_calc_word_length
27678 00009317 3C7A                <1>       cmp    al, 'z'
27679 00009319 77F1                <1>       ja     short get_env_string_calc_word_length
27680 0000931B 24DF                <1>       and    al, 0DFh
27681 0000931D 8846FF              <1>       mov    [esi-1], al
27682 00009320 EBEA                <1>       jmp    short get_env_string_calc_word_length
27683                              <1>
27684                              <1> get_env_string_calc_word_length_ok:
27685 00009322 08C9                <1>       or     cl, cl
27686 00009324 7506                <1>       jnz    short get_env_string_calc_word_length_save
27687                              <1>
27688 00009326 5E                  <1>       pop    esi ; **
27689                              <1>
27690                              <1> get_env_string_stc_retn1:
27691 00009327 59                  <1>       pop    ecx ; *
27692                              <1>
27693                              <1> get_env_string_with_word_stc_retn:
27694 00009328 31C0                <1>       xor    eax, eax
27695 0000932A F9                  <1>       stc
27696 0000932B C3                  <1>       retn
27697                              <1>
27698                              <1> get_env_string_calc_word_length_save:
27699 0000932C 871C24              <1>       xchg   ebx, [esp] ; **
27700 0000932F 89DE                <1>       mov    esi, ebx
27701                              <1>              ; Start of the env string (to be searched)
27702                              <1>
```

```
27703 00009331 57                <1>       push   edi ; ***
27704 00009332 89D7              <1>       mov    edi, edx ; Env_Page
27705                            <1>
27706                            <1> get_env_string_compare:
27707 00009334 57                <1>       push   edi ; ****
27708 00009335 51                <1>       push   ecx ; ***** ; Variable name length
27709                            <1>
27710                            <1> get_env_string_compare_rep:
27711 00009336 AC                <1>       lodsb
27712 00009337 AE                <1>       scasb
27713 00009338 7511              <1>       jne    short get_env_string_compare_next1
27714 0000933A E2FA              <1>       loop   get_env_string_compare_rep
27715                            <1>
27716 0000933C 803F3D            <1>       cmp    byte [edi], '='
27717 0000933F 750A              <1>       jne    short get_env_string_compare_next1
27718                            <1>
27719 00009341 59                <1>       pop    ecx ; *****
27720 00009342 5F                <1>       pop    edi ; ****
27721 00009343 89FE              <1>       mov    esi, edi
27722 00009345 5F                <1>       pop    edi ; ***
27723 00009346 871C24            <1>       xchg   ebx, [esp] ; **
27724 00009349 EB82              <1>       jmp    short get_env_string_move_to_buff
27725                            <1>
27726                            <1> get_env_string_compare_next1:
27727 0000934B 89FE              <1>       mov    esi, edi
27728 0000934D 59                <1>       pop    ecx ; *****
27729 0000934E 5F                <1>       pop    edi ; ****
27730                            <1> get_env_string_compare_next2:
27731 0000934F 81FEFF310900      <1>       cmp    esi, Env_Page + Env_Page_Size - 1 ; +511 (+4095)
27732 00009355 7310              <1>       jnb    short get_env_string_compare_not_ok
27733 00009357 20C0              <1>       and    al, al
27734 00009359 AC                <1>       lodsb
27735 0000935A 75F3              <1>       jnz    short get_env_string_compare_next2
27736 0000935C 08C0              <1>       or     al, al
27737 0000935E 7407              <1>       jz     short get_env_string_compare_not_ok
27738 00009360 4E                <1>       dec    esi ; 12/04/2016
27739 00009361 89F7              <1>       mov    edi, esi
27740 00009363 89DE              <1>       mov    esi, ebx
27741 00009365 EBCD              <1>       jmp    short get_env_string_compare
27742                            <1>
27743                            <1> get_env_string_compare_not_ok:
27744 00009367 5F                <1>       pop    edi ; ***
27745 00009368 89DE              <1>       mov    esi, ebx
27746 0000936A 5B                <1>       pop    ebx ; **
27747 0000936B EBBA              <1>       jmp    short get_env_string_stc_retn1
27748                            <1>
27749                            <1> set_environment_string:
27750                            <1>       ; 13/04/2016
27751                            <1>       ; 12/04/2016
27752                            <1>       ; 11/04/2016
27753                            <1>       ; 06/04/2016
27754                            <1>       ; 05/04/2016 (TRDOS 386 = TRDOS v2.0)
27755                            <1>       ; 02/09/2011 (TRDOS v1, MAINPROG.ASM)
27756                            <1>       ; 29/08/2011
27757                            <1>       ; 29/08/2011
27758                            <1>       ; INPUT->
27759                            <1>       ;     ESI = ASCIIZ environment string
27760                            <1>       ; OUTPUT ->
27761                            <1>       ;     ESI is not changed
27762                            <1>       ;     CF = 1 -> Could not set,
27763                            <1>       ;          insufficient environment space
27764                            <1>       ;
27765                            <1>       ; (EAX, EDX will be changed)
27766                            <1>       ;
27767                            <1>       ;    (EAX = Start address of the env string if > 0)
27768                            <1>       ;    (EDX = Environment string length)
27769                            <1>
27770 0000936D 56                <1>       push   esi ; *
27771                            <1>
27772 0000936E 31C0              <1>       xor    eax, eax
27773                            <1>
27774                            <1> set_env_chk_validation1:
27775 00009370 FEC4              <1>       inc    ah ; variable (string) length
27776 00009372 AC                <1>       lodsb
27777 00009373 3C3D              <1>       cmp    al, '='
27778 00009375 7415              <1>       je     short set_env_chk_validation2
27779 00009377 3C20              <1>       cmp    al, 20h
27780 00009379 720F              <1>       jb     short set_env_string_stc
27781                            <1>
27782                            <1>       ; 06/04/2016
27783 0000937B 3C61              <1>       cmp    al, 'a'
27784 0000937D 72F1              <1>       jb     short set_env_chk_validation1
27785 0000937F 3C7A              <1>       cmp    al, 'z'
27786 00009381 77ED              <1>       ja     short set_env_chk_validation1
27787 00009383 2C20              <1>       sub    al, 'a'-'A'
27788 00009385 8846FF            <1>       mov    [esi-1], al
27789 00009388 EBE6              <1>       jmp    short set_env_chk_validation1
27790                            <1>
27791                            <1> set_env_string_stc:
27792 0000938A 5E                <1>       pop    esi ; *
27793                            <1>       ;stc
27794 0000938B C3                <1>       retn
27795                            <1>
27796                            <1> set_env_chk_validation2:
27797 0000938C 51                <1>       push   ecx ; **
27798 0000938D 53                <1>       push   ebx ; ***
27799 0000938E 57                <1>       push   edi ; ****
27800                            <1>
27801                            <1>       ; 12/04/2016
27802 0000938F 8B5C240C          <1>       mov    ebx, [esp+12]
27803                            <1>
27804                            <1> set_env_chk_validation2w:
27805 00009393 89F7              <1>       mov    edi, esi
```

```
27806 00009395 4F                <1>       dec    edi
27807                            <1>
27808 00009396 807FFF20          <1>       cmp    byte [edi-1], 20h
27809 0000939A 771A              <1>       ja     short set_env_chk_validation2z
27810                            <1>
27811 0000939C 56                <1>       push   esi
27812 0000939D 89FE              <1>       mov    esi, edi
27813 0000939F 4E                <1>       dec    esi
27814                            <1>
27815                            <1> set_env_chk_validation2x:
27816 000093A0 4E                <1>       dec    esi
27817                            <1>
27818 000093A1 39DE              <1>       cmp    esi, ebx
27819 000093A3 7207              <1>       jb     short set_env_chk_validation2y
27820                            <1>
27821 000093A5 4F                <1>       dec    edi
27822                            <1>
27823 000093A6 8A06              <1>       mov    al, [esi]
27824 000093A8 8807              <1>       mov    [edi], al
27825                            <1>
27826 000093AA EBF4              <1>       jmp    short set_env_chk_validation2x
27827                            <1>
27828                            <1> set_env_chk_validation2y:
27829 000093AC 5E                <1>       pop    esi
27830                            <1>
27831                            <1>       ;mov   byte [ebx], 20h
27832                            <1>
27833 000093AD 43                <1>       inc    ebx
27834 000093AE 895C240C          <1>       mov    [esp+12], ebx
27835                            <1>
27836 000093B2 FECC              <1>       dec    ah ; 13/04/2016
27837                            <1>
27838 000093B4 EBDD              <1>       jmp    short set_env_chk_validation2w
27839                            <1>
27840                            <1> set_env_chk_validation2z:
27841 000093B6 BA00300900        <1>       mov    edx, Env_Page
27842 000093BB 89D7              <1>       mov    edi, edx
27843                            <1>
27844                            <1> set_env_chk_validation3:
27845 000093BD AC                <1>       lodsb
27846 000093BE 3C20              <1>       cmp    al, 20h
27847 000093C0 74FB              <1>       je     short set_env_chk_validation3
27848                            <1>
27849 000093C2 9C                <1>       pushf
27850                            <1>
27851                            <1>       ; 12/04/2016
27852                            <1> set_env_chk_validation3n:
27853 000093C3 3C61              <1>       cmp    al, 'a'
27854 000093C5 720C              <1>       jb     short set_env_chk_validation3c
27855 000093C7 3C7A              <1>       cmp    al, 'z'
27856 000093C9 7705              <1>       ja     short set_env_chk_validation3x
27857 000093CB 2C20              <1>       sub    al, 'a'-'A'
27858 000093CD 8846FF            <1>       mov    [esi-1], al
27859                            <1>
27860                            <1> set_env_chk_validation3x:
27861 000093D0 AC                <1>       lodsb
27862 000093D1 EBF0              <1>       jmp    short set_env_chk_validation3n
27863                            <1>
27864                            <1> set_env_chk_validation3c:
27865 000093D3 3C20              <1>       cmp    al, 20h
27866 000093D5 73F9              <1>       jnb    short set_env_chk_validation3x
27867                            <1>
27868 000093D7 803F00            <1>       cmp    byte [edi], 0
27869 000093DA 7731              <1>       ja     short set_env_chk_validation4
27870                            <1>
27871 000093DC 9D                <1>       popf
27872 000093DD 7228              <1>       jb     short set_env_string_nothing
27873                            <1>
27874 000093DF B900020000        <1>       mov    ecx, Env_Page_Size ; 512 (4096)
27875                            <1>
27876 000093E4 89DE              <1>       mov    esi, ebx ; 12/04/2016
27877                            <1>
27878                            <1> set_env_string_copy_to_envb:
27879 000093E6 AC                <1>       lodsb
27880 000093E7 3C20              <1>       cmp    al, 20h
27881 000093E9 720A              <1>       jb     short set_env_string_copy_to_envb_z
27882 000093EB AA                <1>       stosb
27883 000093EC E2F8              <1>       loop   set_env_string_copy_to_envb
27884                            <1>
27885                            <1>       ; 11/04/2016
27886 000093EE 89D7              <1>       mov    edi, edx ; Env_Page
27887 000093F0 B900020000        <1>       mov    ecx, Env_Page_Size
27888                            <1>
27889                            <1> set_env_string_copy_to_envb_z:
27890 000093F5 52                <1>       push   edx  ; Start address of the variable
27891 000093F6 BA00020000        <1>       mov    edx, Env_Page_Size
27892 000093FB 29CA              <1>       sub    edx, ecx ; variable (string) length
27893                            <1>
27894 000093FD 28C0              <1>       sub    al, al ; 0
27895 000093FF F3AA              <1>       rep    stosb ; clear remain bytes of the env page
27896                            <1>
27897 00009401 58                <1>       pop    eax  ; Start address of the variable
27898                            <1>
27899                            <1> set_env_string_allocate_envb_retn:  ; stc or clc return
27900 00009402 5F                <1>       pop    edi ; ****
27901 00009403 5B                <1>       pop    ebx ; ***
27902 00009404 59                <1>       pop    ecx ; **
27903 00009405 5E                <1>       pop    esi ; *
27904 00009406 C3                <1>       retn
27905                            <1>
27906                            <1> set_env_string_nothing:
27907 00009407 31C0              <1>       xor    eax, eax
27908 00009409 31D2              <1>       xor    edx, edx ; 11/04/2016
```

```
27909 0000940B EBF5               <1>       jmp    short set_env_string_allocate_envb_retn
27910                             <1>
27911                             <1> set_env_chk_validation4:
27912                             <1>       ; 11/04/2016
27913 0000940D 9D                 <1>       popf
27914                             <1>
27915 0000940E 89D6               <1>       mov    esi, edx  ; Env_Page
27916                             <1>
27917                             <1> set_env_chk_validation5:
27918 00009410 89DF               <1>       mov    edi, ebx  ; ASCIIZ environment string address
27919 00009412 0FB6CC             <1>       movzx  ecx, ah ; Variable (string) length (with '=')
27920                             <1>
27921                             <1> set_env_chk_validation5_loop:
27922 00009415 AC                 <1>       lodsb
27923 00009416 AE                 <1>       scasb
27924 00009417 750A               <1>       jne    short set_env_chk_validation6
27925 00009419 E2FA               <1>       loop   set_env_chk_validation5_loop
27926                             <1>
27927 0000941B 3C3D               <1>       cmp    al, '='
27928 0000941D 0F8483000000       <1>         je      set_env_change_variable
27929                             <1>
27930                             <1> set_env_chk_validation6:
27931 00009423 08C0               <1>       or     al, al ; 0
27932 00009425 7403               <1>       jz     short set_env_chk_validation7
27933                             <1>
27934 00009427 AC                 <1>       lodsb
27935 00009428 EBF9               <1>       jmp    short set_env_chk_validation6
27936                             <1>
27937                             <1> set_env_chk_validation7:
27938 0000942A 88E1               <1>       mov    cl, ah
27939 0000942C 01F1               <1>       add    ecx, esi
27940 0000942E 81F9FF310900       <1>       cmp    ecx, Env_Page + Env_Page_Size - 1
27941                             <1>              ; 511 (4095)
27942                             <1>              ; strlen + '=' + 0
27943 00009434 72DA               <1>       jb     short set_env_chk_validation5
27944                             <1>
27945                             <1> set_env_chk_validation8: ; variable not found
27946 00009436 0FB6F4             <1>       movzx  esi, ah  ; variable name length (with '=')
27947 00009439 01DE               <1>       add    esi, ebx ; position just after of the '='
27948                             <1>
27949                             <1> set_env_chk_validation8_loop:
27950 0000943B AC                 <1>       lodsb
27951 0000943C 3C20               <1>       cmp    al, 20h
27952 0000943E 74FB               <1>       je     short set_env_chk_validation8_loop
27953 00009440 72C5               <1>       jb     short set_env_string_nothing
27954                             <1>
27955                             <1> set_env_chk_validation9:
27956 00009442 AC                 <1>       lodsb
27957 00009443 3C20               <1>       cmp    al, 20h
27958 00009445 73FB               <1>       jnb    short set_env_chk_validation9
27959                             <1>
27960                             <1>       ; End of ASCIIZ environment string
27961                             <1>
27962                             <1> set_env_add_variable:
27963 00009447 29DE               <1>       sub    esi, ebx ; variable+definition length
27964                             <1>
27965 00009449 56                 <1>       push   esi ; *****
27966                             <1>
27967 0000944A 89D6               <1>       mov    esi, edx ; Environment page address
27968                             <1>
27969 0000944C B900020000         <1>       mov    ecx, Env_Page_Size ; 512 (4096)
27970                             <1>
27971                             <1> set_env_add_variable_loop:
27972 00009451 AC                 <1>       lodsb
27973 00009452 20C0               <1>       and    al, al
27974 00009454 7406               <1>       jz     short set_env_add_variable_chk1 ; 0
27975 00009456 E2F9               <1>       loop   set_env_add_variable_loop
27976                             <1>
27977                             <1>       ; 11/04/2016
27978 00009458 884EFF             <1>       mov    [esi-1], cl ; 0
27979 0000945B 41                 <1>       inc    ecx
27980                             <1>
27981                             <1> set_env_add_variable_chk1:
27982 0000945C 49                 <1>       dec    ecx
27983 0000945D 7408               <1>       jz     short set_env_add_variable_nspc
27984 0000945F AC                 <1>       lodsb
27985 00009460 08C0               <1>       or     al, al
27986 00009462 740C               <1>       jz     short set_env_add_variable_chk2 ; 00
27987 00009464 49                 <1>       dec    ecx
27988 00009465 75EA               <1>       jnz    short set_env_add_variable_loop
27989                             <1>
27990                             <1> set_env_add_variable_nspc: ; no space on environment page
27991 00009467 58                 <1>       pop    eax ; *****
27992 00009468 B808000000         <1>       mov    eax, 8 ; No space for new environment string
27993 0000946D F9                 <1>       stc
27994 0000946E EB92               <1>         jmp     short set_env_string_allocate_envb_retn
27995                             <1>
27996                             <1> set_env_add_variable_chk2:
27997 00009470 8B0C24             <1>       mov    ecx, [esp] ; *****
27998 00009473 4E                 <1>       dec    esi ; beginning address of the new variable
27999 00009474 89F0               <1>       mov    eax, esi
28000 00009476 01C8               <1>       add    eax, ecx ; string length (with CR)
28001 00009478 81C200020000       <1>       add    edx, Env_Page_Size ; 512 (4096)
28002 0000947E 39D0               <1>       cmp    eax, edx
28003 00009480 77E5               <1>       ja     short set_env_add_variable_nspc
28004 00009482 49                 <1>       dec    ecx ; except CR at the end
28005 00009483 89CA               <1>       mov    edx, ecx ; 12/04/2016
28006 00009485 89F7               <1>       mov    edi, esi
28007 00009487 893C24             <1>       mov    [esp], edi ; ***** ; Start address of new variable
28008 0000948A 89DE               <1>       mov    esi, ebx ; ASCIIZ environment string address
28009 0000948C F3A4               <1>       rep    movsb
28010 0000948E 28C0               <1>       sub    al, al
28011 00009490 AA                 <1>       stosb
```

```
28012 00009491 58                <1>       pop    eax ; ***** ; Beginning address of new variable
28013 00009492 81FF00320900      <1>         cmp     edi, Env_Page + Env_Page_Size ; 12/04/2016
28014 00009498 0F8364FFFFFF      <1>         jnb     set_env_string_allocate_envb_retn ; OK !
28015 0000949E 880F              <1>       mov    [edi], cl ; 0
28016 000094A0 F8                <1>       clc    ; 13/04/2016
28017 000094A1 E95CFFFFFF        <1>         jmp     set_env_string_allocate_envb_retn ; OK !
28018                            <1>
28019                            <1> set_env_change_variable:
28020                            <1>       ; 06/04/2016
28021                            <1>       ; esi = Variable's address in environment page (after '=')
28022                            <1>       ; edi = ASCIIZ environment string address (after '=')
28023                            <1>
28024                            <1>       ; ah = variable length from start to the '='
28025 000094A6 8825[545F0100]    <1>       mov    [env_var_length], ah
28026                            <1>
28027 000094AC 28C9              <1>       sub    cl, cl ; ecx = 0
28028                            <1>
28029 000094AE 57                <1>       push   edi ; *****
28030                            <1>
28031 000094AF 89F7              <1>       mov    edi, esi ; 11/04/2016
28032                            <1>
28033                            <1> set_env_change_variable_calc1:
28034 000094B1 AC                <1>       lodsb
28035 000094B2 08C0              <1>       or     al, al
28036 000094B4 7403              <1>       jz     short set_env_change_variable_calc2
28037                            <1>
28038 000094B6 41                <1>       inc    ecx ; length of environment string (after the '=')
28039                            <1>
28040 000094B7 EBF8              <1>       jmp    short set_env_change_variable_calc1
28041                            <1>
28042                            <1> set_env_change_variable_calc2:
28043 000094B9 8B3424            <1>       mov    esi, [esp] ; ASCIIZ environment string address
28044                            <1>
28045 000094BC 29D2              <1>       sub    edx, edx
28046                            <1>
28047                            <1> set_env_change_variable_calc3:
28048 000094BE AC                <1>       lodsb
28049 000094BF 3C20              <1>       cmp    al, 20h
28050 000094C1 7203              <1>       jb     short set_env_change_variable_calc4
28051                            <1>
28052 000094C3 42                <1>       inc    edx ; length of ASCIIZ string (after the '=')
28053                            <1>
28054 000094C4 EBF8              <1>       jmp    short set_env_change_variable_calc3
28055                            <1>
28056                            <1> set_env_change_variable_calc4:
28057 000094C6 C646FF00          <1>       mov    byte [esi-1], 0  ; put ZERO instead of CR
28058                            <1>
28059 000094CA 5E                <1>       pop    esi ; ***** ; ASCIIZ string address (after '=')
28060                            <1>
28061                            <1>       ; EDI = Old variable's address (after '=')
28062                            <1>
28063                            <1>       ; compare the new string with the old string
28064 000094CB 39CA              <1>       cmp    edx, ecx
28065 000094CD 7717              <1>       ja     short set_env_change_variable_calc5 ; longer
28066 000094CF 0F828F000000      <1>         jb      set_env_change_variable_calc9 ; shorter
28067                            <1>
28068                            <1>       ;same length (simple copy)
28069 000094D5 0FB6C4            <1>       movzx  eax, ah
28070 000094D8 01C2              <1>       add    edx, eax
28071 000094DA F7D8              <1>       neg    eax
28072 000094DC 01F8              <1>       add    eax, edi
28073                            <1>       ; EAX = Start address of the variable
28074                            <1>       ; EDX = Variable length (without ZERO at the end of variable)
28075                            <1>
28076 000094DE F3A4              <1>       rep    movsb
28077 000094E0 F8                <1>       clc    ; 13/04/2016
28078 000094E1 E91CFFFFFF        <1>         jmp     set_env_string_allocate_envb_retn ; OK !
28079                            <1>
28080                            <1> set_env_change_variable_calc5:
28081                            <1>       ; 11/04/2016
28082 000094E6 52                <1>       push   edx ; *****
28083 000094E7 29CA              <1>       sub    edx, ecx ; difference ; (the new string is longer)
28084 000094E9 89F3              <1>       mov    ebx, esi
28085 000094EB 89FE              <1>       mov    esi, edi
28086                            <1>
28087                            <1> set_env_change_variable_calc6:
28088 000094ED AC                <1>       lodsb
28089 000094EE 20C0              <1>       and    al, al
28090 000094F0 75FB              <1>       jnz    short set_env_change_variable_calc6
28091                            <1>
28092 000094F2 81FE00320900      <1>       cmp    esi, Env_Page + Env_Page_Size ; 512 (4096)
28093 000094F8 0F8369FFFFFF      <1>         jnb     set_env_add_variable_nspc
28094                            <1>
28095 000094FE 89F9              <1>       mov    ecx, edi  ; current (old) variable's address
28096 00009500 89F7              <1>       mov    edi, esi  ; next variable's address
28097                            <1>
28098 00009502 AC                <1>       lodsb
28099 00009503 08C0              <1>       or     al, al
28100 00009505 7416              <1>       jz     short set_env_change_variable_calc8 ; 00
28101                            <1>
28102                            <1> set_env_change_variable_calc7:
28103 00009507 AC                <1>       lodsb
28104 00009508 20C0              <1>       and    al, al
28105 0000950A 75FB              <1>       jnz    short set_env_change_variable_calc7
28106                            <1>
28107 0000950C 81FE00320900      <1>       cmp    esi, Env_Page + Env_Page_Size ; 512 (4096)
28108 00009512 0F834FFFFFFF      <1>         jnb     set_env_add_variable_nspc
28109                            <1>
28110 00009518 AC                <1>       lodsb
28111 00009519 08C0              <1>       or     al, al
28112 0000951B 75EA              <1>       jnz    short set_env_change_variable_calc7
28113                            <1>
28114                            <1> set_env_change_variable_calc8:
```

```
28115 0000951D 4E                <1>       dec    esi ; address of the second (last) 0 of the 00
28116                            <1>
28117 0000951E 01F2              <1>       add    edx, esi ; final position of the last 0
28118                            <1>
28119 00009520 81FA00320900      <1>       cmp    edx, Env_Page + Env_Page_Size ; 512 (4096)
28120 00009526 0F833BFFFFFF      <1>         jnb     set_env_add_variable_nspc
28121                            <1>
28122 0000952C 89C8              <1>       mov    eax, ecx ; old variable's address (after '=')
28123                            <1>
28124 0000952E 89F1              <1>       mov    ecx, esi
28125 00009530 29F9              <1>       sub    ecx, edi ; count of bytes to move forward
28126                            <1>
28127                            <1>       ; 13/04/2016
28128 00009532 C60200            <1>       mov    byte [edx], 0
28129 00009535 89D7              <1>       mov    edi, edx
28130 00009537 29F2              <1>       sub    edx, esi ; difference (additional byte count)
28131 00009539 4F                <1>       dec    edi ; the last zero address (first byte of the 00)
28132 0000953A 89FE              <1>       mov    esi, edi
28133 0000953C 29D6              <1>       sub    esi, edx ; - displacement
28134                            <1>
28135 0000953E FA                <1>       cli    ; disable interrupts
28136 0000953F FD                <1>       std    ; backward
28137                            <1>
28138 00009540 F3A4              <1>       rep    movsb ; move ECX bytes from DS:ESI to ES:EDI
28139                            <1>
28140 00009542 FC                <1>       cld    ; forward (default)
28141 00009543 FB                <1>       sti    ; enable interrupts
28142                            <1>
28143 00009544 89C7              <1>       mov    edi, eax
28144 00009546 59                <1>       pop    ecx ; ***** ; byte count (after '=')
28145 00009547 89CA              <1>       mov    edx, ecx
28146 00009549 89DE              <1>       mov    esi, ebx ; ASCIIZ string address (after '=')
28147 0000954B 89FB              <1>       mov    ebx, edi
28148                            <1>
28149 0000954D F3A4              <1>       rep    movsb
28150                            <1>
28151 0000954F 880F              <1>       mov    [edi], cl ; 0 ; end of variable
28152                            <1>
28153 00009551 0FB605[545F0100]  <1>       movzx  eax, byte [env_var_length]
28154 00009558 01C2              <1>       add    edx, eax ; variable length (total)
28155 0000955A F7D8              <1>       neg    eax
28156 0000955C 01D8              <1>       add    eax, ebx ; start address of the variable
28157 0000955E F8                <1>       clc    ; 13/04/2016
28158 0000955F E99EFEFFFF        <1>         jmp     set_env_string_allocate_envb_retn ; OK !
28159                            <1>
28160                            <1> set_env_change_variable_calc9:
28161                            <1>       ; 11/04/2016
28162 00009564 21D2              <1>       and    edx, edx ; is empty ?
28163 00009566 753B              <1>       jnz    short set_env_change_variable_calc15
28164                            <1>
28165 00009568 0FB6DC            <1>       movzx  ebx, ah
28166 0000956B F7DB              <1>       neg    ebx
28167 0000956D 01FB              <1>       add    ebx, edi
28168                            <1>
28169                            <1>       ; EBX = Start address of the variable (in env page)
28170                            <1>       ; EDX = Variable length = 0
28171                            <1>
28172 0000956F 89FE              <1>       mov    esi, edi
28173                            <1>
28174                            <1> set_env_change_variable_calc10:
28175 00009571 AC                <1>       lodsb
28176 00009572 08C0              <1>       or     al, al
28177 00009574 75FB              <1>       jnz    short set_env_change_variable_calc10
28178                            <1>
28179 00009576 B9FF310900        <1>       mov    ecx, Env_Page + Env_Page_Size - 1
28180                            <1>
28181 0000957B 39CE              <1>       cmp    esi, ecx ; +511 (+4095)
28182 0000957D 7604              <1>       jna    short set_env_change_variable_calc11
28183                            <1>
28184 0000957F 89CE              <1>       mov    esi, ecx
28185 00009581 8806              <1>       mov    [esi], al ; 0
28186                            <1>
28187                            <1> set_env_change_variable_calc11:
28188 00009583 89DF              <1>       mov    edi, ebx ; old variable's start address
28189                            <1>
28190                            <1> set_env_change_variable_calc12:
28191 00009585 AC                <1>       lodsb
28192 00009586 AA                <1>       stosb
28193 00009587 20C0              <1>       and    al, al
28194 00009589 75FA              <1>       jnz    short set_env_change_variable_calc12
28195 0000958B 39CE              <1>       cmp    esi, ecx
28196 0000958D 7706              <1>       ja     short set_env_change_variable_calc13
28197 0000958F AC                <1>       lodsb
28198 00009590 AA                <1>       stosb
28199 00009591 20C0              <1>       and    al, al
28200 00009593 75F0              <1>       jnz    short set_env_change_variable_calc12
28201                            <1>
28202                            <1> set_env_change_variable_calc13:
28203 00009595 29F9              <1>       sub    ecx, edi
28204 00009597 7203              <1>       jb     short set_env_change_variable_calc14
28205 00009599 41                <1>       inc    ecx ; 1-512 (1-4096)
28206 0000959A F3AA              <1>       rep    stosb ; al = 0
28207                            <1>
28208                            <1> set_env_change_variable_calc14:
28209 0000959C 29C0              <1>       sub    eax, eax ; Start address of the variable
28210                            <1>       ; EAX = 0 -> Variable is removed
28211                            <1>       ; EDX = Variable length = 0
28212                            <1>
28213 0000959E E95FFEFFFF        <1>         jmp     set_env_string_allocate_envb_retn ; OK !
28214                            <1>
28215                            <1> set_env_change_variable_calc15:
28216 000095A3 52                <1>       push   edx ; *****
28217 000095A4 F7DA              <1>       neg    edx
```

```
28218 000095A6 01CA                <1>       add    edx, ecx ; difference (the old string is longer)
28219 000095A8 89F3                <1>       mov    ebx, esi
28220 000095AA 89FE                <1>       mov    esi, edi
28221                              <1>
28222                              <1> set_env_change_variable_calc16:
28223 000095AC AC                  <1>       lodsb
28224 000095AD 20C0                <1>       and    al, al
28225 000095AF 75FB                <1>       jnz    short set_env_change_variable_calc16
28226                              <1>
28227 000095B1 B900320900          <1>       mov    ecx, Env_Page + Env_Page_Size
28228                              <1>
28229 000095B6 39CE                <1>       cmp    esi, ecx ; +512 (+4096)
28230 000095B8 7605                <1>       jna    short set_env_change_variable_calc17
28231                              <1>
28232 000095BA 89CE                <1>       mov    esi, ecx
28233 000095BC 8846FF              <1>       mov    [esi-1], al ; 0
28234                              <1>
28235                              <1> set_env_change_variable_calc17:
28236 000095BF 89F9                <1>       mov    ecx, edi  ; current (old) variable's address
28237 000095C1 89F7                <1>       mov    edi, esi  ; next variable's address
28238                              <1>
28239 000095C3 AC                  <1>       lodsb
28240 000095C4 08C0                <1>       or     al, al
28241 000095C6 741D                <1>       jz     short set_env_change_variable_calc20
28242                              <1>
28243                              <1> set_env_change_variable_calc18:
28244 000095C8 AC                  <1>       lodsb
28245 000095C9 20C0                <1>       and    al, al
28246 000095CB 75FB                <1>       jnz    short set_env_change_variable_calc18
28247                              <1>
28248 000095CD 81FE00320900        <1>       cmp    esi, Env_Page + Env_Page_Size
28249 000095D3 720B                <1>       jb     short set_env_change_variable_calc19
28250 000095D5 740E                <1>       je     short set_env_change_variable_calc20
28251                              <1>
28252 000095D7 BEFF310900          <1>       mov    esi, Env_Page + Env_Page_Size - 1
28253 000095DC 8806                <1>       mov    [esi], al ; 0
28254 000095DE EB06                <1>       jmp    short set_env_change_variable_calc21
28255                              <1>
28256                              <1> set_env_change_variable_calc19:
28257 000095E0 AC                  <1>       lodsb
28258 000095E1 08C0                <1>       or     al, al
28259 000095E3 75E3                <1>       jnz    short set_env_change_variable_calc18
28260                              <1>
28261                              <1> set_env_change_variable_calc20:
28262 000095E5 4E                  <1>       dec    esi ; address of the second (last) 0 of the 00
28263                              <1>
28264                              <1> set_env_change_variable_calc21:
28265                              <1>       ; edx = difference (byte count)
28266                              <1>
28267 000095E6 89C8                <1>       mov    eax, ecx ; old variable's address (after '=')
28268                              <1>
28269 000095E8 89F1                <1>       mov    ecx, esi
28270 000095EA 29F9                <1>       sub    ecx, edi ; count of bytes to move backward
28271                              <1>
28272 000095EC 89FE                <1>       mov    esi, edi ; next variable's address
28273 000095EE 29D7                <1>       sub    edi, edx ; (displacement)
28274                              <1>
28275 000095F0 F3A4                <1>       rep    movsb
28276                              <1>
28277 000095F2 880F                <1>       mov    [edi], cl ; 0 ; 00 ; end of environment variables
28278                              <1>
28279 000095F4 89C7                <1>       mov    edi, eax
28280 000095F6 5A                  <1>       pop    edx ; ***** ; byte count (after '=')
28281 000095F7 89D1                <1>       mov    ecx, edx
28282 000095F9 89DE                <1>       mov    esi, ebx ; ASCIIZ string address (after '=')
28283 000095FB 89FB                <1>       mov    ebx, edi
28284                              <1>
28285 000095FD F3A4                <1>       rep    movsb
28286                              <1>
28287 000095FF 880F                <1>       mov    [edi], cl ; 0 ; end of variable
28288                              <1>
28289 00009601 0FB605[545F0100]    <1>       movzx  eax, byte [env_var_length]
28290 00009608 01C2                <1>       add    edx, eax ; variable length (total)
28291 0000960A F7D8                <1>       neg    eax
28292 0000960C 01D8                <1>       add    eax, ebx ; start address of the variable
28293 0000960E F8                  <1>       clc    ; 13/04/2016
28294 0000960F E9EEFDFFFF          <1>         jmp     set_env_string_allocate_envb_retn ; OK !
28295                              <1>
28296                              <1> mainprog_startup_configuration:
28297                              <1>       ; 22/11/2017
28298                              <1>       ; 06/05/2016
28299                              <1>       ; 14/04/2016 (TRDOS 386 = TRDOS v2.0)
28300                              <1>       ; 17/09/2011 (TRDOS v1, MAINPROG.ASM)
28301                              <1>       ;
28302                              <1> loc_load_mainprog_cfg_file:
28303 00009614 BE[19070100]        <1>       mov    esi, MainProgCfgFile
28304 00009619 66B80018            <1>       mov    ax, 1800h ; Except volume label and dirs
28305 0000961D E8E9E9FFFF          <1>       call   find_first_file
28306 00009622 7256                <1>       jc     short loc_load_mainprog_cfg_exit
28307                              <1>
28308                              <1>       ;or    eax, eax
28309                              <1>       ;jz    short loc_load_mainprog_cfg_exit
28310                              <1>
28311                              <1> loc_start_mainprog_configuration:
28312                              <1>       ; ESI = FindFile_DirEntry Location
28313                              <1>       ; EAX = File Size
28314                              <1>
28315 00009624 A3[D4520100]        <1>       mov    [MainProgCfg_FileSize], eax
28316                              <1>
28317 00009629 668B5614            <1>       mov    dx, [esi+DirEntry_FstClusHI]
28318 0000962D C1E210              <1>       shl    edx, 16
28319 00009630 668B561A            <1>       mov    dx, [esi+DirEntry_FstClusLO]
28320 00009634 8915[085F0100]      <1>       mov    [csftdf_sf_cluster], edx
```

```
28321                                  <1>
28322 0000963A 89C1                    <1>       mov    ecx, eax
28323 0000963C 29C0                    <1>       sub    eax, eax
28324                                  <1>
28325                                  <1>       ; TRDOS 386 (TRDOS v2.0)
28326                                  <1>       ; Allocate contiguous memory block for loading the file
28327                                  <1>
28328                                  <1>       ; eax = 0 (Allocate memory from the beginning)
28329                                  <1>       ; ecx = File (Allocation) size in bytes
28330                                  <1>
28331 0000963E E8E1BDFFFF              <1>       call   allocate_memory_block
28332 00009643 7235                    <1>       jc     short loc_load_mainprog_cfg_exit
28333                                  <1>
28334 00009645 A3[005F0100]            <1>       mov    [csftdf_sf_mem_addr], eax ; loading address
28335 0000964A 890D[045F0100]          <1>       mov    [csftdf_sf_mem_bsize], ecx ; block size
28336                                  <1>
28337 00009650 31DB                    <1>       xor    ebx, ebx
28338                                  <1>       ;mov   [csftdf_sf_rbytes], ebx ; 0, reset
28339                                  <1>
28340 00009652 8A3D[E6520100]          <1>       mov    bh, [Current_Drv] ; [FindFile_Drv]
28341 00009658 BE00010900              <1>       mov    esi, Logical_DOSDisks
28342 0000965D 01DE                    <1>       add    esi, ebx
28343                                  <1>
28344 0000965F 8B1D[005F0100]          <1>       mov    ebx, [csftdf_sf_mem_addr] ; memory block address
28345                                  <1>
28346 00009665 807E0300                <1>       cmp    byte [esi+LD_FATType], 0
28347 00009669 7710                    <1>         ja   short loc_mcfg_load_fat_file
28348                                  <1>
28349 0000966B C705[105F0100]0000-     <1>       mov    dword [csftdf_r_size], 65536
28350 00009673 0100                    <1>
28351 00009675 E9A1010000              <1>         jmp     loc_mcfg_load_fs_file
28352                                  <1>
28353                                  <1> loc_load_mainprog_cfg_exit:
28354 0000967A C3                      <1>       retn
28355                                  <1>
28356                                  <1> loc_mcfg_load_fat_file:
28357 0000967B 0FB74611                <1>       movzx  eax, word [esi+LD_BPB+BytesPerSec]
28358 0000967F 0FB64E13                <1>       movzx  ecx, byte [esi+LD_BPB+SecPerClust]
28359 00009683 F7E1                    <1>       mul    ecx
28360 00009685 A3[105F0100]            <1>       mov    [csftdf_r_size], eax
28361                                  <1>
28362                                  <1> loc_mcfg_load_fat_file_next:
28363 0000968A E822010000              <1>       call   mcfg_read_fat_file_sectors
28364 0000968F 0F8206010000            <1>         jc      mcfg_deallocate_mem
28365                                  <1>
28366 00009695 09D2                    <1>       or     edx, edx ; edx > 0 -> EOF
28367 00009697 74F1                    <1>       jz     short loc_mcfg_load_fat_file_next
28368                                  <1>
28369                                  <1> loc_mcfg_load_fat_file_ok:
28370                                  <1>       ; 06/05/2016
28371 00009699 C705[A45F0100]-         <1>       mov    dword [mainprog_return_addr], loc_mcfg_ci_return_addr
28372 0000969F [5C970000]              <1>
28373                                  <1>       ;
28374 000096A3 8B35[005F0100]          <1>       mov    esi, [csftdf_sf_mem_addr]
28375 000096A9 8935[D8520100]          <1>       mov    [MainProgCfg_LineOffset], esi
28376                                  <1>
28377 000096AF A1[D4520100]            <1>       mov    eax, [MainProgCfg_FileSize]
28378 000096B4 89C2                    <1>       mov    edx, eax
28379 000096B6 01F2                    <1>       add    edx, esi
28380                                  <1>
28381                                  <1> loc_mcfg_process_next_line_check:
28382 000096B8 89C1                    <1>       mov    ecx, eax
28383                                  <1>
28384 000096BA 803E2A                  <1>       cmp    byte [esi], "*" ; Remark sign
28385 000096BD 7503                    <1>       jne    short loc_mcfg_process_next_line
28386 000096BF 46                      <1>       inc    esi
28387 000096C0 EB17                    <1>       jmp    short loc_move_mainprog_cfg_nl1
28388                                  <1>
28389                                  <1> loc_mcfg_process_next_line:
28390 000096C2 83F94F                  <1>       cmp    ecx, 79
28391 000096C5 7605                    <1>       jna    short loc_start_mainprog_cfg_process
28392                                  <1>
28393 000096C7 B94F000000              <1>       mov    ecx, 79
28394                                  <1>
28395                                  <1> loc_start_mainprog_cfg_process:
28396 000096CC BF[96530100]            <1>       mov    edi, CommandBuffer
28397                                  <1>
28398                                  <1> loc_move_mainprog_cfg_line:
28399 000096D1 AC                      <1>       lodsb
28400 000096D2 3C20                    <1>       cmp    al, 20h
28401 000096D4 720C                    <1>       jb     short loc_move_mainprog_cfg_nl2
28402 000096D6 AA                      <1>       stosb
28403 000096D7 E2F8                    <1>       loop   loc_move_mainprog_cfg_line
28404                                  <1>
28405                                  <1> loc_move_mainprog_cfg_nl1:
28406 000096D9 39D6                    <1>       cmp    esi, edx ; + configuration file size
28407 000096DB 7312                    <1>       jnb    short loc_end_of_mainprog_cfg_line
28408 000096DD AC                      <1>       lodsb
28409 000096DE 3C20                    <1>       cmp    al, 20h
28410 000096E0 73F7                    <1>       jnb    short loc_move_mainprog_cfg_nl1
28411                                  <1>
28412                                  <1> loc_move_mainprog_cfg_nl2:
28413 000096E2 39D6                    <1>       cmp    esi, edx
28414 000096E4 7309                    <1>       jnb    short loc_end_of_mainprog_cfg_line
28415 000096E6 8A06                    <1>       mov    al, [esi]
28416 000096E8 3C20                    <1>       cmp    al, 20h
28417 000096EA 7703                    <1>       ja     short loc_end_of_mainprog_cfg_line
28418 000096EC 46                      <1>       inc    esi
28419 000096ED EBF3                    <1>       jmp    short loc_move_mainprog_cfg_nl2
28420                                  <1>
28421                                  <1> loc_end_of_mainprog_cfg_line:
28422 000096EF C60700                  <1>       mov    byte [edi], 0
28423                                  <1>
```

```
28424 000096F2 8935[D8520100]      <1>       mov    [MainProgCfg_LineOffset], esi
28425                              <1>
28426                              <1>       ; 22/11/2017
28427 000096F8 BE[9E530100]        <1>       mov    esi, CommandBuffer + 8
28428 000096FD 29FE                <1>       sub    esi, edi
28429 000096FF 7606                <1>       jna    short loc_move_mainprog_cfg_command
28430 00009701 30C0                <1>       xor    al, al
28431                              <1> loc_mainprog_cfg_clear_chrs:
28432 00009703 AA                  <1>       stosb
28433 00009704 4E                  <1>       dec    esi
28434 00009705 75FC                <1>       jnz    short loc_mainprog_cfg_clear_chrs
28435                              <1>
28436                              <1> loc_move_mainprog_cfg_command:
28437 00009707 BE[96530100]        <1>       mov    esi, CommandBuffer
28438 0000970C 89F7                <1>       mov    edi, esi
28439 0000970E 31DB                <1>       xor    ebx, ebx
28440                              <1>       ;xor   ecx, ecx
28441 00009710 30C9                <1>       xor    cl, cl
28442                              <1>
28443                              <1> loc_move_mcfg_first_cmd_char:
28444 00009712 8A041E              <1>       mov    al, [esi+ebx]
28445 00009715 FEC3                <1>       inc    bl
28446 00009717 3C20                <1>       cmp    al, 20h
28447 00009719 7712                <1>       ja     short loc_move_mcfg_cmd_capitalizing
28448 0000971B 7237                <1>       jb     short loc_move_mcfg_cmd_arguments_ok
28449 0000971D 80FB4F              <1>       cmp    bl, 79
28450 00009720 72F0                <1>       jb     short loc_move_mcfg_first_cmd_char
28451 00009722 EB30                <1>       jmp    short loc_move_mcfg_cmd_arguments_ok
28452                              <1>
28453                              <1> loc_move_mcfg_next_cmd_char:
28454 00009724 8A041E              <1>       mov    al, [esi+ebx]
28455 00009727 FEC3                <1>       inc    bl
28456 00009729 3C20                <1>       cmp    al, 20h
28457 0000972B 7614                <1>       jna    short loc_move_mcfg_cmd_ok
28458                              <1>
28459                              <1> loc_move_mcfg_cmd_capitalizing:
28460 0000972D 3C61                <1>       cmp    al, 61h ; 'a'
28461 0000972F 7206                <1>       jb     short loc_move_mcfg_cmd_caps_ok
28462 00009731 3C7A                <1>       cmp    al, 7Ah ; 'z'
28463 00009733 7702                <1>       ja     short loc_move_mcfg_cmd_caps_ok
28464 00009735 24DF                <1>       and    al, 0DFh ; sub      al, 'a'-'A'
28465                              <1>
28466                              <1> loc_move_mcfg_cmd_caps_ok:
28467 00009737 AA                  <1>       stosb
28468 00009738 FEC1                <1>       inc    cl
28469 0000973A 80FB4F              <1>       cmp    bl, 79
28470 0000973D 72E5                <1>       jb     short loc_move_mcfg_next_cmd_char
28471 0000973F EB13                <1>       jmp    short loc_move_mcfg_cmd_arguments_ok
28472                              <1>
28473                              <1> loc_move_mcfg_cmd_ok:
28474 00009741 30C0                <1>       xor    al, al ; 0
28475                              <1>
28476                              <1> loc_move_mcfg_cmd_arguments:
28477 00009743 8807                <1>       mov    [edi], al
28478 00009745 47                  <1>       inc    edi
28479 00009746 80FB4F              <1>       cmp    bl, 79
28480 00009749 7309                <1>       jnb    short loc_move_mcfg_cmd_arguments_ok
28481 0000974B 8A041E              <1>       mov    al, [esi+ebx]
28482 0000974E FEC3                <1>       inc    bl
28483 00009750 3C20                <1>       cmp    al, 20h
28484 00009752 73EF                <1>       jnb    short loc_move_mcfg_cmd_arguments
28485                              <1>
28486                              <1> loc_move_mcfg_cmd_arguments_ok:
28487 00009754 C60700              <1>       mov    byte [edi], 0
28488                              <1>
28489                              <1> loc_mcfg_process_cmd_interpreter:
28490 00009757 E8DDDFFFFF          <1>       call    command_interpreter
28491                              <1>
28492                              <1> loc_mcfg_ci_return_addr:
28493 0000975C A1[D4520100]        <1>       mov    eax, [MainProgCfg_FileSize]
28494 00009761 89C2                <1>       mov    edx, eax
28495 00009763 8B35[D8520100]      <1>       mov    esi, [MainProgCfg_LineOffset]
28496 00009769 01F2                <1>       add    edx, esi
28497 0000976B 0305[005F0100]      <1>       add    eax, [csftdf_sf_mem_addr]
28498 00009771 29F0                <1>       sub    eax, esi
28499 00009773 0F873FFFFFFF        <1>         ja      loc_mcfg_process_next_line_check
28500                              <1>
28501 00009779 E81D000000          <1>       call   mcfg_deallocate_mem
28502                              <1>
28503 0000977E B94F000000          <1>       mov    ecx, 79 ; 80 ?
28504 00009783 BF[96530100]        <1>       mov    edi, CommandBuffer
28505 00009788 30C0                <1>       xor    al, al
28506 0000978A F3AA                <1>       rep    stosb
28507                              <1>
28508                              <1>       ; 06/05/2016
28509 0000978C BE[6F130100]        <1>       mov    esi, nextline
28510 00009791 E8C7CBFFFF          <1>       call   print_msg
28511 00009796 E916D6FFFF          <1>       jmp    dos_prompt
28512                              <1>
28513                              <1> mcfg_deallocate_mem:
28514 0000979B A1[005F0100]        <1>       mov    eax, [csftdf_sf_mem_addr] ; start address
28515 000097A0 8B0D[045F0100]      <1>       mov    ecx, [csftdf_sf_mem_bsize] ; block size
28516                              <1>       ;call  deallocate_memory_block
28517                              <1>       ;retn
28518 000097A6 E986BEFFFF          <1>       jmp    deallocate_memory_block
28519                              <1>
28520                              <1> mcfg_read_file_sectors:
28521                              <1>       ; 14/04/2016
28522 000097AB 807E0300            <1>       cmp    byte [esi+LD_FATType], 0
28523 000097AF 7669                <1>         jna  short mcfg_read_fs_file_sectors
28524                              <1>
28525                              <1> mcfg_read_fat_file_sectors:
28526                              <1>       ; return:
```

```
28527                                  <1>       ;   CF = 0 & EDX > 0 -> END OF FILE
28528                                  <1>       ;   CF = 0 & EDX = 0 -> not EOF
28529                                  <1>       ;   CF = 1 -> read error (error code in AL)
28530                                  <1>
28531                                  <1> mcfg_read_fat_file_secs_0:
28532 000097B1 8B15[D4520100]          <1>       mov   edx, [MainProgCfg_FileSize]
28533 000097B7 2B15[185F0100]          <1>       sub   edx, [csftdf_sf_rbytes]
28534 000097BD 3B15[105F0100]          <1>       cmp   edx, [csftdf_r_size]
28535 000097C3 7306                    <1>       jnb   short mcfg_read_fat_file_secs_1
28536 000097C5 8915[105F0100]          <1>       mov   [csftdf_r_size], edx
28537                                  <1>
28538                                  <1> mcfg_read_fat_file_secs_1:
28539 000097CB A1[105F0100]            <1>       mov   eax, [csftdf_r_size]
28540 000097D0 29D2                    <1>       sub   edx, edx
28541 000097D2 0FB74E11                <1>       movzx ecx, word [esi+LD_BPB+BytesPerSec]
28542 000097D6 01C8                    <1>       add   eax, ecx
28543 000097D8 48                      <1>       dec   eax
28544 000097D9 F7F1                    <1>       div   ecx
28545 000097DB 89C1                    <1>       mov   ecx, eax ; sector count
28546 000097DD A1[085F0100]            <1>       mov   eax, [csftdf_sf_cluster]
28547                                  <1>
28548                                  <1>       ; EBX = memory block address (current)
28549                                  <1>
28550 000097E2 E88E230000              <1>       call  read_fat_file_sectors
28551 000097E7 7230                    <1>       jc    short mcfg_read_fat_file_secs_3
28552                                  <1>
28553                                  <1>       ; EBX = next memory address
28554                                  <1>
28555 000097E9 A1[185F0100]            <1>       mov   eax, [csftdf_sf_rbytes]
28556 000097EE 0305[105F0100]          <1>       add   eax, [csftdf_r_size]
28557 000097F4 8B15[D4520100]          <1>       mov   edx, [MainProgCfg_FileSize]
28558 000097FA 39D0                    <1>       cmp   eax, edx
28559 000097FC 731B                    <1>       jnb   short mcfg_read_fat_file_secs_3 ; edx > 0
28560 000097FE A3[185F0100]            <1>       mov   [csftdf_sf_rbytes], eax
28561                                  <1>
28562 00009803 53                      <1>       push  ebx ; *
28563                                  <1>       ; get next cluster (csftdf_r_size! bytes)
28564 00009804 A1[085F0100]            <1>       mov   eax, [csftdf_sf_cluster]
28565 00009809 E839210000              <1>       call  get_next_cluster
28566 0000980E 5B                      <1>       pop   ebx ; *
28567 0000980F 7301                    <1>       jnc   short mcfg_read_fat_file_secs_2
28568                                  <1>
28569                                  <1>       ;mov  eax, 17; Read error !
28570 00009811 C3                      <1>       retn
28571                                  <1>
28572                                  <1> mcfg_read_fat_file_secs_2:
28573 00009812 29D2                    <1>       sub   edx, edx ; 0
28574 00009814 A3[085F0100]            <1>       mov   [csftdf_sf_cluster], eax ; next cluster
28575                                  <1>
28576                                  <1> mcfg_read_fat_file_secs_3:
28577 00009819 C3                      <1>       retn
28578                                  <1>
28579                                  <1> mcfg_read_fs_file_sectors:
28580 0000981A C3                      <1>       retn
28581                                  <1>
28582                                  <1> loc_mcfg_load_fs_file:
28583 0000981B C3                      <1>       retn
28584                                  <1>
28585                                  <1> load_and_execute_file:
28586                                  <1>       ; 04/01/2017
28587                                  <1>       ; 06/05/2016, 07/05/2016, 11/05/2016
28588                                  <1>       ; 23/04/2016, 24/04/2016
28589                                  <1>       ; 22/04/2016 (TRDOS 386 = TRDOS v2.0)
28590                                  <1>       ; 05/11/2011
28591                                  <1>       ; (TRDOS v1, CMDINTR.ASM, 'cmp_cmd_run', 'cmp_cmd_external')
28592                                  <1>       ; ('loc_run_check_filename')
28593                                  <1>       ; 29/08/2011
28594                                  <1>       ; 10/09/2011
28595                                  <1>       ; INPUT->
28596                                  <1>       ;     ESI = Path Name address (CommandBuffer address)
28597                                  <1>       ; OUTPUT ->
28598                                  <1>       ;     none (error message will be shown if an error will occur)
28599                                  <1>       ;
28600                                  <1>       ; (EAX, EBX, ECX, EDX, ESI, EDI, EBP will be changed)
28601                                  <1>       ;
28602                                  <1> loc_run_check_filename:
28603 0000981C 803E20                  <1>       cmp   byte [esi], 20h
28604 0000981F 0F82E4E2FFFF            <1>       jb    loc_cmd_failed
28605 00009825 7703                    <1>       ja    short loc_run_check_filename_ok
28606 00009827 46                      <1>       inc   esi
28607 00009828 EBF2                    <1>       jmp   short loc_run_check_filename
28608                                  <1>
28609                                  <1> loc_run_check_filename_ok:
28610 0000982A C605[47530100]00        <1>       mov   byte [CmdArgStart], 0 ; reset
28611 00009831 56                      <1>       push  esi ; *
28612                                  <1> loc_run_get_first_arg_pos:
28613 00009832 46                      <1>       inc   esi
28614 00009833 8A06                    <1>       mov   al, [esi]
28615 00009835 3C20                    <1>       cmp   al, 20h
28616 00009837 77F9                    <1>       ja    short loc_run_get_first_arg_pos
28617 00009839 C60600                  <1>       mov   byte [esi], 0
28618                                  <1> loc_run_get_external_arg_pos:
28619                                  <1>       ; 11/05/2016
28620 0000983C 46                      <1>       inc   esi
28621 0000983D 8A06                    <1>       mov   al, [esi]
28622 0000983F 3C20                    <1>       cmp   al, 20h
28623 00009841 760C                    <1>       jna   short loc_run_parse_path_name
28624 00009843 89F0                    <1>       mov   eax, esi
28625 00009845 2D[96530100]            <1>       sub   eax, CommandBuffer
28626 0000984A A2[47530100]            <1>       mov   byte [CmdArgStart], al
28627                                  <1> loc_run_parse_path_name:
28628 0000984F 5E                      <1>       pop   esi ; *
28629 00009850 BF[8A5C0100]            <1>       mov   edi, FindFile_Drv
```

```
28630 00009855 E8D7090000          <1>       call   parse_path_name
28631 0000985A 0F82A9E2FFFF        <1>       jc     loc_cmd_failed
28632                              <1>
28633                              <1> loc_run_check_filename_exists:
28634 00009860 BE[CC5C0100]        <1>       mov    esi, FindFile_Name
28635 00009865 803E20              <1>       cmp    byte [esi], 20h
28636 00009868 0F869BE2FFFF        <1>       jna    loc_cmd_failed
28637                              <1>
28638                              <1> loc_run_check_exe_filename_ext:
28639 0000986E E890020000          <1>       call   check_prg_filename_ext
28640 00009873 0F8290E2FFFF        <1>       jc     loc_cmd_failed
28641                              <1>
28642                              <1> loc_run_check_exe_filename_ext_ok:
28643 00009879 66A3[A25F0100]      <1>       mov    word [EXE_ID], ax
28644                              <1>
28645                              <1> loc_run_drv:
28646 0000987F C605[A15F0100]00    <1>       mov    byte [Run_Manual_Path], 0
28647 00009886 A1[E0520100]        <1>       mov    eax, [Current_Dir_FCluster]
28648 0000988B A3[9C5F0100]        <1>         mov     [Run_CDirFC], eax
28649                              <1>       ;
28650 00009890 8A35[E6520100]      <1>       mov    dh, [Current_Drv]
28651 00009896 8835[465B0100]      <1>       mov    [RUN_CDRV], dh
28652                              <1>
28653 0000989C 8A15[8A5C0100]      <1>       mov    dl, [FindFile_Drv]
28654 000098A2 38F2                <1>       cmp    dl, dh
28655 000098A4 7412                <1>       je     short loc_run_change_directory
28656                              <1>
28657 000098A6 8005[A15F0100]02    <1>       add    byte [Run_Manual_Path], 2
28658                              <1>
28659 000098AD E8BED3FFFF          <1>       call   change_current_drive
28660 000098B2 0F827CE2FFFF        <1>       jc     loc_run_cmd_failed
28661                              <1>
28662                              <1> loc_run_change_directory:
28663 000098B8 803D[8B5C0100]20    <1>       cmp    byte [FindFile_Directory], 20h
28664 000098BF 7623                <1>       jna    short loc_run_find_executable_file
28665                              <1>
28666 000098C1 FE05[A15F0100]      <1>       inc    byte [Run_Manual_Path]
28667                              <1>
28668 000098C7 FE05[D3060100]      <1>       inc    byte [Restore_CDIR]
28669                              <1>
28670 000098CD BE[8B5C0100]        <1>       mov    esi, FindFile_Directory
28671 000098D2 30E4                <1>       xor    ah, ah ; CD_COMMAND sign -> 0
28672 000098D4 E842030000          <1>       call   change_current_directory
28673 000098D9 0F8255E2FFFF        <1>       jc     loc_run_cmd_failed
28674                              <1>
28675                              <1> loc_run_change_prompt_dir_string:
28676 000098DF E857020000          <1>       call   change_prompt_dir_string
28677                              <1>
28678                              <1> loc_run_find_executable_file:
28679 000098E4 66C705[A05F0100]00- <1>       mov    word [Run_Auto_Path], 0
28680 000098EC 00                  <1>
28681                              <1>
28682                              <1> loc_run_find_executable_file_next:
28683 000098ED BE[CC5C0100]        <1>       mov    esi, FindFile_Name
28684                              <1> loc_run_find_program_file_next:
28685 000098F2 66B80018            <1>       mov    ax, 1800h ; Except volume label and dirs
28686 000098F6 E810E7FFFF          <1>       call   find_first_file
28687                              <1>       ; ESI = Directory Entry (FindFile_DirEntry) Location
28688                              <1>       ; EDI = Directory Buffer Directory Entry Location
28689                              <1>       ; EAX = File size
28690 000098FB 0F835C010000        <1>       jnc    loc_load_and_run_file
28691                              <1>
28692 00009901 3C02                <1>       cmp    al, 2 ; file not found
28693 00009903 0F852BE2FFFF        <1>       jne    loc_run_cmd_failed
28694                              <1>
28695 00009909 66A1[A25F0100]      <1>       mov    ax, word [EXE_ID]
28696 0000990F 80FC2E              <1>       cmp    ah, '.' ; File name has extension sign
28697 00009912 7424                <1>       je     short loc_run_check_auto_path
28698                              <1>
28699 00009914 08C0                <1>       or     al, al
28700 00009916 7520                <1>       jnz    short loc_run_check_auto_path
28701                              <1>
28702 00009918 80FC08              <1>       cmp    ah, 8 ; count of file name chars
28703 0000991B 771B                <1>       ja     short loc_run_check_auto_path
28704                              <1>
28705                              <1> loc_run_change_file_ext_to_prg:
28706 0000991D 0FB6DC              <1>       movzx  ebx, ah ; count of file name chars
28707 00009920 BE[CC5C0100]        <1>       mov    esi, FindFile_Name
28708 00009925 01F3                <1>       add    ebx, esi
28709                              <1>       ; 07/05/2016
28710 00009927 C7032E505247        <1>       mov    dword [ebx],  '.PRG'
28711 0000992D 66C705[A25F0100]50- <1>       mov    word [EXE_ID], 'P.'
28712 00009935 2E                  <1>
28713 00009936 EBBA                <1>       jmp    short loc_run_find_program_file_next
28714                              <1>
28715                              <1> loc_run_check_auto_path:
28716                              <1>       ; NOTE: /// 07/05/2016 ///
28717                              <1>       ; If the path is given, value of byte [Run_Manual_Path]
28718                              <1>       ; will not be ZERO. If so, file searching by using
28719                              <1>       ; Automatic Path (via 'PATH' environment variable)
28720                              <1>       ; will not be applicable, because the program file
28721                              <1>       ; is already/absolutely not found.
28722                              <1>
28723 00009938 A0[A15F0100]        <1>       mov    al, [Run_Manual_Path]
28724 0000993D 08C0                <1>       or     al, al
28725 0000993F 0F85C4E1FFFF        <1>       jnz    loc_cmd_failed
28726                              <1>
28727                              <1> loc_run_check_auto_path_again:
28728 00009945 66833D[A05F0100]FF  <1>       cmp    word [Run_Auto_Path], 0FFFFh
28729                              <1>              ; 0FFFFh = Not a valid run path (in ENV block)
28730 0000994D 0F83B6E1FFFF        <1>       jnb    loc_cmd_failed
28731                              <1>       ; xor  al, al
28732 00009953 BE[9F070100]        <1>       mov    esi, Cmd_Path ; 'PATH'
```

```
28733 00009958 BF[E6530100]        <1>       mov    edi, TextBuffer
28734 0000995D E848F9FFFF          <1>       call   get_environment_string
28735 00009962 730E                <1>       jnc    short loc_run_chk_filename_ext_again
28736 00009964 66C705[A05F0100]FF- <1>       mov    word [Run_Auto_Path], 0FFFFh ; invalid
28737 0000996C FF                  <1>
28738 0000996D E997E1FFFF          <1>       jmp    loc_cmd_failed
28739                              <1>
28740                              <1> loc_run_chk_filename_ext_again:
28741 00009972 89C1                <1>       mov    ecx, eax ; string length (with zero tail)
28742 00009974 49                  <1>       dec    ecx ; without zero tail
28743 00009975 66A1[A25F0100]      <1>       mov    ax, [EXE_ID]
28744 0000997B 80FC2E              <1>       cmp    ah, '.'
28745 0000997E 740E                <1>       je     short loc_run_chk_auto_path_pos
28746                              <1>
28747                              <1> loc_run_change_file_ext_to_noext_again:
28748 00009980 0FB6DC              <1>       movzx  ebx, ah
28749 00009983 BE[CC5C0100]        <1>       mov    esi, FindFile_Name
28750 00009988 01F3                <1>       add    ebx, esi
28751 0000998A 29C0                <1>       sub    eax, eax
28752 0000998C 8903                <1>       mov    [ebx], eax ; 0 ; erase extension (.PRG)
28753                              <1>
28754                              <1> loc_run_chk_auto_path_pos:
28755                              <1>       ;movzx eax,  word [Run_Auto_Path]
28756 0000998E 66A1[A05F0100]      <1>       mov    ax, [Run_Auto_Path]
28757 00009994 39C8                <1>       cmp    eax, ecx ; ecx = string length (except zero tail)
28758 00009996 0F836DE1FFFF        <1>       jnb    loc_cmd_failed
28759                              <1>       ;or    eax, eax
28760 0000999C 6609C0              <1>       or     ax, ax
28761 0000999F 7502                <1>       jnz    short loc_run_auto_path_pos_move
28762 000099A1 B005                <1>       mov    al, 5
28763                              <1>
28764                              <1> loc_run_auto_path_pos_move:
28765 000099A3 89FE                <1>       mov    esi, edi ; offset TextBuffer
28766 000099A5 01C6                <1>       add    esi, eax
28767                              <1>
28768                              <1> loc_run_auto_path_pos_space_loop:
28769 000099A7 AC                  <1>       lodsb
28770 000099A8 3C20                <1>       cmp    al, 20h
28771 000099AA 74FB                <1>       je     short loc_run_auto_path_pos_space_loop
28772 000099AC 0F8257E1FFFF        <1>       jb     loc_cmd_failed
28773 000099B2 AA                  <1>       stosb
28774                              <1> loc_run_auto_path_pos_move_next:
28775 000099B3 AC                  <1>       lodsb
28776 000099B4 3C3B                <1>       cmp    al, ';'
28777 000099B6 7414                <1>       je     short loc_run_auto_path_pos_move_last_byte
28778 000099B8 3C20                <1>       cmp    al, 20h
28779 000099BA 74F7                <1>       je     short loc_run_auto_path_pos_move_next
28780 000099BC 7203                <1>       jb     short loc_byte_ptr_end_of_path
28781 000099BE AA                  <1>       stosb
28782 000099BF EBF2                <1>       jmp    short loc_run_auto_path_pos_move_next
28783                              <1>
28784                              <1> loc_byte_ptr_end_of_path:
28785 000099C1 66C705[A05F0100]FF- <1>       mov    word [Run_Auto_Path], 0FFFFh ; end of path
28786 000099C9 FF                  <1>
28787 000099CA EB0D                <1>       jmp    short loc_run_auto_path_move_ok
28788                              <1>
28789                              <1> loc_run_auto_path_pos_move_last_byte:
28790 000099CC 89F0                <1>       mov    eax, esi
28791 000099CE 2D[E6530100]        <1>       sub    eax, TextBuffer
28792 000099D3 66A3[A05F0100]      <1>       mov    [Run_Auto_Path], ax ; next path position
28793                              <1>
28794                              <1> loc_run_auto_path_move_ok:
28795 000099D9 4F                  <1>       dec    edi
28796 000099DA B02F                <1>       mov    al, '/'
28797 000099DC 3807                <1>       cmp    [edi], al
28798 000099DE 7403                <1>       je     short loc_run_auto_path_move_file_name
28799 000099E0 47                  <1>       inc    edi
28800 000099E1 8807                <1>       mov    [edi], al
28801                              <1>
28802                              <1> loc_run_auto_path_move_file_name:
28803 000099E3 47                  <1>       inc    edi
28804 000099E4 BE[CC5C0100]        <1>       mov    esi, FindFile_Name
28805                              <1>
28806                              <1> loc_run_auto_path_move_fn_loop:
28807 000099E9 AC                  <1>       lodsb
28808 000099EA AA                  <1>       stosb
28809 000099EB 08C0                <1>       or     al, al
28810 000099ED 75FA                <1>       jnz    short loc_run_auto_path_move_fn_loop
28811                              <1>
28812 000099EF BE[E6530100]        <1>       mov    esi, TextBuffer
28813 000099F4 BF[8A5C0100]        <1>       mov    edi, FindFile_Drv
28814 000099F9 E833080000          <1>       call   parse_path_name
28815 000099FE 0F8205E1FFFF        <1>       jc     loc_cmd_failed
28816                              <1>
28817 00009A04 8A35[E6520100]      <1>       mov    dh, [Current_Drv]
28818 00009A0A 8A15[8A5C0100]      <1>       mov    dl, [FindFile_Drv]
28819 00009A10 38F2                <1>       cmp    dl, dh
28820 00009A12 740B                <1>       je     short loc_run_change_directory_again
28821                              <1>
28822 00009A14 E857D2FFFF          <1>       call   change_current_drive
28823 00009A19 0F8215E1FFFF        <1>       jc     loc_run_cmd_failed
28824                              <1>
28825                              <1> loc_run_change_directory_again:
28826 00009A1F 803D[8B5C0100]20    <1>       cmp    byte [FindFile_Directory], 20h
28827 00009A26 761D                <1>       jna    short loc_load_executable_cdir_chk_again
28828                              <1>
28829 00009A28 FE05[D3060100]      <1>       inc    byte [Restore_CDIR]
28830 00009A2E BE[8B5C0100]        <1>       mov    esi, FindFile_Directory
28831 00009A33 30E4                <1>       xor    ah, ah ; CD_COMMAND sign -> 0
28832 00009A35 E8E1010000          <1>       call   change_current_directory
28833 00009A3A 0F82F4E0FFFF        <1>       jc     loc_run_cmd_failed
28834                              <1>
28835                              <1> loc_run_chg_prompt_dir_str_again:
```

```
28836 00009A40 E8F6000000          <1>       call   change_prompt_dir_string
28837                              <1>
28838                              <1> loc_load_executable_cdir_chk_again:
28839 00009A45 A1[E0520100]        <1>       mov    eax, [Current_Dir_FCluster]
28840 00009A4A 3B05[9C5F0100]      <1>       cmp    eax, [Run_CDirFC]
28841 00009A50 0F8597FEFFFF        <1>       jne    loc_run_find_executable_file_next
28842 00009A56 30C0                <1>       xor    al, al ; 0
28843 00009A58 E9E8FEFFFF          <1>       jmp    loc_run_check_auto_path_again
28844                              <1>
28845                              <1> loc_load_and_run_file:
28846                              <1>       ; 13/11/2017
28847                              <1>       ; 04/01/2017
28848                              <1>       ; 23/04/2016
28849 00009A5D BE[CC5C0100]        <1>       mov    esi, FindFile_Name
28850 00009A62 BF[E6530100]        <1>       mov    edi, TextBuffer
28851                              <1>
28852                              <1>       ; 24/04/2016
28853 00009A67 31D2                <1>       xor    edx, edx
28854 00009A69 668915[4A040300]    <1>       mov    word [argc], dx ; 0
28855 00009A70 8915[8C030300]      <1>       mov    dword [u.nread], edx ; 0
28856                              <1>
28857                              <1> loc_load_and_run_file_1:
28858 00009A76 AC                  <1>       lodsb
28859 00009A77 AA                  <1>       stosb
28860 00009A78 FF05[8C030300]      <1>       inc    dword [u.nread]
28861 00009A7E 20C0                <1>       and    al, al
28862 00009A80 75F4                <1>       jnz    short loc_load_and_run_file_1
28863                              <1>
28864 00009A82 A0[47530100]        <1>       mov    al, [CmdArgStart]
28865 00009A87 20C0                <1>       and    al, al
28866 00009A89 7445                <1>       jz     short loc_load_and_run_file_7
28867                              <1>
28868 00009A8B 0FB6F0              <1>       movzx  esi, al ; 11/05/2016
28869 00009A8E B950000000          <1>       mov    ecx, 80
28870 00009A93 29F1                <1>       sub    ecx, esi
28871 00009A95 81C6[96530100]      <1>       add    esi, CommandBuffer
28872                              <1>
28873 00009A9B 66FF05[4A040300]    <1>       inc    word [argc] ; 11/05/2016
28874                              <1>
28875                              <1> loc_load_and_run_file_2:
28876 00009AA2 AC                  <1>       lodsb
28877 00009AA3 3C20                <1>       cmp    al, 20h
28878 00009AA5 7717                <1>       ja     short loc_load_and_run_file_5
28879 00009AA7 721E                <1>       jb     short loc_load_and_run_file_6
28880                              <1>
28881                              <1> loc_load_and_run_file_3:
28882 00009AA9 803E20              <1>       cmp    byte [esi], 20h
28883 00009AAC 7707                <1>       ja     short loc_load_and_run_file_4
28884 00009AAE 7217                <1>       jb     short loc_load_and_run_file_6
28885 00009AB0 46                  <1>       inc    esi
28886 00009AB1 E2F6                <1>       loop   loc_load_and_run_file_3
28887 00009AB3 EB12                <1>       jmp    short loc_load_and_run_file_6
28888                              <1>
28889                              <1> loc_load_and_run_file_4:
28890 00009AB5 28C0                <1>       sub    al, al ; 0
28891 00009AB7 66FF05[4A040300]    <1>       inc    word [argc]
28892                              <1> loc_load_and_run_file_5:
28893 00009ABE AA                  <1>       stosb
28894 00009ABF FF05[8C030300]      <1>       inc    dword [u.nread]
28895 00009AC5 E2DB                <1>       loop   loc_load_and_run_file_2
28896                              <1>
28897                              <1> loc_load_and_run_file_6:
28898 00009AC7 30C0                <1>       xor    al, al ; 0
28899 00009AC9 AA                  <1>       stosb
28900 00009ACA FF05[8C030300]      <1>       inc    dword [u.nread]
28901                              <1> loc_load_and_run_file_7:
28902 00009AD0 8807                <1>       mov    [edi], al ; 0
28903 00009AD2 66FF05[4A040300]    <1>       inc    word [argc] ; 24/04/2016
28904 00009AD9 FF05[8C030300]      <1>       inc    dword [u.nread] ; 24/04/2016
28905 00009ADF BE[E6530100]        <1>       mov    esi, TextBuffer
28906 00009AE4 8B15[F85C0100]      <1>       mov    edx, [FindFile_DirEntry+DirEntry_FileSize]
28907 00009AEA 66A1[F05C0100]      <1>       mov    ax, [FindFile_DirEntry+DirEntry_FstClusHI]
28908 00009AF0 C1E010              <1>       shl    eax, 16 ; 13/11/2017
28909 00009AF3 66A1[F65C0100]      <1>       mov    ax, [FindFile_DirEntry+DirEntry_FstClusLO]
28910                              <1>       ; EAX = First Cluster number
28911                              <1>       ; EDX = File Size
28912                              <1>       ; ESI = Argument list address
28913                              <1>       ; [argc] = argument count
28914                              <1>       ; [u.nread] = argument list length
28915 00009AF9 E8CD450000          <1>       call   load_and_run_file ; trdosk6.s
28916                              <1>         ;jc  loc_run_cmd_failed ; 04/01/2017
28917                              <1> loc_load_and_run_file_8: ; 06/05/2016
28918 00009AFE E938E9FFFF          <1>       jmp    loc_file_rw_restore_retn
28919                              <1>
28920                              <1> check_prg_filename_ext:
28921                              <1>       ; 23/04/2016 (TRDOS 386 = TRDOS v2.0)
28922                              <1>       ; 10/09/2011
28923                              <1>       ; (TRDOS v1, CMDINTR.ASM, 'proc_check_exe_filename_ext')
28924                              <1>       ; 14/11/2009
28925                              <1>       ; INPUT ->
28926                              <1>       ;     ESI = Dot File Name
28927                              <1>       ; OUTPUT ->
28928                              <1>       ;     cf = 0 -> EXE_ID in AL
28929                              <1>       ;      ESI = Last char + 1 position
28930                              <1>       ;     cf = 1 -> Invalid executable file name
28931                              <1>       ;      or no file name extension if AH<=8
28932                              <1>       ;      AL = Last file name char
28933                              <1>       ;     cf = 0 -> AL='P' (PRG), AL=0 (no extension)
28934                              <1>       ;
28935                              <1>       ; (Modified registers: EAX, ESI)
28936                              <1>
28937 00009B03 30E4                <1>       xor    ah, ah
28938                              <1> loc_run_check_filename_ext:
```

```
28939 00009B05 AC                <1>       lodsb
28940 00009B06 3C21              <1>       cmp    al, 21h
28941 00009B08 7229              <1>       jb     short loc_check_exe_fn_retn
28942 00009B0A FEC4              <1>       inc    ah
28943 00009B0C 3C2E              <1>       cmp    al, '.'
28944 00009B0E 75F5              <1>       jne    short loc_run_check_filename_ext
28945                            <1>
28946                            <1> loc_run_check_filename_ext_dot:
28947 00009B10 80FC02            <1>       cmp    ah, 2 ; .??? is not valid
28948 00009B13 88C4              <1>       mov    ah, al ; '.'
28949 00009B15 7219              <1>       jb     short loc_check_prg_fn_retn
28950                            <1>
28951                            <1> loc_run_check_filename_ext_dot_ok:
28952 00009B17 AC                <1>       lodsb
28953 00009B18 24DF              <1>       and    al, 0DFh
28954                            <1>
28955                            <1> loc_run_check_filename_ext_prg:
28956 00009B1A 3C50              <1>       cmp    al, 'P'
28957 00009B1C 7212              <1>       jb     short loc_check_prg_fn_retn
28958 00009B1E 7711              <1>       ja     short loc_check_prg_fn_stc
28959 00009B20 AC                <1>       lodsb
28960 00009B21 24DF              <1>       and    al, 0DFh
28961 00009B23 3C52              <1>       cmp    al, 'R'
28962 00009B25 750A              <1>       jne    short loc_check_prg_fn_stc
28963 00009B27 AC                <1>       lodsb
28964 00009B28 24DF              <1>       and    al, 0DFh
28965 00009B2A 3C47              <1>       cmp    al, 'G'
28966 00009B2C 7503              <1>       jne    short loc_check_prg_fn_stc
28967                            <1>
28968 00009B2E B050              <1>       mov    al, 'P'
28969                            <1> loc_check_prg_fn_retn:
28970 00009B30 C3                <1>       retn
28971                            <1>
28972                            <1> loc_check_prg_fn_stc:
28973 00009B31 F9                <1>       stc
28974 00009B32 C3                <1>       retn
28975                            <1>
28976                            <1> loc_check_exe_fn_retn:
28977 00009B33 28C0              <1>       sub    al, al ; 0
28978 00009B35 C3                <1>       retn
28979                            <1>
28980                            <1> find_and_list_files:
28981 00009B36 C3                <1>       retn
28982                            <1> set_exec_arguments:
28983 00009B37 C3                <1>       retn
28984                            <1> delete_fs_directory:
28985 00009B38 31C0              <1>       xor eax, eax
28986 00009B3A C3                <1>       retn
28987                                %include 'trdosk4.s' ; 24/01/2016
28988                            <1> ; ****************************************************************************
28989                            <1> ; TRDOS386.ASM (TRDOS 386 Kernel - v2.0.0) - Directory Functions : trdosk4.s
28990                            <1> ; ----------------------------------------------------------------------------
28991                            <1> ; Last Update: 09/12/2017
28992                            <1> ; ----------------------------------------------------------------------------
28993                            <1> ; Beginning: 24/01/2016
28994                            <1> ; ----------------------------------------------------------------------------
28995                            <1> ; Assembler: NASM version 2.11 (trdos386.s)
28996                            <1> ; ----------------------------------------------------------------------------
28997                            <1> ; Derived from TRDOS Operating System v1.0 (8086) source code by Erdogan Tan
28998                            <1> ; DIR.ASM (09/10/2011)
28999                            <1> ; ****************************************************************************
29000                            <1>
29001                            <1> ; DIR.ASM  [ TRDOS KERNEL - COMMAND EXECUTER SECTION - DIRECTORY FUNCTIONS ]
29002                            <1> ; (c) 2004-2010  Erdogan TAN  [ 17/01/2004 ]  Last Update: 09/10/2011
29003                            <1> ; FILE.ASM [ FILE FUNCTIONS ] Last Update: 09/10/2011
29004                            <1>
29005                            <1> change_prompt_dir_string:
29006                            <1>       ; 05/10/2016
29007                            <1>       ; 24/01/2016 (TRDOS 386 = TRDOS v2.0)
29008                            <1>       ; 27/03/2011
29009                            <1>       ; 09/10/2009
29010                            <1>       ; INPUT/OUTPUT => none
29011                            <1>       ; this procedure changes current directory string/text
29012                            <1>       ; 2005
29013                            <1>
29014 00009B3B BE[475B0100]      <1>       mov    esi, PATH_Array
29015                            <1> change_prompt_dir_str: ; 05/10/2016 (call from 'set_working_path')
29016 00009B40 BF[EA520100]      <1>       mov    edi, Current_Directory
29017 00009B45 8A25[E4520100]    <1>       mov    ah, [Current_Dir_Level]
29018 00009B4B E807000000        <1>       call   set_current_directory_string
29019 00009B50 880D[45530100]    <1>       mov    [Current_Dir_StrLen], cl
29020                            <1>
29021 00009B56 C3                <1>       retn
29022                            <1>
29023                            <1> set_current_directory_string:
29024                            <1>       ; 24/01/2016 (TRDOS 386 = TRDOS v2.0)
29025                            <1>       ; 27/03/2011
29026                            <1>       ; 09/10/2009
29027                            <1>       ; INPUT:
29028                            <1>       ;    ESI = Path Array Address
29029                            <1>       ;    EDI = Current Directory String Buffer
29030                            <1>       ;    AH = Current Directory Level
29031                            <1>       ; OUTPUT => EAX, EBX, ESI will be changed
29032                            <1>       ;    EDI will be same with input
29033                            <1>       ;    ECX = Current Directory String Length
29034                            <1>
29035 00009B57 57                <1>       push    edi
29036 00009B58 80FC00            <1>       cmp     ah, 0
29037 00009B5B 7652              <1>       jna    short pass_write_path
29038 00009B5D 83C610            <1>       add    esi, 16
29039 00009B60 89F3              <1>       mov    ebx, esi
29040                            <1> loc_write_path:
29041 00009B62 B908000000        <1>       mov    ecx, 8
```

```
29042                                  <1> path_write_dirname1:
29043 00009B67 AC                      <1>       lodsb
29044 00009B68 3C20                    <1>       cmp    al, 20h
29045 00009B6A 7612                    <1>       jna    short pass_write_dirname1
29046 00009B6C AA                      <1>       stosb
29047 00009B6D 81FF[44530100]          <1>       cmp    edi, End_Of_Current_Dir_Str
29048 00009B73 733A                    <1>       jnb    short pass_write_path
29049 00009B75 E2F0                    <1>       loop   path_write_dirname1
29050 00009B77 803E20                  <1>       cmp    byte [esi], 20h
29051 00009B7A 7624                    <1>       jna    short pass_write_dirname2
29052 00009B7C EB0A                    <1>       jmp     short loc_put_dot_cont_ext
29053                                  <1> pass_write_dirname1:
29054 00009B7E 89DE                    <1>       mov    esi, ebx
29055 00009B80 83C608                  <1>       add    esi, 8
29056 00009B83 803E20                  <1>       cmp    byte [esi], 20h
29057 00009B86 7618                    <1>       jna    short pass_write_dirname2
29058                                  <1> loc_put_dot_cont_ext:
29059 00009B88 C6072E                  <1>       mov    byte [edi], "."
29060                                  <1>       ;mov   ecx, 3
29061 00009B8B B103                    <1>       mov    cl, 3
29062                                  <1> loc_check_dir_name_ext:
29063 00009B8D AC                      <1>       lodsb
29064 00009B8E 47                      <1>       inc    edi
29065 00009B8F 3C20                    <1>       cmp    al, 20h
29066 00009B91 760D                    <1>       jna    short pass_write_dirname2
29067 00009B93 8807                    <1>       mov    [edi], al
29068 00009B95 81FF[44530100]          <1>       cmp    edi, End_Of_Current_Dir_Str
29069 00009B9B 7312                    <1>       jnb    short pass_write_path
29070 00009B9D E2EE                    <1>       loop    loc_check_dir_name_ext
29071 00009B9F 47                      <1>       inc    edi
29072                                  <1> pass_write_dirname2:
29073 00009BA0 FECC                    <1>       dec    ah
29074 00009BA2 740B                    <1>       jz      short pass_write_path
29075 00009BA4 83C310                  <1>       add    ebx, 16
29076 00009BA7 89DE                    <1>       mov    esi, ebx
29077 00009BA9 C6072F                  <1>       mov    byte [edi],"/"
29078 00009BAC 47                      <1>       inc    edi
29079 00009BAD EBB3                    <1>       jmp    short loc_write_path
29080                                  <1> pass_write_path:
29081 00009BAF C60700                  <1>       mov    byte [edi], 0
29082 00009BB2 47                      <1>       inc    edi
29083 00009BB3 89F9                    <1>       mov    ecx, edi
29084 00009BB5 5F                      <1>       pop    edi
29085 00009BB6 29F9                    <1>       sub    ecx, edi
29086                                  <1>       ; ECX = Current Directory String Length
29087 00009BB8 C3                      <1>       retn
29088                                  <1>
29089                                  <1> get_current_directory:
29090                                  <1>       ; 15/10/2016
29091                                  <1>       ; 14/02/2016
29092                                  <1>       ; 24/01/2016 (TRDOS 386 = TRDOS v2.0)
29093                                  <1>       ; 27/03/2011
29094                                  <1>       ;
29095                                  <1>       ; INPUT-> ESI = Current Directory Buffer
29096                                  <1>       ;         DL = TRDOS Logical Dos Drive Number + 1
29097                                  <1>       ;              (0= Default/Current Drive)
29098                                  <1>       ;
29099                                  <1>       ;   Note: Required dir buffer length may be <= 92 bytes
29100                                  <1>       ;         for TRDOS (7*12 name chars + 7 slash + 0)
29101                                  <1>       ; OUTPUT ->  ESI = Current Directory Buffer
29102                                  <1>       ;            EAX, EBX, ECX, EDX, EDI will be changed
29103                                  <1>       ;            CX/CL = Current Directory String Length
29104                                  <1>       ;          DL = Drive Number (0 based)
29105                                  <1>       ;            (If input is 0, output is current drv number)
29106                                  <1>       ;            DH = same with input
29107                                  <1>       ;   cf = 0 -> AL = 0
29108                                  <1>       ;   cf = 1 -> error code in AL
29109                                  <1>
29110                                  <1> loc_get_current_drive_0:
29111 00009BB9 80FA00                  <1>       cmp    dl, 0
29112 00009BBC 7708                    <1>       ja     short loc_get_current_drive_1
29113 00009BBE 8A15[E6520100]          <1>       mov    dl, [Current_Drv]
29114 00009BC4 EB17                    <1>       jmp    short loc_get_current_drive_2
29115                                  <1> loc_get_current_drive_1:
29116 00009BC6 FECA                    <1>       dec    dl
29117 00009BC8 3A15[D2060100]          <1>       cmp    dl, [Last_DOS_DiskNo]
29118 00009BCE 760D                    <1>       jna    short loc_get_current_drive_2
29119 00009BD0 B80F000000              <1>       mov    eax, 0Fh ; Invalid drive (Drive not ready!)
29120 00009BD5 F5                      <1>       cmc    ; stc
29121 00009BD6 C3                      <1>       retn
29122                                  <1>
29123                                  <1> loc_get_current_drive_not_ready_retn:
29124 00009BD7 5E                      <1>       pop    esi
29125                                  <1>       ;mov   eax, 15
29126 00009BD8 66B80F00                <1>       mov    ax, 15 ; Drive not ready
29127 00009BDC C3                      <1>       retn
29128                                  <1>
29129                                  <1> loc_get_current_drive_2:
29130 00009BDD 31C0                    <1>       xor    eax, eax
29131 00009BDF 88D4                    <1>       mov    ah, dl
29132 00009BE1 56                      <1>       push   esi
29133 00009BE2 BE00010900              <1>       mov    esi, Logical_DOSDisks
29134 00009BE7 01C6                    <1>       add    esi, eax
29135 00009BE9 8A06                    <1>       mov    al, [esi+LD_Name]
29136 00009BEB 3C41                    <1>       cmp    al, 'A'
29137 00009BED 72E8                    <1>       jb     short loc_get_current_drive_not_ready_retn
29138                                  <1>
29139 00009BEF 8A667F                  <1>       mov    ah, [esi+LD_CDirLevel]
29140 00009BF2 08E4                    <1>       or     ah, ah
29141 00009BF4 7506                    <1>       jnz    short loc_get_current_drive_3
29142                                  <1>
29143                                  <1>       ;xor   ah, ah ; mov ah, 0
29144 00009BF6 8826                    <1>       mov    [esi], ah
```

```
29145 00009BF8 31C9                <1>       xor    ecx, ecx
29146 00009BFA EB1C                <1>       jmp    short loc_get_current_drive_4
29147                              <1>
29148                              <1> loc_get_current_drive_3:
29149 00009BFC BF[475B0100]        <1>         mov     edi, PATH_Array
29150 00009C01 57                  <1>       push   edi
29151 00009C02 81C680000000        <1>       add    esi, LD_CurrentDirectory
29152 00009C08 B920000000          <1>       mov    ecx, 32
29153 00009C0D F3A5                <1>       rep    movsd
29154 00009C0F 5E                  <1>       pop    esi ; Path Array Address
29155 00009C10 5F                  <1>       pop    edi ; pushed esi (current dir buffer offset)
29156                              <1>       ;
29157 00009C11 E841FFFFFF          <1>       call   set_current_directory_string
29158 00009C16 89FE                <1>       mov    esi, edi
29159                              <1>
29160                              <1> loc_get_current_drive_4:
29161 00009C18 30C0                <1>       xor    al, al
29162 00009C1A C3                  <1>       retn
29163                              <1>
29164                              <1> change_current_directory:
29165                              <1>       ; 19/02/2016
29166                              <1>       ; 11/02/2016
29167                              <1>       ; 10/02/2016
29168                              <1>       ; 08/02/2016
29169                              <1>       ; 06/02/2016 (TRDOS 386 = TRDOS v2.0)
29170                              <1>       ; 18/09/2011 (DIR.ASM, 09/10/2011)
29171                              <1>       ; 04/10/2009
29172                              <1>       ; 2005
29173                              <1>       ; INPUT ->
29174                              <1>       ;      ESI = Directory string
29175                              <1>       ;      ah = CD command (CDh = save current dir string)
29176                              <1>       ; OUTPUT ->
29177                              <1>       ;      EDI = DOS Drive Description Table
29178                              <1>       ;      cf = 1 -> error
29179                              <1>       ;         EAX = Error code
29180                              <1>       ;      cf = 0 -> succesful
29181                              <1>       ;         ESI = PATH_Array
29182                              <1>       ;         EAX = Current Directory First Cluster
29183                              <1>       ;
29184                              <1>       ; (EAX, EBX, ECX, EDX, ESI, EDI will be changed)
29185                              <1>
29186 00009C1B 8825[D55B0100]      <1>       mov    [CD_COMMAND], ah
29187 00009C21 803E2F              <1>       cmp    byte [esi], '/'
29188 00009C24 7505                <1>       jne    short loc_ccd_cdir_level
29189 00009C26 46                  <1>       inc    esi
29190 00009C27 30C0                <1>       xor    al, al
29191 00009C29 EB05                <1>       jmp    short loc_ccd_parse_path_name
29192                              <1> loc_ccd_cdir_level:
29193 00009C2B A0[E4520100]        <1>       mov    al, [Current_Dir_Level]
29194                              <1> loc_ccd_parse_path_name:
29195 00009C30 88C4                <1>       mov    ah, al
29196 00009C32 BF[475B0100]        <1>       mov    edi, PATH_Array
29197                              <1>
29198                              <1> ; Reset directory levels > cdir level
29199                              <1>       ; is this required !?
29200                              <1>       ;
29201                              <1>       ; Relations:
29202                              <1>       ; MAINPROG.ASM (pass_ccdrv_reset_cdir_FAT_fcluster)
29203                              <1>       ; proc_parse_dir_name,
29204                              <1>       ; proc_change_current_directory (this procedure)
29205                              <1>       ; proc_change_prompt_dir_string
29206                              <1>
29207 00009C37 0FB6C8             <1>       movzx  ecx, al
29208 00009C3A FEC1                <1>       inc    cl
29209 00009C3C C0E104              <1>       shl    cl, 4
29210 00009C3F 01CF                <1>       add    edi, ecx
29211 00009C41 B107                <1>       mov    cl, 7
29212 00009C43 28C1                <1>       sub    cl, al
29213 00009C45 C0E102              <1>       shl    cl, 2
29214 00009C48 89C3                <1>       mov    ebx, eax
29215 00009C4A 31C0                <1>       xor    eax, eax ; 0
29216 00009C4C F3AB                <1>       rep    stosd
29217 00009C4E 89D8                <1>       mov    eax, ebx
29218                              <1>
29219 00009C50 BF[475B0100]        <1>       mov    edi, PATH_Array
29220                              <1>
29221 00009C55 803E20              <1>       cmp    byte [esi], 20h
29222 00009C58 F5                  <1>       cmc
29223 00009C59 7305                <1>       jnc    short pass_ccd_parse_dir_name
29224                              <1>
29225                              <1>             ; ESI = Path name
29226                              <1>             ; AL = CCD_Level
29227 00009C5B E872010000          <1>         call    parse_dir_name
29228                              <1>             ; AL = CCD_Level
29229                              <1>             ; AH = Last_Dir_Level
29230                              <1>             ; (EDI = PATH_Array)
29231                              <1>
29232                              <1> pass_ccd_parse_dir_name:
29233 00009C60 9C                  <1>       pushf
29234                              <1>
29235                              <1>       ;mov   [CCD_Level], al
29236                              <1>         ;mov [Last_Dir_Level], ah
29237 00009C61 66A3[CB5B0100]      <1>       mov    [CCD_Level], ax
29238                              <1>
29239 00009C67 31DB                <1>       xor    ebx, ebx
29240 00009C69 8A3D[E6520100]      <1>       mov    bh, [Current_Drv]
29241 00009C6F BE00010900          <1>       mov    esi, Logical_DOSDisks
29242 00009C74 01DE                <1>       add    esi, ebx
29243                              <1>
29244 00009C76 9D                  <1>       popf
29245 00009C77 720A                <1>       jc     short loc_ccd_bad_path_name_retn
29246                              <1>
29247 00009C79 8935[C75B0100]      <1>       mov    [CCD_DriveDT], esi
```

```
                                <1>
00009C7F 3C07                   <1>      cmp    al, 7
00009C81 7209                   <1>      jb     short loc_ccd_load_child_dir
                                <1>
                                <1> loc_ccd_bad_path_name_retn:
00009C83 87F7                   <1>      xchg   esi, edi
00009C85 B813000000             <1>      mov    eax, 19 ; Bad directory/path name
00009C8A F9                     <1>      stc
                                <1> loc_ccd_retn_p:
00009C8B C3                     <1>      retn
                                <1>
                                <1> loc_ccd_load_child_dir:
                                <1>      ; AL = CCD_Level
00009C8C 08C0                   <1>      or     al, al
00009C8E 7468                   <1>      jz     short loc_ccd_load_root_dir
                                <1>
00009C90 6689C1                 <1>      mov    cx, ax
00009C93 C0E004                 <1>      shl    al, 4
00009C96 0FB6F0                 <1>      movzx  esi, al
00009C99 01FE                   <1>      add    esi, edi  ; offset PATH_Array
                                <1>
00009C9B 8B460C                 <1>      mov    eax, [esi+12]
00009C9E 38E9                   <1>      cmp    cl, ch
00009CA0 0F84FA000000           <1>        je      loc_ccd_load_sub_directory
00009CA6 A3[E0520100]           <1>      mov    [Current_Dir_FCluster], eax
                                <1>
                                <1> loc_ccd_load_child_dir_next:
00009CAB 83C610                 <1>      add    esi, 16 ; DOS DirEntry Format FileName Address
                                <1>
                                <1>      ; Directory attribute : 10h
00009CAE B010                   <1>      mov    al, 00010000b ; 10h (Attrib AND mask)
                                <1>      ;mov   ah, 11001000b ; C8h
                                <1>      ; Volume name attribute: 8h
00009CB0 B408                   <1>      mov    ah, 00001000b ; 08h (Attrib NAND, AND --> zero mask)
                                <1>
00009CB2 6631C9                 <1>      xor    cx, cx
00009CB5 E8B5010000             <1>      call   locate_current_dir_file
00009CBA 7353                   <1>      jnc    short loc_ccd_set_dir_cluster_ptr
                                <1>
                                <1>       ; 19/02/2016
                                <1>      ;mov   edi, [CCD_DriveDT]
00009CBC 8A25[CB5B0100]         <1>      mov    ah, [CCD_Level]
00009CC2 803D[D55B0100]CD       <1>      cmp    byte [CD_COMMAND], 0CDh ;'CD' command or another
00009CC9 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.)
00009CCB 88E1                   <1>      mov    cl, ah
00009CCD 50                     <1>      push   eax
00009CCE E8E3000000             <1>      call   loc_ccd_save_current_dir
00009CD3 58                     <1>      pop    eax
                                <1> loc_ccd_load_child_dir_err:
00009CD4 3C03                   <1>      cmp    al, 3  ; AL = 2 => File not found error
00009CD6 7202                   <1>      jb     short loc_ccd_path_not_found_retn
00009CD8 F9                     <1>      stc
00009CD9 C3                     <1>      retn
                                <1>
                                <1> loc_ccd_path_not_found_retn:
00009CDA B003                   <1>      mov    al, 3  ; Path not found
00009CDC C3                     <1>      retn
                                <1>
                                <1> loc_ccd_load_FAT_root_dir:
00009CDD 803D[E5520100]02       <1>      cmp    byte [Current_FATType], 2
00009CE4 776B                   <1>      ja     short loc_ccd_load_FAT32_root_dir
                                <1>
                                <1>      ;mov   esi, [CCD_DriveDT]
                                <1>      ;push  esi
00009CE6 E8B71D0000             <1>      call   load_FAT_root_directory
                                <1>      ;pop   edi ; Dos Drv Description Table
                                <1>
00009CEB 89F7                   <1>      mov    edi, esi
00009CED BE[475B0100]           <1>      mov    esi, PATH_Array
00009CF2 7297                   <1>      jc     short loc_ccd_retn_p
                                <1>
00009CF4 31C0                   <1>      xor    eax, eax
00009CF6 EB78                   <1>        jmp  short loc_ccd_set_cdfc
                                <1>
                                <1> loc_ccd_load_root_dir:
00009CF8 803D[E5520100]01       <1>      cmp    byte [Current_FATType], 1
00009CFF 73DC                   <1>      jnb    short loc_ccd_load_FAT_root_dir
                                <1>
                                <1> loc_ccd_load_FS_root_dir:
00009D01 E8631E0000             <1>      call   load_FS_root_directory
00009D06 EB5C                   <1>      jmp    short pass_ccd_load_FAT_sub_directory
                                <1>
                                <1> loc_ccd_load_FS_sub_directory_next:
00009D08 E85D1E0000             <1>      call   load_FS_sub_directory
00009D0D EB1F                   <1>      jmp    short pass_ccd_set_dir_cluster_ptr
                                <1>
                                <1> loc_ccd_set_dir_cluster_ptr:
                                <1>      ; EDI = Directory Entry
00009D0F 668B4714               <1>      mov    ax, [edi+20] ; First Cluster High Word
00009D13 C1E010                 <1>      shl    eax, 16
00009D16 668B471A               <1>      mov    ax, [edi+26] ; First Cluster Low Word
                                <1>
00009D1A 8B35[C75B0100]         <1>      mov    esi, [CCD_DriveDT]
00009D20 803D[E5520100]01       <1>      cmp    byte [Current_FATType], 1
00009D27 72DF                   <1>      jb     short loc_ccd_load_FS_sub_directory_next
                                <1>      ;push  esi
00009D29 E8FF1D0000             <1>      call   load_FAT_sub_directory
                                <1>      ;pop   edi ; Dos Drv Description Table
                                <1>
```

```
29351                                  <1> pass_ccd_set_dir_cluster_ptr:
29352                                  <1>       ;mov   edi, esi
29353 00009D2E BE[475B0100]            <1>       mov    esi, PATH_Array
29354 00009D33 7264                    <1>       jc     short loc_ccd_retn_c
29355                                  <1>
29356 00009D35 A1[155B0100]            <1>       mov    eax, [DirBuff_Cluster]
29357                                  <1>
29358 00009D3A FE05[CB5B0100]          <1>       inc    byte [CCD_Level]
29359 00009D40 0FB61D[CB5B0100]        <1>       movzx  ebx, byte [CCD_Level]
29360 00009D47 C0E304                  <1>       shl    bl, 4 ; * 16 (<= 128)
29361 00009D4A 01DE                    <1>       add    esi, ebx ; 19/02/2016
29362 00009D4C 89460C                  <1>       mov    [esi+12], eax
29363 00009D4F EB1F                    <1>       jmp    short loc_ccd_set_cdfc
29364                                  <1>
29365                                  <1> loc_ccd_load_FAT32_root_dir:
29366 00009D51 BE[475B0100]            <1>       mov    esi, PATH_Array
29367 00009D56 8B460C                  <1>       mov    eax, [esi+12]
29368 00009D59 8B35[C75B0100]          <1>       mov    esi, [CCD_DriveDT]
29369                                  <1>
29370                                  <1> loc_ccd_load_FAT_sub_directory:
29371                                  <1>       ;push  esi
29372 00009D5F E8C91D0000              <1>       call   load_FAT_sub_directory
29373                                  <1>       ;pop   edi ; Dos Drv Description Table
29374                                  <1>
29375                                  <1> pass_ccd_load_FAT_sub_directory:
29376                                  <1>       ;mov   edi, esi
29377 00009D64 BE[475B0100]            <1>       mov    esi, PATH_Array
29378 00009D69 722E                    <1>       jc     short loc_ccd_retn_c
29379                                  <1>
29380 00009D6B A1[155B0100]            <1>       mov    eax, [DirBuff_Cluster]
29381                                  <1>
29382                                  <1> loc_ccd_set_cdfc:
29383 00009D70 8A0D[CB5B0100]          <1>       mov    cl, [CCD_Level]
29384 00009D76 880D[E4520100]          <1>       mov    [Current_Dir_Level], cl
29385 00009D7C A3[E0520100]            <1>       mov    [Current_Dir_FCluster], eax
29386                                  <1>
29387 00009D81 8A2D[CC5B0100]          <1>       mov    ch, [Last_Dir_Level]
29388 00009D87 38E9                    <1>       cmp    cl, ch
29389 00009D89 0F821CFFFFFF            <1>       jb     loc_ccd_load_child_dir_next
29390                                  <1>
29391 00009D8F 803D[D55B0100]CD        <1>       cmp    byte [CD_COMMAND], 0CDh ;'CD' command or another
29392 00009D96 741E                    <1>       je     short loc_ccd_save_current_dir
29393                                  <1>
29394                                  <1>         ; jne -> don't save, restore (the previous cdir) later !
29395                                  <1>         ; (saving the cdir would prevent previous cdir restoration!)
29396                                  <1>
29397 00009D98 F8                      <1>       clc
29398                                  <1>
29399                                  <1> loc_ccd_retn_c:
29400 00009D99 8B3D[C75B0100]          <1>       mov    edi, [CCD_DriveDT]
29401 00009D9F C3                      <1>       retn
29402                                  <1>
29403                                  <1> loc_ccd_load_sub_directory:
29404 00009DA0 8B35[C75B0100]          <1>       mov    esi, [CCD_DriveDT]
29405 00009DA6 803D[E5520100]01        <1>       cmp    byte [Current_FATType], 1
29406 00009DAD 73B0                    <1>       jnb    short loc_ccd_load_FAT_sub_directory
29407 00009DAF E8B61D0000              <1>       call   load_FS_sub_directory
29408 00009DB4 EBAE                    <1>       jmp    short pass_ccd_load_FAT_sub_directory
29409                                  <1>
29410                                  <1> loc_ccd_save_current_dir:
29411 00009DB6 BE[475B0100]            <1>       mov    esi, PATH_Array ; 19/02/2016
29412 00009DBB 8B3D[C75B0100]          <1>       mov    edi, [CCD_DriveDT]
29413 00009DC1 57                      <1>       push   edi
29414 00009DC2 83C77F                  <1>         add     edi, LD_CDirLevel
29415 00009DC5 880F                    <1>       mov    [edi], cl
29416 00009DC7 47                      <1>       inc    edi ; LD_CurrentDirectory
29417 00009DC8 56                      <1>       push   esi
29418                                  <1>       ;mov   ecx, 32  ; always < 65536 (in this procedure)
29419 00009DC9 66B92000                <1>       mov    cx, 32
29420 00009DCD F3A5                    <1>       rep    movsd
29421                                  <1>       ; Current directory has been saved to
29422                                  <1>       ; the DOS drive description table, cdir area !
29423 00009DCF 5E                      <1>       pop    esi  ; PATH_Array
29424 00009DD0 5F                      <1>       pop    edi  ; Dos Drv Description Table
29425                                  <1>
29426 00009DD1 C3                      <1>       retn
29427                                  <1>
29428                                  <1> parse_dir_name:
29429                                  <1>       ; 11/02/2016
29430                                  <1>       ; 10/02/2016
29431                                  <1>       ; 07/02/2016 (TRDOS 386 = TRDOS v2.0)
29432                                  <1>       ; 18/09/2011
29433                                  <1>       ; 17/10/2009
29434                                  <1>       ; INPUT ->
29435                                  <1>       ;     ESI = ASCIIZ Directory String Address
29436                                  <1>       ;     AL = Current Directory Level
29437                                  <1>       ;     EDI = Destination Adress
29438                                  <1>       ;          (8 levels, each one 12+4 byte)
29439                                  <1>       ; OUTPUT ->
29440                                  <1>       ;     EDI = Dir Entry Formatted Array
29441                                  <1>       ;          with zero cluster pointer at the last level
29442                                  <1>       ;     AH = Last Dir Level
29443                                  <1>       ;     AL = Current Dir Level
29444                                  <1>       ;
29445                                  <1>       ; (esi, ebx, ecx will be changed)
29446                                  <1>
29447                                  <1>       ;mov   [PATH_Array_Ptr], edi
29448 00009DD2 88C4                    <1>       mov    ah, al
29449 00009DD4 66A3[6C5C0100]          <1>       mov    [PATH_CDLevel], ax
29450                                  <1> repeat_ppdn_check_slash:
29451 00009DDA AC                      <1>       lodsb
29452 00009DDB 3C2F                    <1>       cmp    al, '/'
29453 00009DDD 74FB                    <1>       je     short repeat_ppdn_check_slash
```

```
29454 00009DDF 3C21              <1>       cmp    al, 21h
29455 00009DE1 7219              <1>       jb     short loc_ppdn_retn
29456 00009DE3 57                <1>       push   edi
29457                            <1> loc_ppdn_get_dir_name:
29458 00009DE4 B90C000000        <1>       mov    ecx, 12
29459 00009DE9 BF[6E5C0100]      <1>       mov    edi, Dir_File_Name
29460                            <1> repeat_ppdn_get_dir_name:
29461 00009DEE AA                <1>       stosb
29462 00009DEF AC                <1>       lodsb
29463 00009DF0 3C2F              <1>       cmp    al, '/'
29464 00009DF2 740A              <1>       je     short loc_check_level_dot_conv_dir_name
29465 00009DF4 3C20              <1>       cmp    al, 20h
29466 00009DF6 7605              <1>       jna    short loc_ppdn_end_of_path_scan
29467 00009DF8 E2F4              <1>       loop   repeat_ppdn_get_dir_name
29468 00009DFA 5F                <1>       pop    edi
29469 00009DFB F9                <1>       stc
29470                            <1> loc_ppdn_retn:
29471 00009DFC C3                <1>       retn
29472                            <1>
29473                            <1> loc_ppdn_end_of_path_scan:
29474 00009DFD 4E                <1>       dec    esi
29475                            <1> loc_check_level_dot_conv_dir_name:
29476 00009DFE 31C0              <1>       xor    eax, eax
29477 00009E00 AA                <1>       stosb
29478 00009E01 89F3              <1>       mov    ebx, esi
29479 00009E03 BE[6E5C0100]      <1>       mov    esi, Dir_File_Name
29480 00009E08 AC                <1>       lodsb
29481                            <1> repeat_ppdn_name_check_dot:
29482 00009E09 3C2E              <1>       cmp    al, '.'
29483 00009E0B 7509              <1>       jne    short loc_ppdn_convert_sub_dir_name
29484                            <1> repeat_ppdn_name_dot_dot:
29485 00009E0D AC                <1>       lodsb
29486 00009E0E 3C2E              <1>       cmp    al, '.'
29487 00009E10 743E              <1>       je     short loc_ppdn_dot_dot
29488 00009E12 3C21              <1>       cmp    al, 21h
29489 00009E14 7226              <1>       jb     short pass_ppdn_convert_sub_dir_name
29490                            <1> loc_ppdn_convert_sub_dir_name:
29491 00009E16 8A25[6D5C0100]    <1>       mov    ah, [PATH_Level]
29492 00009E1C 80FC07            <1>       cmp    ah, 7
29493 00009E1F 731B              <1>       jnb    short pass_ppdn_convert_sub_dir_name
29494 00009E21 FEC4              <1>       inc    ah
29495 00009E23 8825[6D5C0100]    <1>       mov    [PATH_Level], ah
29496 00009E29 BE[6E5C0100]      <1>       mov    esi, Dir_File_Name
29497                            <1>       ;mov   edi, [PATH_Array_Ptr]
29498 00009E2E B010              <1>       mov    al, 16
29499 00009E30 F6E4              <1>       mul    ah
29500 00009E32 8B3C24            <1>       mov    edi, [esp]
29501                            <1>       ;push  edi
29502 00009E35 01C7              <1>       add    edi, eax
29503 00009E37 E82A030000        <1>       call   convert_file_name
29504                            <1>       ;pop   edi
29505                            <1> pass_ppdn_convert_sub_dir_name:
29506 00009E3C 89DE              <1>       mov    esi, ebx
29507                            <1> repeat_ppdn_check_last_slash:
29508 00009E3E AC                <1>       lodsb
29509 00009E3F 3C2F              <1>       cmp    al, '/'
29510 00009E41 74FB              <1>       je     short repeat_ppdn_check_last_slash
29511 00009E43 3C21              <1>       cmp    al, 21h
29512 00009E45 739D              <1>       jnb    short loc_ppdn_get_dir_name
29513                            <1> end_of_parse_dir_name:
29514 00009E47 5F                <1>       pop    edi
29515 00009E48 F5                <1>       cmc
29516                            <1>       ;mov   al, [PATH_CDLevel]
29517                            <1>       ;mov   ah, [PATH_Level]
29518 00009E49 66A1[6C5C0100]    <1>       mov    ax, [PATH_CDLevel]
29519 00009E4F C3                <1>       retn
29520                            <1>
29521                            <1> loc_ppdn_dot_dot:
29522 00009E50 AC                <1>       lodsb
29523 00009E51 3C21              <1>       cmp    al, 21h
29524 00009E53 73F2              <1>       jnb    short end_of_parse_dir_name
29525                            <1> loc_ppdn_dot_dot_prev_level:
29526 00009E55 66A1[6C5C0100]    <1>       mov    ax, [PATH_CDLevel]
29527 00009E5B 80EC01            <1>       sub    ah, 1
29528 00009E5E 80D400            <1>       adc    ah, 0
29529 00009E61 38E0              <1>       cmp    al, ah
29530 00009E63 7602              <1>       jna    short pass_ppdn_set_al_to_ah
29531 00009E65 88E0              <1>       mov    al, ah
29532                            <1> pass_ppdn_set_al_to_ah:
29533 00009E67 66A3[6C5C0100]    <1>       mov    [PATH_CDLevel], ax
29534 00009E6D EBCD              <1>       jmp    short pass_ppdn_convert_sub_dir_name
29535                            <1>
29536                            <1> locate_current_dir_file:
29537                            <1>       ; 20/11/2017
29538                            <1>       ; 14/02/2016
29539                            <1>       ; 13/02/2016
29540                            <1>       ; 10/02/2016
29541                            <1>       ; 06/02/2016 (TRDOS 386 = TRDOS v2.0)
29542                            <1>       ; 14/08/2010
29543                            <1>       ; 19/09/2009
29544                            <1>         ; 2005
29545                            <1>       ; INPUT ->
29546                            <1>       ;     ESI = DOS DirEntry Format FileName Address
29547                            <1>       ;     AL = Attributes Mask
29548                            <1>       ;     (<AL AND EntryAttrib> must be equal to AL)
29549                            <1>       ;     AH = Negative Attributes Mask (If AH>0)
29550                            <1>       ;     (<AH AND EntryAttrib> must be ZERO)
29551                            <1>       ;     CH > 0 Find First Free Dir Entry or Deleted Entry
29552                            <1>       ;     CL = 0 -> Return the First Free Dir Entry
29553                            <1>       ;     CL = E5h -> Return the 1st deleted entry
29554                            <1>       ;     CL = FFh -> Return the 1st deleted or free entry
29555                            <1>       ;     CL > 0 and CL <> E5h and CL <> FFh -> Return the first
29556                            <1>       ;          proper entry (which fits with Atributes Masks)
```

```
29557                                  <1>       ;     CX = 0 Find Valid File/Directory/VolumeName
29558                                  <1>       ;     ? = Any One Char
29559                                  <1>       ;     * = Every Chars
29560                                  <1>       ; OUTPUT ->
29561                                  <1>       ;     EDI = Directory Entry Address (in Directory Buffer)
29562                                  <1>       ;     ESI = DOS DirEntry Format FileName Address
29563                                  <1>       ;     CF = 0 -> No Error, Proper Entry,
29564                                  <1>       ;     DL = Attributes
29565                                  <1>       ;     DH = Previous Entry Attr (LongName Check)
29566                                  <1>       ;     AL > 0 -> Ambiguous filename wildcard "?" used
29567                                  <1>       ;     AH > 0 -> Ambiguous filename wildcard "*" used
29568                                  <1>       ;     AX = 0 -> Filename full fits with directory entry
29569                                  <1>       ;     CH = The 1st Name Char of Current Dir Entry
29570                                  <1>       ;     CF = 1 -> Proper entry not found, Error Code in EAX/AL
29571                                  <1>       ;     CL = 0 and CH = 0 -> Free Entry (End Of Dir)
29572                                  <1>       ;     CL = 0 and CH = E5h -> Deleted Entry fits with filters
29573                                  <1>       ;     CL > 0 -> Entry not found, CH invalid
29574                                  <1>       ;     CF = 0 ->
29575                                  <1>       ;     EBX = Current Directory Entry Index/Number (BX)
29576                                  <1>
29577                                  <1>       ;mov   word [DirBuff_EntryCounter], 0 ; Zero Based
29578                                  <1>
29579 00009E6F 8935[CF5B0100]           <1>       mov    [CDLF_FNAddress], esi
29580 00009E75 66A3[CD5B0100]           <1>       mov    [CDLF_AttributesMask], ax
29581 00009E7B 66890D[D35B0100]         <1>       mov    [CDLF_DEType], cx
29582                                  <1>
29583 00009E82 31DB                     <1>       xor    ebx, ebx
29584 00009E84 881D[E45B0100]           <1>       mov    [PreviousAttr], bl ; 0  ; 13/02/2016
29585                                  <1>
29586 00009E8A 8A3D[E6520100]           <1>       mov    bh, [Current_Drv]
29587 00009E90 381D[105B0100]           <1>       cmp    byte [DirBuff_ValidData], bl ; 0
29588 00009E96 761D                     <1>       jna    short loc_lcdf_reload_current_dir2
29589 00009E98 8A1D[0E5B0100]           <1>         mov     bl, [DirBuff_DRV]
29590 00009E9E 80EB41                   <1>       sub    bl, 'A'
29591 00009EA1 38DF                     <1>       cmp    bh, bl
29592 00009EA3 750E                     <1>       jne    short loc_lcdf_reload_current_dir1
29593 00009EA5 8B15[155B0100]           <1>       mov    edx, [DirBuff_Cluster]
29594 00009EAB 3B15[E0520100]           <1>       cmp    edx, [Current_Dir_FCluster]
29595 00009EB1 7412                     <1>       je     short loc_cdir_locatefile_search
29596                                  <1>
29597                                  <1> loc_lcdf_reload_current_dir1:
29598 00009EB3 30DB                     <1>       xor    bl, bl
29599                                  <1> loc_lcdf_reload_current_dir2:
29600 00009EB5 89DE                     <1>       mov    esi, ebx
29601 00009EB7 81C600010900             <1>         add     esi, Logical_DOSDisks
29602 00009EBD E874000000               <1>       call   reload_current_directory
29603 00009EC2 735D                     <1>       jnc    short loc_locatefile_search_again
29604 00009EC4 C3                       <1>       retn
29605                                  <1>
29606                                  <1> loc_cdir_locatefile_search:
29607 00009EC5 31DB                     <1>       xor    ebx, ebx
29608 00009EC7 55                       <1>       push   ebp ; 20/11/2017
29609 00009EC8 E8A6000000               <1>       call   find_directory_entry
29610 00009ECD 5D                       <1>       pop    ebp ; 20/11/2017
29611 00009ECE 7349                     <1>       jnc    short loc_cdir_locate_file_retn
29612                                  <1>
29613                                  <1> loc_locatefile_check_stc_reason:
29614 00009ED0 08ED                     <1>       or     ch, ch
29615 00009ED2 7444                     <1>       jz     short loc_cdir_locate_file_stc_retn
29616                                  <1>
29617                                  <1> loc_locatefile_check_next_entryblock:
29618 00009ED4 8A3D[E6520100]           <1>       mov    bh, [Current_Drv]
29619 00009EDA 28DB                     <1>       sub    bl, bl
29620 00009EDC 0FB7F3                   <1>       movzx  esi, bx
29621 00009EDF 81C600010900             <1>         add     esi, Logical_DOSDisks
29622                                  <1>
29623 00009EE5 803D[E4520100]00         <1>       cmp    byte [Current_Dir_Level], 0
29624 00009EEC 760A                     <1>       jna    short loc_locatefile_check_FAT_type
29625                                  <1>
29626 00009EEE 803D[E5520100]01         <1>       cmp    byte [Current_FATType], 1
29627 00009EF5 730A                     <1>       jnb    short loc_locatefile_load_subdir_cluster
29628 00009EF7 C3                       <1>       retn
29629                                  <1>
29630                                  <1> loc_locatefile_check_FAT_type:
29631 00009EF8 803D[E5520100]03         <1>       cmp    byte [Current_FATType], 3
29632 00009EFF 7218                     <1>       jb     short loc_cdir_locate_file_retn
29633                                  <1>
29634                                  <1> loc_locatefile_load_subdir_cluster:
29635 00009F01 A1[155B0100]             <1>       mov    eax, [DirBuff_Cluster]
29636 00009F06 E83C1A0000               <1>       call   get_next_cluster
29637 00009F0B 730D                     <1>       jnc    short loc_locatefile_next_cluster
29638 00009F0D 09C0                     <1>       or     eax, eax
29639 00009F0F 7507                     <1>       jnz    short loc_locatefile_drive_not_ready_read_err
29640 00009F11 F9                       <1>       stc
29641                                  <1> loc_locatefile_file_notfound:
29642 00009F12 B802000000               <1>       mov    eax, 2 ; File/Directory/VolName not found
29643 00009F17 C3                       <1>       retn
29644                                  <1>
29645                                  <1> loc_locatefile_drive_not_ready_read_err:
29646                                  <1>       ;mov   eax, 17 ;Drive not ready or read error
29647                                  <1> loc_cdir_locate_file_stc_retn:
29648 00009F18 F5                       <1>       cmc ;stc
29649                                  <1> loc_cdir_locate_file_retn:
29650 00009F19 C3                       <1>       retn
29651                                  <1>
29652                                  <1> loc_locatefile_next_cluster:
29653 00009F1A E80E1C0000               <1>       call   load_FAT_sub_directory
29654                                  <1>       ;jc    short loc_locatefile_drive_not_ready_read_err
29655 00009F1F 72F8                     <1>       jc     short loc_cdir_locate_file_retn
29656                                  <1>
29657                                  <1> loc_locatefile_search_again:
29658 00009F21 8B35[CF5B0100]           <1>       mov    esi, [CDLF_FNAddress]
29659 00009F27 66A1[CD5B0100]           <1>       mov    ax, [CDLF_AttributesMask]
```

```
29660 00009F2D 668B0D[D35B0100]    <1>       mov    cx, [CDLF_DEType]
29661 00009F34 EB8F                <1>       jmp    short loc_cdir_locatefile_search
29662                              <1>
29663                              <1> reload_current_directory:
29664                              <1>       ; 06/02/2016 (TRDOS 386 = TRDOS v2.0)
29665                              <1>       ; 13/06/2010
29666                              <1>       ; 22/09/2009
29667                              <1>         ;
29668                              <1>       ; INPUT ->
29669                              <1>       ;      ESI = Dos drive description table address
29670                              <1>
29671                              <1>       ;mov   al, [esi+LD_FATType]
29672 00009F36 A0[E5520100]        <1>       mov    al, [Current_FATType]
29673 00009F3B 3C02                <1>       cmp    al, 2
29674 00009F3D 7729                <1>       ja     short loc_reload_FAT_sub_directory
29675 00009F3F 8A25[E4520100]      <1>       mov    ah, [Current_Dir_Level]
29676 00009F45 08C0                <1>       or     al, al
29677 00009F47 740A                <1>       jz     short loc_reload_FS_directory
29678 00009F49 08E4                <1>       or     ah, ah
29679 00009F4B 751B                <1>       jnz    short loc_reload_FAT_sub_directory
29680                              <1> loc_reload_FAT_12_16_root_directory:
29681 00009F4D E8501B0000          <1>       call   load_FAT_root_directory
29682 00009F52 C3                  <1>       retn
29683                              <1> loc_reload_FS_directory:
29684 00009F53 20E4                <1>       and    ah, ah
29685 00009F55 7506                <1>       jnz    short loc_reload_FS_sub_directory
29686                              <1> loc_reload_FS_root_directory:
29687 00009F57 E80D1C0000          <1>       call   load_FS_root_directory
29688 00009F5C C3                  <1>       retn
29689                              <1> loc_reload_FS_sub_directory:
29690 00009F5D A1[E0520100]        <1>       mov    eax, [Current_Dir_FCluster]
29691 00009F62 E8031C0000          <1>       call   load_FS_sub_directory
29692 00009F67 C3                  <1>       retn
29693                              <1> loc_reload_FAT_sub_directory:
29694 00009F68 A1[E0520100]        <1>       mov    eax, [Current_Dir_FCluster]
29695 00009F6D E8BB1B0000          <1>       call   load_FAT_sub_directory
29696 00009F72 C3                  <1>       retn
29697                              <1>
29698                              <1> find_directory_entry:
29699                              <1>       ; 14/02/2016
29700                              <1>       ; 13/02/2016
29701                              <1>       ; 10/02/2016
29702                              <1>       ; 06/02/2016 (TRDOS 386 = TRDOS v2.0)
29703                              <1>       ; 14/08/2010 (DIR.ASM, "proc_find_direntry")
29704                              <1>       ; 19/09/2009
29705                              <1>       ; 2005
29706                              <1>       ; INPUT ->
29707                              <1>       ;      ESI = Sub Dir or File Name Address
29708                              <1>       ;      AL = Attributes Mask
29709                              <1>       ;      (<AL AND EntryAttrib> must be equal to AL)
29710                              <1>       ;      AH = Negative Attributes Mask (If AH>0)
29711                              <1>       ;      (<AH AND EntryAttrib> must be ZERO)
29712                              <1>       ;      CH > 0 Find First Free Dir Entry or Deleted Entry
29713                              <1>       ;      CL = 0 -> Return the First Free Dir Entry
29714                              <1>       ;      CL = E5h -> Return the 1st deleted entry
29715                              <1>       ;      CL = FFh -> Return the 1st deleted or free entry
29716                              <1>       ;      CL > 0 and CL <> E5h and CL <> FFh -> Return the first
29717                              <1>       ;             proper entry (which fits with Atributes Masks)
29718                              <1>       ;      CX = 0 -> Find Valid File/Directory/VolumeName
29719                              <1>       ;      ? = Any One Char
29720                              <1>       ;      * = Every Chars
29721                              <1>       ;      EBX = Current Dir Entry (BX)
29722                              <1>       ;
29723                              <1>       ; OUTPUT ->
29724                              <1>       ;      EDI = Directory Entry Address (in DirectoryBuffer)
29725                              <1>       ;      ESI = Sub Dir or File Name Address
29726                              <1>       ;      CF = 0 -> No Error, Proper Entry,
29727                              <1>       ;      DL = Attributes
29728                              <1>       ;      DH = Previous Entry Attr (LongName Check)
29729                              <1>       ;      AL > 0 -> Ambiguous filename wildcard "?" used
29730                              <1>       ;      AH > 0 -> Ambiguous filename wildcard "*" used
29731                              <1>       ;      AX = 0 -> Filename full fits with directory entry
29732                              <1>       ;      EBX = CurrentDirEntry (BX)
29733                              <1>       ;      CH = The 1st Name Char of Current Dir Entry
29734                              <1>       ;      CF = 1 -> Proper entry not found, Error Code in AX/AL
29735                              <1>       ;      CL = 0 and CH = 0 -> Free Entry (End Of Dir)
29736                              <1>       ;      CL = 0 and CH = E5h -> Deleted Entry fits with filters
29737                              <1>       ;      CL > 0 -> Entry not found, CH invalid
29738                              <1>       ;
29739                              <1>       ; (EAX, EBX, ECX, EDX, EDI, EBP will be changed)
29740                              <1>
29741 00009F73 663B1D[135B0100]    <1>       cmp    bx, [DirBuff_LastEntry]
29742 00009F7A 0F8739010000        <1>         ja      loc_ffde_stc_retn_255
29743                              <1>
29744                              <1>       ;mov    [DirBuff_CurrentEntry], bx
29745                              <1>
29746 00009F80 BF00000800          <1>       mov    edi, Directory_Buffer
29747 00009F85 66A3[E05B0100]      <1>       mov    [FDE_AttrMask], ax
29748                              <1>
29749 00009F8B 29C0                <1>       sub    eax, eax
29750                              <1>
29751                              <1>       ;;mov  [PreviousAttr], al ; 0 ;; 13/02/2016
29752 00009F8D 66A3[E25B0100]      <1>       mov    [AmbiguousFileName], ax ; 0
29753                              <1>
29754 00009F93 6689D8              <1>       mov    ax, bx
29755 00009F96 66C1E005            <1>       shl    ax, 5 ; ; * 32 ; Directory entry size
29756 00009F9A 01C7                <1>       add     edi, eax
29757                              <1>
29758 00009F9C 08ED                <1>       or     ch, ch
29759 00009F9E 0F852C010000        <1>         jnz     loc_find_free_deleted_entry_0
29760                              <1>
29761 00009FA4 08C9                <1>       or      cl, cl
29762 00009FA6 0F850D010000        <1>         jnz     loc_ffde_stc_retn_255
```

```
                                <1>
                                <1> check_find_dir_entry:
00009FAC 66A1[E05B0100]         <1>       mov    ax, [FDE_AttrMask]
00009FB2 8A2F                   <1>       mov    ch, [edi]
00009FB4 80FD00                 <1>       cmp     ch, 0 ; Is it never used entry?
00009FB7 0F86FF000000           <1>       jna    loc_find_direntry_stc_retn
00009FBD 56                     <1>       push   esi
00009FBE 8A570B                 <1>       mov    dl, [edi+0Bh] ; File attributes
00009FC1 80FDE5                 <1>       cmp    ch, 0E5h ; Is it a deleted file?
00009FC4 746D                   <1>       je     short loc_find_dir_next_entry_prevdeleted
                                <1>
00009FC6 80FA0F                 <1>       cmp     dl, 0Fh ; longname sub component check
00009FC9 7505                   <1>       jne     short loc_check_attributes_mask
00009FCB E8ED010000             <1>       call   save_longname_sub_component
                                <1>
                                <1> loc_check_attributes_mask:
00009FD0 88C6                   <1>       mov    dh, al
00009FD2 20D6                   <1>       and    dh, dl
00009FD4 38F0                   <1>       cmp    al, dh
00009FD6 0F85BA000000           <1>         jne     loc_find_dir_next_entry
00009FDC 20D4                   <1>       and    ah, dl
00009FDE 0F85B2000000           <1>         jnz     loc_find_dir_next_entry
00009FE4 80FA0F                 <1>       cmp    dl, 0Fh
00009FE7 751A                   <1>       jne    short pass_direntry_attr_check
                                <1>
00009FE9 3C0F                   <1>       cmp    al, 0Fh ; AL = 0Fh -> find long name
00009FEB 0F85A5000000           <1>         jne     loc_find_dir_next_entry
                                <1>
00009FF1 5E                     <1>       pop    esi
00009FF2 6631C0                 <1>       xor    ax, ax
00009FF5 8A35[E45B0100]         <1>       mov    dh, [PreviousAttr]
00009FFB 66891D[115B0100]       <1>       mov    [DirBuff_CurrentEntry], bx
0000A002 C3                     <1>       retn
                                <1>
                                <1> pass_direntry_attr_check:
0000A003 89FD                   <1>       mov    ebp, edi ; 14/02/2016
0000A005 B908000000             <1>       mov    ecx, 8
                                <1> loc_lodsb_find_dir:
0000A00A AC                     <1>       lodsb
0000A00B 3C2A                   <1>       cmp    al, '*'
0000A00D 7508                   <1>       jne    short pass_fde_ambiguous1_check
0000A00F FE05[E35B0100]         <1>         inc     byte [AmbiguousFileName+1]
0000A015 EB28                   <1>       jmp    short loc_check_direntry_extension
                                <1>
                                <1> pass_fde_ambiguous1_check:
0000A017 3C3F                   <1>       cmp    al, '?'
0000A019 750D                   <1>       jne    short pass_fde_ambiguous2_check
0000A01B FE05[E25B0100]         <1>       inc    byte [AmbiguousFileName]
0000A021 803F20                 <1>       cmp    byte [edi], 20h
0000A024 764E                   <1>       jna    short loc_find_dir_next_entry_ebp
0000A026 EB14                   <1>       jmp    short loc_scasb_find_dir_inc_di
                                <1>
                                <1> pass_fde_ambiguous2_check:
0000A028 3C20                   <1>       cmp    al, 20h
0000A02A 750C                   <1>       jne    short loc_scasb_find_dir
0000A02C 803F20                 <1>       cmp    byte [edi], 20h
0000A02F 7543                   <1>       jne    short loc_find_dir_next_entry_ebp
0000A031 EB0C                   <1>       jmp    short loc_check_direntry_extension
                                <1>
                                <1> loc_find_dir_next_entry_prevdeleted:
0000A033 80CA80                 <1>       or     dl, 80h  ; Bit 7 -> deleted entry sign
0000A036 EB5E                   <1>       jmp    short loc_find_dir_next_entry
                                <1>
                                <1> loc_scasb_find_dir:
0000A038 3A07                   <1>       cmp    al, [edi]
0000A03A 7538                   <1>       jne    short loc_find_dir_next_entry_ebp
                                <1> loc_scasb_find_dir_inc_di:
0000A03C 47                     <1>       inc    edi
0000A03D E2CB                   <1>       loop   loc_lodsb_find_dir
                                <1>
                                <1> loc_check_direntry_extension:
0000A03F BE08000000             <1>       mov    esi, 8
0000A044 89F7                   <1>       mov    edi, esi ; 8
0000A046 033424                 <1>       add    esi, [esp] ; Sub Dir or File Name Address
0000A049 01EF                   <1>       add    edi, ebp
0000A04B B103                   <1>       mov    cl, 3
                                <1> loc_lodsb_find_dir_ext:
0000A04D AC                     <1>       lodsb
0000A04E 3C2A                   <1>       cmp    al, '*'
0000A050 7508                   <1>       jne    short pass_fde_ambiguous3_check
0000A052 FE05[E35B0100]         <1>       inc    byte [AmbiguousFileName+1]
0000A058 EB1E                   <1>       jmp    short loc_find_dir_proper_direntry
                                <1>
                                <1> pass_fde_ambiguous3_check:
0000A05A 3C3F                   <1>       cmp    al, '?'
0000A05C 750D                   <1>       jne    short pass_fde_ambiguous4_check
0000A05E FE05[E25B0100]         <1>       inc    byte [AmbiguousFileName]
0000A064 803F20                 <1>       cmp    byte [edi], 20h
0000A067 760B                   <1>       jna    short loc_find_dir_next_entry_ebp
0000A069 EB49                   <1>       jmp    short loc_scasb_find_dir_ext_inc_di
                                <1>
                                <1> pass_fde_ambiguous4_check:
0000A06B 3C20                   <1>       cmp    al, 20h
0000A06D 7541                   <1>       jne    short loc_scasb_find_dir_ext
0000A06F 803F20                 <1>       cmp    byte [edi], 20h
0000A072 7404                   <1>       je     short loc_find_dir_proper_direntry
                                <1>
                                <1> loc_find_dir_next_entry_ebp:
0000A074 89EF                   <1>       mov    edi, ebp ; 14/02/2016
0000A076 EB1E                   <1>       jmp    short loc_find_dir_next_entry
                                <1>
                                <1> loc_find_dir_proper_direntry:
0000A078 30C9                   <1>       xor    cl, cl
```

```
29866                                <1> loc_find_dir_proper_direntry_1:
29867 0000A07A 5E                    <1>       pop    esi
29868 0000A07B 89EF                  <1>         mov     edi, ebp
29869 0000A07D 8A2F                  <1>       mov    ch, [edi]
29870 0000A07F 8A570B                <1>       mov     dl, [edi+0Bh] ; Dir entry attributes
29871 0000A082 66A1[E25B0100]        <1>       mov    ax, [AmbiguousFileName]
29872                                <1> loc_find_dir_proper_direntry_2:
29873 0000A088 8A35[E45B0100]        <1>       mov     dh, [PreviousAttr]
29874 0000A08E 66891D[115B0100]      <1>       mov    [DirBuff_CurrentEntry], bx
29875 0000A095 C3                    <1>       retn
29876                                <1>
29877                                <1> loc_find_dir_next_entry:
29878 0000A096 8815[E45B0100]        <1>       mov    byte [PreviousAttr], dl ; LongName check
29879                                <1> loc_find_dir_next_entry_1:
29880 0000A09C 5E                    <1>       pop    esi
29881 0000A09D 83C720                <1>       add    edi, 32
29882                                <1>       ;inc   word [DirBuff_EntryCounter]
29883 0000A0A0 6643                  <1>       inc    bx
29884 0000A0A2 663B1D[135B0100]      <1>       cmp    bx, [DirBuff_LastEntry]
29885 0000A0A9 770E                  <1>       ja     short loc_ffde_stc_retn_255
29886 0000A0AB E9FCFEFFFF            <1>         jmp     check_find_dir_entry
29887                                <1>
29888                                <1> loc_scasb_find_dir_ext:
29889 0000A0B0 3A07                  <1>       cmp    al, [edi]
29890 0000A0B2 75C0                  <1>       jne    short loc_find_dir_next_entry_ebp
29891                                <1> loc_scasb_find_dir_ext_inc_di:
29892 0000A0B4 47                    <1>       inc    edi
29893 0000A0B5 E296                  <1>       loop    loc_lodsb_find_dir_ext
29894 0000A0B7 EBC1                  <1>       jmp    short loc_find_dir_proper_direntry_1
29895                                <1>
29896                                <1> loc_ffde_stc_retn_255:
29897                                <1>       ;mov   cx, 0FFFFh
29898 0000A0B9 31C9                  <1>       xor    ecx, ecx
29899 0000A0BB 49                    <1>       dec    ecx ; 0FFFFFFFFh
29900                                <1>       ;xor   eax, eax
29901                                <1> loc_find_direntry_stc_retn:
29902                                <1> loc_check_ffde_retn_1:
29903                                <1>       ;mov   ax, 2
29904 0000A0BC B802000000            <1>       mov    eax, 2 ; File Not Found
29905 0000A0C1 8A35[E45B0100]        <1>       mov    dh, [PreviousAttr]
29906 0000A0C7 66891D[115B0100]      <1>       mov    [DirBuff_CurrentEntry], bx
29907 0000A0CE F9                    <1>       stc
29908 0000A0CF C3                    <1>       retn
29909                                <1>
29910                                <1> loc_find_free_deleted_entry_0:
29911 0000A0D0 66A1[E05B0100]        <1>       mov    ax, [FDE_AttrMask]
29912 0000A0D6 8A2F                  <1>       mov    ch, [edi]
29913 0000A0D8 8A570B                <1>       mov    dl, [edi+0Bh] ; File attributes
29914 0000A0DB 08C9                  <1>       or     cl, cl
29915 0000A0DD 7407                  <1>       jz     short loc_check_ffde_0_repeat
29916                                <1>       ;cmp   cl, 0E5h
29917                                <1>       ;je    short pass_loc_check_ffde_0_err
29918 0000A0DF 80F9FF                <1>       cmp    cl, 0FFh
29919 0000A0E2 7432                  <1>       je     short loc_find_free_deleted_entry_1
29920 0000A0E4 EB4D                  <1>       jmp    short pass_loc_check_ffde_0_err
29921                                <1>
29922                                <1> loc_check_ffde_0_repeat:
29923 0000A0E6 08ED                  <1>       or     ch, ch
29924 0000A0E8 7511                  <1>       jnz    short loc_check_ffde_0_next
29925                                <1>
29926                                <1> loc_check_ffde_retn_2:
29927 0000A0EA 6629C0                <1>       sub    ax, ax
29928 0000A0ED 8A35[E45B0100]        <1>       mov    dh, [PreviousAttr]
29929 0000A0F3 66891D[115B0100]      <1>       mov    [DirBuff_CurrentEntry], bx
29930 0000A0FA C3                    <1>       retn
29931                                <1>
29932                                <1> loc_check_ffde_0_next:
29933 0000A0FB 6643                  <1>       inc    bx
29934 0000A0FD 83C720                <1>       add    edi, 32
29935                                <1>       ;inc   word [DirBuff_EntryCounter]
29936                                <1>
29937 0000A100 663B1D[135B0100]      <1>         cmp  bx, [DirBuff_LastEntry]
29938 0000A107 77B0                  <1>       ja     short loc_ffde_stc_retn_255
29939 0000A109 8815[E45B0100]        <1>       mov    [PreviousAttr], dl
29940 0000A10F 8A2F                  <1>       mov    ch, [edi]
29941 0000A111 8A570B                <1>       mov    dl, [edi+0Bh] ; file attributes
29942 0000A114 EBD0                  <1>       jmp    short loc_check_ffde_0_repeat
29943                                <1>
29944                                <1> loc_find_free_deleted_entry_1:
29945 0000A116 28D2                  <1>       sub    dl, dl
29946                                <1> loc_find_free_deleted_entry_2:
29947 0000A118 20ED                  <1>       and    ch, ch
29948 0000A11A 74CE                  <1>       jz     short loc_check_ffde_retn_2
29949 0000A11C 80FDE5                <1>       cmp    ch, 0E5h
29950 0000A11F 74C9                  <1>       je     short loc_check_ffde_retn_2
29951 0000A121 6643                  <1>       inc    bx
29952 0000A123 83C720                <1>       add    edi, 32
29953 0000A126 663B1D[135B0100]      <1>       cmp    bx, [DirBuff_LastEntry]
29954 0000A12D 778A                  <1>       ja     short loc_ffde_stc_retn_255
29955 0000A12F 8A2F                  <1>       mov    ch, [edi]
29956 0000A131 EBE5                  <1>       jmp    short loc_find_free_deleted_entry_2
29957                                <1>
29958                                <1> pass_loc_check_ffde_0_err:
29959 0000A133 38CD                  <1>       cmp    ch, cl
29960 0000A135 741F                  <1>       je     short loc_check_ffde_attrib
29961                                <1>
29962 0000A137 6643                  <1>       inc    bx
29963 0000A139 83C720                <1>       add    edi, 32
29964 0000A13C 663B1D[135B0100]      <1>       cmp    bx, [DirBuff_LastEntry]
29965 0000A143 0F8770FFFFFF          <1>         ja      loc_ffde_stc_retn_255
29966 0000A149 8815[E45B0100]        <1>       mov    [PreviousAttr], dl
29967 0000A14F 8A2F                  <1>       mov    ch, [edi]
29968 0000A151 8A570B                <1>       mov    dl, [edi+0Bh]
```

```
29969 0000A154 EBDD              <1>       jmp    short pass_loc_check_ffde_0_err
29970                            <1>
29971                            <1> loc_check_ffde_attrib:
29972 0000A156 88C6              <1>       mov    dh, al
29973 0000A158 20D6              <1>       and    dh, dl
29974 0000A15A 38F0              <1>       cmp    al, dh
29975 0000A15C 759D              <1>       jne    short loc_check_ffde_0_next
29976 0000A15E 20D4              <1>       and    ah, dl
29977 0000A160 7599              <1>       jnz    short loc_check_ffde_0_next
29978 0000A162 30C9              <1>       xor    cl, cl
29979 0000A164 EB84              <1>         jmp     loc_check_ffde_retn_2
29980                            <1>
29981                            <1> convert_file_name:
29982                            <1>       ; 06/03/2016
29983                            <1>       ; 11/02/2016
29984                            <1>       ; 07/02/2016 (TRDOS 386 = TRDOS v2.0)
29985                            <1>       ; 06/10/2009
29986                            <1>       ; 2005
29987                            <1>       ;
29988                            <1>       ; INPUT  ->
29989                            <1>       ;      ESI = Dot File Name Location
29990                            <1>       ;      EDI = Dir Entry Format File Name Location
29991                            <1>       ; OUTPUT ->
29992                            <1>       ;      EDI = Dir Entry Format File Name Location
29993                            <1>       ;      ESI = Dot File Name Location (capitalized)
29994                            <1>       ;
29995                            <1>       ; (ECX, AL will be changed)
29996                            <1>
29997 0000A166 56                <1>       push   esi
29998 0000A167 57                <1>       push   edi
29999                            <1>
30000 0000A168 B90B000000        <1>       mov    ecx, 11
30001 0000A16D B020              <1>       mov    al, 20h
30002 0000A16F F3AA              <1>       rep    stosb
30003                            <1>
30004 0000A171 8B3C24            <1>       mov    edi, [esp]
30005                            <1>
30006 0000A174 B10C              <1>       mov    cl, 12 ; file name length (max.)
30007                            <1>       ; 06/03/2016
30008                            <1>       ; Directory entry name limit (11 bytes) check for
30009                            <1>       ; 'rename_directory_entry' procedure.
30010                            <1>       ; (EDI points to Directory Entry)
30011                            <1>       ; (If the file name would not contain a dot
30012                            <1>       ; and file name length would be 12, this would cause to
30013                            <1>       ; overwrite the attributes byte of the directory entry.)
30014                            <1>       ;
30015 0000A176 B50B              <1>       mov    ch, 11 ; directory entry's name length
30016                            <1> loc_check_first_dot:
30017 0000A178 8A06              <1>       mov    al, [esi]
30018 0000A17A 3C2E              <1>       cmp    al, 2Eh
30019 0000A17C 750C              <1>       jne    short pass_check_first_dot
30020 0000A17E 8807              <1>       mov    [edi], al
30021 0000A180 47                <1>       inc    edi
30022 0000A181 46                <1>       inc    esi
30023 0000A182 FEC9              <1>       dec    cl
30024 0000A184 75F2              <1>       jnz    short loc_check_first_dot
30025                            <1>       ;;(ecx <= 12)
30026                            <1>       ;;loop loc_check_first_dot
30027 0000A186 EB30              <1>       jmp    short stop_convert_file
30028                            <1>
30029                            <1> loc_get_fchar:
30030 0000A188 8A06              <1>       mov    al, [esi]
30031                            <1> pass_check_first_dot:
30032 0000A18A 3C61              <1>       cmp    al, 61h ; 'a'
30033 0000A18C 7208              <1>       jb     short pass_name_capitalize
30034 0000A18E 3C7A              <1>       cmp    al, 7Ah ; 'z'
30035 0000A190 7704              <1>       ja     short pass_name_capitalize
30036 0000A192 24DF              <1>       and    al, 0DFh
30037 0000A194 8806              <1>       mov    [esi], al
30038                            <1> pass_name_capitalize:
30039 0000A196 3C21              <1>       cmp    al, 21h
30040 0000A198 721E              <1>       jb     short stop_convert_file
30041 0000A19A 3C2E              <1>       cmp    al, 2Eh ; '.'
30042 0000A19C 750C              <1>       jne    short pass_dot_space
30043                            <1> add_dot_space:
30044 0000A19E 80F904            <1>       cmp    cl, 4
30045 0000A1A1 760E              <1>       jna    short inc_and_loop
30046 0000A1A3 47                <1>       inc    edi
30047 0000A1A4 FECD              <1>       dec    ch ; 06/03/2016
30048 0000A1A6 FEC9              <1>       dec    cl
30049 0000A1A8 EBF4              <1>       jmp    short add_dot_space
30050                            <1>
30051                            <1>       ;mov   al, 4
30052                            <1>       ;cmp   cl, al
30053                            <1>       ;jna   short inc_and_loop
30054                            <1>       ;sub   cl, al
30055                            <1>       ;add   edi, ecx
30056                            <1>       ;mov   cl, al
30057                            <1>       ;jmp   short inc_and_loop
30058                            <1>
30059                            <1> pass_dot_space:
30060 0000A1AA 8807              <1>       mov    [edi], al
30061                            <1> loc_after_double_dot:
30062                            <1>       ; 06/03/2016
30063 0000A1AC FECD              <1>       dec    ch ; count down for 11 bytes dir entry limit
30064 0000A1AE 740A              <1>       jz     short stop_convert_file_x
30065 0000A1B0 47                <1>       inc    edi
30066                            <1> inc_and_loop:
30067 0000A1B1 FEC9              <1>       dec    cl ; count down for 12 bytes filename limit
30068 0000A1B3 7403              <1>       jz     short stop_convert_file
30069 0000A1B5 46                <1>       inc    esi
30070                            <1>       ;;(ecx <= 12)
30071                            <1>       ;;loop loc_get_fchar
```

```
30072 0000A1B6 EBD0               <1>       jmp    short loc_get_fchar
30073                             <1>
30074                             <1> stop_convert_file:
30075                             <1>       ; 06/03/2016
30076 0000A1B8 30ED               <1>       xor    ch, ch
30077                             <1>       ; ECX < 256 ; 'find_first_file' -> xor cl, cl
30078                             <1> stop_convert_file_x:
30079 0000A1BA 5F                 <1>       pop    edi
30080 0000A1BB 5E                 <1>       pop    esi
30081 0000A1BC C3                 <1>       retn
30082                             <1>
30083                             <1> save_longname_sub_component:
30084                             <1>       ; 13/02/2016
30085                             <1>       ; 06/02/2016 (TRDOS 386 = TRDOS v2.0)
30086                             <1>       ; 28/02/2010
30087                             <1>       ; 17/10/2009
30088                             <1>       ; INPUT ->
30089                             <1>       ;      EDI = Directory Entry
30090                             <1>       ;      // This procedure is called
30091                             <1>       ;      // from 'find_directory_entry' procedure.
30092                             <1>       ;      // If the last entry returns with
30093                             <1>       ;      // a non-zero LongnameFound value and
30094                             <1>       ;      // if LFN_CheckSum value is equal to
30095                             <1>       ;      // the next shortname checksum,
30096                             <1>       ;      // long name is valid.
30097                             <1>       ;      // If a longname is longer than 65 bytes,
30098                             <1>       ;      // it is invalid for trdos. (>45h)
30099                             <1>
30100 0000A1BD 57                 <1>       push   edi
30101 0000A1BE 56                 <1>       push   esi
30102                             <1>       ;push  ebx
30103                             <1>       ;push  ecx
30104                             <1>       ;push  edx
30105 0000A1BF 50                 <1>       push   eax
30106                             <1>
30107 0000A1C0 29C9               <1>       sub    ecx, ecx
30108                             <1>       ;sub   eax, eax
30109 0000A1C2 B11A               <1>       mov    cl, 26
30110                             <1>
30111 0000A1C4 0FB607             <1>       movzx  eax, byte [edi] ; LDIR_Order
30112 0000A1C7 3C41               <1>       cmp    al, 41h  ; 40h (last long entry sign) + 1
30113 0000A1C9 722B               <1>       jb     short pass_pslnsc_last_long_entry
30114                             <1>
30115 0000A1CB 88C4               <1>       mov    ah, al
30116 0000A1CD 80EC40             <1>       sub    ah, 40h
30117 0000A1D0 8825[E65B0100]     <1>       mov    [LFN_EntryLength], ah
30118                             <1>
30119 0000A1D6 3C45               <1>       cmp    al, 45h  ; 40h (last long entry sign) + 5
30120                             <1>              ; Max 130 byte length is usable in TRDOS
30121                             <1> ; 26*5 = 130
30122 0000A1D8 7753               <1>       ja     short loc_pslnsc_retn
30123                             <1>
30124 0000A1DA 2407               <1>       and    al, 07h ; 0Fh
30125 0000A1DC A2[E55B0100]       <1>       mov    [LongNameFound], al
30126                             <1>
30127 0000A1E1 FEC8               <1>       dec    al
30128                             <1>       ;mov   cl, 26
30129 0000A1E3 F6E1               <1>       mul    cl
30130                             <1>
30131 0000A1E5 89C6               <1>       mov    esi, eax
30132 0000A1E7 01CE               <1>       add    esi, ecx
30133                             <1>              ; to make is an ASCIZZ string
30134                             <1>              ; with ax+26 bytes length
30135 0000A1E9 81C6[E85B0100]     <1>       add    esi, LongFileName
30136 0000A1EF 66C7060000         <1>       mov    word [esi], 0
30137 0000A1F4 EB16               <1>       jmp    short loc_pslsc_move_ldir_name2
30138                             <1>
30139                             <1> pass_pslnsc_last_long_entry:
30140 0000A1F6 3C04               <1>       cmp    al, 04h
30141 0000A1F8 7733               <1>       ja     short loc_pslnsc_retn
30142 0000A1FA FE0D[E55B0100]     <1>       dec    byte [LongNameFound]
30143 0000A200 3A05[E55B0100]     <1>       cmp    al, [LongNameFound]
30144 0000A206 7525               <1>       jne    short loc_pslnsc_retn
30145                             <1>
30146                             <1> loc_pslsc_move_ldir_name1:
30147 0000A208 FEC8               <1>       dec    al
30148                             <1>       ;mov   cl, 26
30149 0000A20A F6E1               <1>       mul    cl
30150                             <1>
30151                             <1> loc_pslsc_move_ldir_name2:
30152 0000A20C 8A4F0D             <1>       mov    cl, [edi+0Dh] ; long name checksum
30153 0000A20F 880D[E75B0100]     <1>       mov    [LFN_CheckSum], cl
30154 0000A215 89FE               <1>       mov    esi, edi ; LDIR_Order
30155 0000A217 BF[E85B0100]       <1>       mov    edi, LongFileName
30156 0000A21C 01C7               <1>       add    edi, eax
30157 0000A21E 46                 <1>       inc    esi
30158 0000A21F B105               <1>       mov    cl, 5 ; chars 1 to 5
30159 0000A221 F366A5             <1>       rep    movsw
30160 0000A224 83C603             <1>       add    esi, 3
30161 0000A227 A5                 <1>       movsd  ; char 6 & 7
30162 0000A228 A5                 <1>       movsd  ; char 8 & 9
30163 0000A229 A5                 <1>       movsd  ; char 10 & 11
30164 0000A22A 46                 <1>       inc    esi
30165 0000A22B 46                 <1>       inc    esi
30166 0000A22C A5                 <1>       movsd   ; char 12 & 13
30167                             <1>
30168                             <1> loc_pslnsc_retn:
30169 0000A22D 58                 <1>       pop    eax
30170                             <1>       ;pop   edx
30171                             <1>       ;pop   ecx
30172                             <1>       ;pop   ebx
30173 0000A22E 5E                 <1>       pop    esi
30174 0000A22F 5F                 <1>       pop    edi
```

```
30175                                  <1>
30176 0000A230 C3                      <1>       retn
30177                                  <1>
30178                                  <1> parse_path_name:
30179                                  <1>       ; 10/02/2016
30180                                  <1>       ; 08/02/2016 (TRDOS 386 = TRDOS v2.0)
30181                                  <1>       ; 10/009/2011 ('proc_parse_pathname')
30182                                  <1>       ; 27/11/2009
30183                                  <1>       ; 05/12/2004
30184                                  <1>       ;
30185                                  <1>       ; INPUT ->
30186                                  <1>       ;     ESI = Beginning of ASCIIZ pathname string
30187                                  <1>       ;       EDI = Destination Address
30188                                  <1>       ;            (which is TR-DOS FindFile data buffer)
30189                                  <1>       ; OUTPUT ->
30190                                  <1>       ;     CF = 1 -> Error
30191                                  <1>       ;          EAX = Error Code (AL)
30192                                  <1>       ;
30193                                  <1>       ; (Modified registers: eax, ecx, esi, edi)
30194                                  <1>
30195                                  <1>       ; Clear the pathname bytes in TR-DOS Findfile data buffer
30196 0000A231 57                      <1>       push   edi
30197 0000A232 B914000000              <1>       mov    ecx, 20  ; 80 bytes
30198 0000A237 31C0                    <1>       xor    eax, eax
30199 0000A239 F3AB                    <1>       rep    stosd
30200 0000A23B 5F                      <1>       pop    edi
30201                                  <1>
30202 0000A23C 668B06                  <1>       mov    ax, [esi]
30203 0000A23F 80FC3A                  <1>       cmp    ah, ':'
30204 0000A242 741C                    <1>       je     short loc_ppn_change_drive
30205 0000A244 A0[E6520100]            <1>       mov    al, [Current_Drv]
30206 0000A249 EB33                    <1>       jmp    short pass_ppn_change_drive
30207                                  <1>
30208                                  <1> pass_ppn_cdir:
30209 0000A24B 8B35[0A5D0100]          <1>       mov    esi, [First_Path_Pos]
30210 0000A251 AC                      <1>       lodsb
30211                                  <1> loc_ppn_get_filename:
30212 0000A252 83C741                  <1>       add    edi, 65 ; FindFile_Name location
30213                                  <1>       ; TRDOS Filename length must not be more than 12 bytes
30214                                  <1>       ;mov   ecx, 12
30215 0000A255 B10C                    <1>       mov    cl, 12
30216                                  <1> loc_ppn_get_fnchar_next:
30217 0000A257 AA                      <1>       stosb
30218 0000A258 AC                      <1>       lodsb
30219 0000A259 3C21                    <1>       cmp    al, 21h
30220 0000A25B 7274                    <1>       jb     short loc_ppn_clc_return
30221 0000A25D E2F8                    <1>         loop    loc_ppn_get_fnchar_next
30222                                  <1> loc_ppn_return:
30223 0000A25F C3                      <1>       retn
30224                                  <1>
30225                                  <1> loc_ppn_change_drive:
30226 0000A260 24DF                    <1>       and    al, 0DFh
30227 0000A262 2C41                    <1>       sub    al, 'A'; A:
30228 0000A264 726F                    <1>       jc     short loc_ppn_invalid_drive
30229 0000A266 3805[D2060100]          <1>       cmp    [Last_DOS_DiskNo], al
30230 0000A26C 7267                    <1>       jb     short loc_ppn_invalid_drive
30231                                  <1>
30232 0000A26E 46                      <1>       inc    esi
30233 0000A26F 46                      <1>       inc    esi
30234 0000A270 8A26                    <1>       mov    ah, [esi]
30235 0000A272 80FC21                  <1>       cmp    ah, 21h
30236 0000A275 7307                    <1>       jnb    short pass_ppn_change_drive
30237                                  <1>
30238                                  <1> loc_ppn_cmd_failed:
30239                                  <1>       ; File or directory name is not existing
30240 0000A277 8807                    <1>       mov    [edi], al ; Drv
30241 0000A279 66B80100                <1>       mov    ax, 1 ; eax = 1
30242                                  <1>       ; TR-DOS Error Code 01h = Bad Command Argument
30243                                  <1>       ; MS-DOS Error Code 01h : Invalid Function Number
30244                                  <1>       ;stc
30245                                  <1>       ; (MainProg ErrMsg: "Bad command or file name!")
30246 0000A27D C3                      <1>       retn
30247                                  <1>
30248                                  <1> pass_ppn_change_drive:
30249 0000A27E 8935[0A5D0100]          <1>       mov    [First_Path_Pos], esi
30250 0000A284 C705[0E5D0100]0000-     <1>       mov    dword [Last_Slash_Pos], 0
30251 0000A28C 0000                    <1>
30252 0000A28E AA                      <1>       stosb
30253 0000A28F 8A06                    <1>       mov    al, [esi]
30254                                  <1> loc_scan_ppn_dslash:
30255 0000A291 3C2F                    <1>       cmp    al, '/'
30256 0000A293 7506                    <1>       jne    short loc_scan_next_slash_pos
30257 0000A295 8935[0E5D0100]          <1>       mov    [Last_Slash_Pos], esi
30258                                  <1> loc_scan_next_slash_pos:
30259 0000A29B 46                      <1>       inc    esi
30260 0000A29C 8A06                    <1>       mov    al, [esi]
30261 0000A29E 3C20                    <1>       cmp    al, 20h
30262 0000A2A0 77EF                    <1>       ja     short loc_scan_ppn_dslash
30263 0000A2A2 833D[0E5D0100]00        <1>       cmp    dword [Last_Slash_Pos], 0
30264 0000A2A9 76A0                    <1>       jna    short pass_ppn_cdir
30265                                  <1>
30266 0000A2AB 8B0D[0E5D0100]          <1>       mov    ecx, [Last_Slash_Pos]
30267 0000A2B1 8B35[0A5D0100]          <1>       mov    esi, [First_Path_Pos]
30268 0000A2B7 29F1                    <1>       sub    ecx, esi
30269 0000A2B9 41                      <1>       inc    ecx
30270                                  <1>       ;cmp   ecx, 64
30271 0000A2BA 80F940                  <1>       cmp    cl, 64
30272 0000A2BD 7715                    <1>       ja     short loc_ppn_invalid_drive_stc
30273                                  <1>
30274 0000A2BF 89F8                    <1>       mov    eax, edi ; Dest Dir String Location (65 byte)
30275 0000A2C1 F3A4                    <1>       rep    movsb
30276                                  <1>       ;mov   [edi], cl ; 0, End of Dir String
30277 0000A2C3 8B35[0E5D0100]          <1>       mov    esi, [Last_Slash_Pos]
```

```
30278 0000A2C9 46                <1>       inc    esi
30279 0000A2CA 89C7              <1>       mov    edi, eax
30280 0000A2CC AC                <1>       lodsb
30281 0000A2CD 3C21              <1>       cmp    al, 21h
30282 0000A2CF 7381              <1>       jnb    short loc_ppn_get_filename
30283                            <1> loc_ppn_clc_return:
30284                            <1>       ;clc
30285 0000A2D1 31C0              <1>       xor    eax, eax
30286 0000A2D3 C3                <1>       retn
30287                            <1>
30288                            <1> loc_ppn_invalid_drive_stc:
30289 0000A2D4 F5                <1>       cmc     ; stc
30290                            <1> loc_ppn_invalid_drive:
30291                            <1>       ; cf = 1
30292                            <1>       ; The Drive Letter/Char < "A" or > "Z"
30293 0000A2D5 66B80F00          <1>       mov    ax, 0Fh
30294                            <1>       ; MS-DOS Error Code 0Fh = Disk Drive Invalid
30295                            <1>       ; (MainProg ErrMsg: "Drive not ready or read error!")
30296 0000A2D9 C3                <1>       retn
30297                            <1>
30298                            <1> find_longname:
30299                            <1>       ; 13/02/2016 (TRDOS 386 = TRDOS v2.0)
30300                            <1>       ; 24/01/2010 (DIR.ASM, 'proc_find_longname')
30301                            <1>       ; 17/10/2009
30302                            <1>
30303                            <1>       ; INPUT ->
30304                            <1>       ;     ESI = DOS short file name address
30305                            <1>       ;     for example: "filename.ext"
30306                            <1>       ;
30307                            <1>       ; OUTPUT ->
30308                            <1>       ;     ESI = ASCIIZ longname address (cf = 0)
30309                            <1>       ;     cf = 1 -> error number returns in EAX (AL)
30310                            <1>       ;     AL = 0 & CF=1 -> longname not found
30311                            <1>       ;          the file/directory has no longname
30312                            <1>       ;     cf = 0 -> AL = FAT Type
30313                            <1>
30314                            <1>       ; 17/10/2009
30315                            <1>       ; ASCIIZ string will be returned
30316                            <1>       ; as LongFileName
30317                            <1>       ; clearing/reset is not needed
30318                            <1>       ;mov   ecx, 33
30319                            <1>       ;mov   edi, LongFileName
30320                            <1>       ;sub   ax, ax ; 0
30321                            <1>       ;rep   stosw
30322                            <1>
30323                            <1>       ;mov   byte [LongNameFound], 0
30324                            <1>
30325                            <1>       ; ESI = ASCIIZ file/directory name address
30326                            <1>       ;   AL = Attributes AND mask
30327                            <1>       ;      (Result of AND must be equal to AL)
30328                            <1>       ;   AH = Negative attributes mask
30329                            <1>       ;      (Result of AND must be ZERO)
30330 0000A2DA 66B80008          <1>       mov    ax, 0800h
30331                            <1>              ; it must not be volume name or longname
30332 0000A2DE E828DDFFFF        <1>       call   find_first_file
30333 0000A2E3 7216              <1>       jc     short loc_fln_retn
30334                            <1>
30335                            <1> loc_fln_check_FAT_Type:
30336 0000A2E5 803D[E5520100]01  <1>       cmp    byte [Current_FATType], 1
30337 0000A2EC 7306              <1>       jnb    short loc_fln_check_longname_yes_sign
30338                            <1>
30339 0000A2EE E839000000        <1>       call   get_fs_longname
30340 0000A2F3 C3                <1>       retn
30341                            <1>
30342                            <1> loc_fln_check_longname_yes_sign:
30343 0000A2F4 08FF              <1>       or     bh, bh
30344 0000A2F6 7504              <1>       jnz    short loc_fln_check_longnamefound_number
30345                            <1> loc_fln_longname_not_found_retn:
30346 0000A2F8 31C0              <1>       xor    eax, eax
30347                            <1>       ; cf = 1 & al = 0 -> longname not found
30348 0000A2FA F9                <1>       stc
30349                            <1> loc_fln_retn:
30350 0000A2FB C3                <1>       retn
30351                            <1>
30352                            <1> loc_fln_check_longnamefound_number:
30353                            <1>       ; 'LongNameFound' is set by
30354                            <1>         ; by 'save_longname_sub_component'
30355                            <1>       ; which is called from
30356                            <1>       ; 'find_directory_entry'
30357                            <1>       ; which is called from
30358                            <1>       ; 'find_first_file'
30359                            <1>       ; It must 1 if the longname is valid
30360 0000A2FC 803D[E55B0100]01  <1>         cmp     byte [LongNameFound], 1
30361 0000A303 75F3              <1>       jne    short loc_fln_longname_not_found_retn
30362                            <1>
30363                            <1> loc_fln_calculate_checksum:
30364 0000A305 E813000000        <1>       call   calculate_checksum
30365                            <1>       ; AL = shortname checksum
30366                            <1>
30367                            <1> loc_fln_longname_validation:
30368                            <1>       ; 'LFN_CheckSum' has been set already
30369                            <1>       ; by 'save_longname_sub_component'
30370                            <1>       ; which is called from
30371                            <1>       ; 'find_directory_entry'
30372                            <1>       ; which is called from
30373                            <1>       ; 'find_first_file'
30374 0000A30A 3805[E75B0100]    <1>       cmp    [LFN_CheckSum], al
30375 0000A310 75E6              <1>       jne    short loc_fln_longname_not_found_retn
30376                            <1>
30377 0000A312 BE[E85B0100]      <1>       mov    esi, LongFileName
30378 0000A317 A0[E5520100]      <1>       mov    al, [Current_FATType]
30379 0000A31C C3                <1>       retn
30380                            <1>
```

```
30381                                  <1> calculate_checksum:
30382                                  <1>       ; 13/02/2016 (TRDOS 386 = TRDOS v2.0)
30383                                  <1>       ; 17/10/2009 (DIR.ASM, 'proc_calculate_checksum')
30384                                  <1>         ;
30385                                  <1>       ; INPUT ->
30386                                  <1>       ;     ESI = 11 byte DOS File Name location
30387                                  <1>       ;     (in DOS Directory Entry Format)
30388                                  <1>       ; OUTPUT ->
30389                                  <1>       ;      AL = 8 bit checksum (CRC) value
30390                                  <1>       ;
30391                                  <1>       ; (Modified registers: EAX, ECX, ESI)
30392                                  <1>
30393                                  <1>       ; Erdogan Tan [ 17-10-2009 ]
30394                                  <1>       ;  'ror al, 1' instruction
30395                                  <1>
30396                                  <1>       ; Erdogan Tan [ 20-06-2004 ]
30397                                  <1>       ; This 8086 assembly code is an original code
30398                                  <1>       ; which is adapted from C code in
30399                                  <1>       ; Microsoft FAT32 File System Specification
30400                                  <1>       ; Version 1.03, December 6, 2000
30401                                  <1>       ; Page 28
30402                                  <1>
30403 0000A31D 30C0                   <1>       xor    al, al
30404 0000A31F B90B000000             <1>       mov    ecx, 11
30405                                  <1> loc_next_sum:
30406                                  <1>       ;xor   ah, ah
30407                                  <1>       ;test  al, 1
30408                                  <1>       ;jz    short pass_ah_80h
30409                                  <1>       ;mov   ah, 80h
30410                                  <1> ;pass_ah_80h:
30411                                  <1>       ;shr   al, 1
30412 0000A324 D0C8                   <1>       ror    al, 1 ; 17/10/2009
30413 0000A326 0206                   <1>       add    al, [esi]
30414 0000A328 46                     <1>       inc    esi
30415                                  <1>       ;add   al, ah
30416 0000A329 E2F9                   <1>       loop   loc_next_sum
30417 0000A32B C3                     <1>       retn
30418                                  <1>
30419                                  <1> get_fs_longname:
30420                                  <1>       ; temporary (13/02/2016)
30421 0000A32C 31C0                   <1>       xor eax, eax
30422 0000A32E F9                     <1>       stc
30423 0000A32F C3                     <1>       retn
30424                                  <1>
30425                                  <1> make_sub_directory:
30426                                  <1>       ; 16/10/2016
30427                                  <1>       ; 02/03/2016, 03/03/2016
30428                                  <1>       ; 26/02/2016, 27/02/2016
30429                                  <1>       ; 21/02/2016 (TRDOS 386 = TRDOS v2.0)
30430                                  <1>       ; 01/08/2011 (DIR.ASM, 'proc_make_directory')
30431                                  <1>       ; 10/07/2010
30432                                  <1>       ; INPUT ->
30433                                  <1>       ;     ESI = ASCIIZ Directory Name
30434                                  <1>       ;     CL = Directory Attributes
30435                                  <1>       ; OUTPUT ->
30436                                  <1>       ;     EAX = New sub dir's first cluster
30437                                  <1>       ;     ESI = Logical Dos Drv Descr. Table Addr.
30438                                  <1>       ;     CF = 1 -> error code in AL (EAX)
30439                                  <1>
30440                                  <1>       ;test  cl, 10h  ; directory
30441                                  <1>       ;jz    short loc_make_directory_access_denied
30442                                  <1>       ;test  cl, 08h ; volume name
30443                                  <1>       ;jnz   short loc_make_directory_access_denied
30444                                  <1>
30445 0000A330 80E107                 <1>       and    cl, 07h
30446 0000A333 880D[645D0100]         <1>       mov    byte [mkdir_attrib], cl
30447                                  <1>
30448 0000A339 56                     <1>       push   esi
30449 0000A33A 31DB                   <1>       xor    ebx, ebx
30450 0000A33C 8A3D[E6520100]         <1>       mov    bh, [Current_Drv]
30451 0000A342 BE00010900             <1>       mov    esi, Logical_DOSDisks
30452 0000A347 01DE                   <1>       add    esi, ebx
30453 0000A349 5B                     <1>       pop    ebx
30454                                  <1>
30455                                  <1>       ; 10/07/2010 -> 1st writable disk check for trdos
30456                                  <1>       ; LD_DiskType = 0 for write protection (read only)
30457 0000A34A 807E0101               <1>       cmp    byte [esi+LD_DiskType], 1 ; 0 = Invalid
30458 0000A34E 730B                   <1>       jnb    short loc_mkdir_check_file_sytem
30459                                  <1>       ; 16/10/2016 (13h -> 30)
30460 0000A350 B81E000000             <1>       mov    eax, 30 ; 'Disk write-protected' error
30461 0000A355 BA00000000             <1>       mov    edx, 0
30462                                  <1>       ; err retn: EDX = 0, EBX = Dir name offset
30463                                  <1>       ;ESI = Logical DOS drive description table address
30464 0000A35A C3                     <1>       retn
30465                                  <1>
30466                                  <1> ;loc_make_directory_access_denied:
30467                                  <1>       ;mov   ax, 05h ; access denied (invalid attributes input)
30468                                  <1>       ;stc
30469                                  <1>       ;retn
30470                                  <1>
30471                                  <1> loc_mkdir_check_file_sytem:
30472 0000A35B 807E0301               <1>       cmp    byte [esi+LD_FATType], 1
30473 0000A35F 730B                   <1>       jnb    short loc_mkdir_check_free_sectors
30474                                  <1>
30475                                  <1> loc_make_fs_directory:
30476 0000A361 A1[E0520100]           <1>       mov    eax, [Current_Dir_FCluster]
30477                                  <1>       ; EAX = Parent directory DDT Address
30478                                  <1>       ; ESI = Logical DOS Drive DT Address
30479                                  <1>       ; EBX = Directory name offset (as ASCIIZ name)
30480 0000A366 E8D7150000             <1>       call   make_fs_directory
30481 0000A36B C3                     <1>       retn
30482                                  <1>
30483                                  <1> loc_mkdir_check_free_sectors:
```

```
30484 0000A36C 0FB64613            <1>         movzx   eax, byte [esi+LD_BPB+SecPerClust]
30485 0000A370 8B4E74              <1>       mov   ecx, [esi+LD_FreeSectors]
30486 0000A373 39C1                <1>       cmp   ecx, eax
30487 0000A375 7255                <1>       jb    short loc_mkdir_insufficient_disk_space
30488                              <1>
30489                              <1> loc_make_fat_directory:
30490 0000A377 891D[545D0100]      <1>       mov   [mkdir_DirName_Offset], ebx
30491 0000A37D 890D[605D0100]      <1>       mov   [mkdir_FreeSectors], ecx
30492                              <1>
30493                              <1>       ;mov  al, [esi+LD_BPB+SecPerClust]
30494 0000A383 A2[665D0100]        <1>       mov   byte [mkdir_SecPerClust], al
30495                              <1>
30496                              <1> loc_mkdir_gffc_1:
30497 0000A388 E811180000          <1>       call  get_first_free_cluster
30498 0000A38D 722A                <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_1_save_fcluster:
30505 0000A38F A3[585D0100]        <1>       mov   [mkdir_FFCluster], eax
30506                              <1>
30507                              <1> loc_mkdir_locate_ffe:
30508                              <1>       ; Current directory fcluster <> Directory buffer cluster
30509                              <1>       ; Current directory will be reloaded by
30510                              <1>       ; 'locate_current_dir_file' procedure
30511                              <1>       ;
30512                              <1>       ; ESI = Logical DOS Drive Description Table Address
30513                              <1>       ;push  esi ; 27/02/2016
30514 0000A394 31C0                <1>       xor   eax, eax
30515 0000A396 89C1                <1>         mov  ecx, eax
30516 0000A398 6649                <1>       dec   cx ; FFFFh
30517                              <1>       ; CX = FFFFh -> find first deleted or free entry
30518                              <1>       ; ESI would be ASCIIZ filename address if the call
30519                              <1>       ; would not be for first free or deleted dir entry
30520 0000A39A E8D0FAFFFF          <1>       call  locate_current_dir_file
30521 0000A39F 734C                <1>       jnc   short loc_mkdir_set_ff_dir_entry_1
30522                              <1>       ;pop  esi
30523                              <1>       ; ESI = Logical DOS Drive Description Table Address
30524 0000A3A1 83F802              <1>       cmp   eax, 2  ; cmp al, 2 ; File/Dir not found !
30525 0000A3A4 752B                <1>       jne   short loc_mkdir_stc_return
30526                              <1>
30527                              <1> loc_mkdir_add_new_cluster:
30528 0000A3A6 3805[E5520100]      <1>       cmp   byte [Current_FATType], al ; 2
30529                              <1>       ;cmp  byte ptr [esi+LD_FATType], 2
30530 0000A3AC 770C                <1>       ja    short loc_mkdir_add_new_cluster_check_fsc
30531 0000A3AE 803D[E4520100]01    <1>       cmp   byte [Current_Dir_Level], 1
30532                              <1>       ;cmp  byte [esi+LD_CDirLevel], 1
30533 0000A3B5 7303                <1>       jnb   short loc_mkdir_add_new_cluster_check_fsc
30534                              <1>
30535 0000A3B7 B00C                <1>       mov   al, 12 ; No more files
30536                              <1> loc_mkdir_gffc_retn:
30537 0000A3B9 C3                  <1>       retn
30538                              <1>
30539                              <1> loc_mkdir_add_new_cluster_check_fsc:
30540 0000A3BA 8B0D[605D0100]      <1>       mov   ecx, [mkdir_FreeSectors]
30541                              <1>       ;movzx eax, byte [mkdir_SecPerClust]
30542 0000A3C0 A0[665D0100]        <1>       mov   al, [mkdir_SecPerClust]
30543 0000A3C5 66D1E0              <1>       shl   ax, 1 ; AX = 2 * AX
30544 0000A3C8 39C1                <1>       cmp   ecx, eax
30545 0000A3CA 7350                <1>       jnb   short loc_mkdir_add_new_subdir_cluster
30546                              <1>
30547                              <1> loc_mkdir_insufficient_disk_space:
30548                              <1>       ;mov  edx, ecx
30549                              <1> ;loc_mkdir_gffc_insufficient_disk_space:
30550 0000A3CC 66B82700            <1>       mov   ax, 27h ; MSDOS err => insufficient disk space
30551                              <1>       ; err retn: EDX = Free sectors, EBX = Dir name offset
30552                              <1>         ; ESI -> Dos drive description table address
30553                              <1>       ;; ecx = edx
30554                              <1>       ;
30555 0000A3D0 C3                  <1>       retn
30556                              <1>
30557                              <1> loc_mkdir_stc_return:
30558 0000A3D1 F9                  <1>       stc
30559 0000A3D2 C3                  <1>       retn
30560                              <1>
30561                              <1> loc_mkdir_gffc_2:
30562 0000A3D3 E8C6170000          <1>       call  get_first_free_cluster
30563 0000A3D8 72DF                <1>       jc    short loc_mkdir_gffc_retn
30564                              <1>
30565                              <1> ;loc_mkdir_gffc_1_cont:
30566                              <1>       ;cmp  eax, 2
30567                              <1>       ;jb   short loc_mkdir_gffc_insufficient_disk_space
30568                              <1>
30569                              <1> ;loc_mkdir_gffc_2_save_fcluster:
30570 0000A3DA A3[585D0100]        <1>       mov   [mkdir_FFCluster], eax
30571                              <1>
30572 0000A3DF A1[5C5D0100]        <1>       mov   eax, [mkdir_LastDirCluster]
30573                              <1>
30574 0000A3E4 E844170000          <1>       call  load_FAT_sub_directory
30575 0000A3E9 72CE                <1>       jc    short loc_mkdir_gffc_retn
30576                              <1>
30577 0000A3EB 31FF                <1>       xor   edi, edi
30578                              <1> loc_mkdir_set_ff_dir_entry_1:
30579                              <1>       ; 27/02/2016
30580 0000A3ED 56                  <1>       push  esi ; Logical DOS Drv Desc. Tbl. address
30581                              <1>       ; EDI = Directory Entry Address
30582 0000A3EE 8B35[545D0100]      <1>       mov   esi, [mkdir_DirName_Offset]
30583 0000A3F4 A1[585D0100]        <1>       mov   eax, [mkdir_FFCluster]
30584                              <1>
30585 0000A3F9 66B91000            <1>       mov   cx, 10h      ; CL = Directory attribute
30586                              <1>                   ; CH = 0 -> File size is 0
```

```
30587 0000A3FD 0A0D[645D0100]      <1>       or    cl, [mkdir_attrib] ; S, H, R
30588 0000A403 E8B0010000          <1>       call  make_directory_entry
30589                              <1>
30590 0000A408 5E                  <1>       pop   esi
30591                              <1>
30592 0000A409 C605[105B0100]02    <1>       mov   byte [DirBuff_ValidData], 2
30593 0000A410 E880020000          <1>       call  save_directory_buffer
30594 0000A415 0F83DA000000        <1>         jnc     loc_mkdir_set_ff_dir_entry_2
30595                              <1>
30596                              <1> loc_mkdir_return:
30597 0000A41B C3                  <1>       retn
30598                              <1>
30599                              <1> loc_mkdir_add_new_subdir_cluster:
30600 0000A41C 8B15[155B0100]      <1>       mov   edx, [DirBuff_Cluster]
30601 0000A422 8915[5C5D0100]      <1>       mov   [mkdir_LastDirCluster], edx
30602                              <1>
30603 0000A428 A1[585D0100]        <1>       mov   eax, [mkdir_FFCluster]
30604 0000A42D E8FB160000          <1>       call  load_FAT_sub_directory
30605 0000A432 72E7                <1>       jc    short loc_mkdir_return
30606                              <1>       ; eax = 0
30607                              <1>       ; ecx =  directory buffer sector count (<= 128)
30608                              <1>
30609                              <1> pass_mkdir_add_new_subdir_cluster:
30610 0000A434 29FF                <1>       sub   edi, edi ; 0
30611                              <1>       ;mov  al, 128 ; double word
30612                              <1>       ;mul  ecx ; ecx =  directory buffer sector count
30613                              <1>       ;mov  ecx, eax
30614                              <1>       ;shl  cx, 7 ; 128 * sector count
30615 0000A436 668B4611            <1>       mov   ax, [esi+LD_BPB+BytesPerSec] ; 512
30616 0000A43A 66C1E802            <1>       shr   ax, 2 ; 'byte count / 4' for 'stosd'
30617 0000A43E 66F7E1              <1>       mul   cx ; max = 128*(512/4) -> 16384 (stosd)
30618 0000A441 6689C1              <1>       mov   cx, ax
30619 0000A444 6629C0              <1>       sub   ax, ax ; 0
30620 0000A447 F3AB                <1>       rep   stosd ; clear directory buffer
30621                              <1>
30622 0000A449 C605[105B0100]02    <1>       mov   byte [DirBuff_ValidData], 2
30623 0000A450 E840020000          <1>       call  save_directory_buffer
30624 0000A455 72C4                <1>       jc    short loc_mkdir_return
30625                              <1>
30626                              <1> loc_mkdir_save_added_cluster:
30627 0000A457 A1[5C5D0100]        <1>       mov   eax, [mkdir_LastDirCluster]
30628 0000A45C 8B0D[585D0100]      <1>       mov   ecx, [mkdir_FFCluster]
30629                              <1>       ; 01/03/2016
30630 0000A462 31D2                <1>       xor   edx, edx
30631 0000A464 8915[065B0100]      <1>       mov   [FAT_ClusterCounter], edx ; 0 ; reset
30632 0000A46A E802180000          <1>       call  update_cluster
30633 0000A46F 7304                <1>       jnc   short loc_mkdir_save_fat_buffer_0
30634 0000A471 09C0                <1>       or    eax, eax ; EAX = 0 -> cluster value is 0 or eocc
30635 0000A473 7518                <1>       jnz   short loc_mkdir_save_fat_buffer_stc_retn
30636                              <1>
30637                              <1> loc_mkdir_save_fat_buffer_0:
30638 0000A475 A1[585D0100]        <1>       mov   eax, [mkdir_FFCluster]
30639 0000A47A A3[5C5D0100]        <1>       mov   [mkdir_LastDirCluster], eax
30640                              <1>
30641 0000A47F 31C9                <1>       xor   ecx, ecx
30642 0000A481 49                  <1>       dec   ecx ; FFFFFFFFh
30643                              <1>       ; ESI = Logical DOS Drive Description Table address
30644 0000A482 E8EA170000          <1>       call  update_cluster
30645 0000A487 731A                <1>       jnc   short loc_mkdir_save_fat_buffer_1
30646 0000A489 09C0                <1>       or    eax, eax
30647 0000A48B 7416                <1>       jz    short loc_mkdir_save_fat_buffer_1
30648                              <1>
30649                              <1> loc_mkdir_save_fat_buffer_stc_retn:
30650                              <1>       ; 01/03/2016
30651 0000A48D 803D[065B0100]01    <1>       cmp   byte [FAT_ClusterCounter], 1
30652 0000A494 720C                <1>       jb    short loc_mkdir_save_fat_buffer_retn
30653                              <1>
30654 0000A496 66BB00FF            <1>       mov   bx, 0FF00h ; recalculate free space (BL = 0)
30655                              <1>                        ; (BH = FFh -> Use ESI as Drv Param. Tbl.)
30656 0000A49A 50                  <1>       push  eax
30657 0000A49B E8231B0000          <1>       call  calculate_fat_freespace
30658 0000A4A0 58                  <1>       pop   eax
30659 0000A4A1 F9                  <1>       stc
30660                              <1> loc_mkdir_save_fat_buffer_retn:
30661 0000A4A2 C3                  <1>       retn
30662                              <1>
30663                              <1> loc_mkdir_save_fat_buffer_1:
30664                              <1>       ; byte [FAT_BuffValidData] = 2
30665 0000A4A3 E8861A0000          <1>       call  save_fat_buffer
30666 0000A4A8 72E3                <1>       jc    short loc_mkdir_save_fat_buffer_stc_retn
30667                              <1>
30668                              <1>       ; 01/03/2016
30669 0000A4AA 803D[065B0100]01    <1>       cmp   byte [FAT_ClusterCounter], 1
30670 0000A4B1 721B                <1>       jb    short loc_mkdir_save_fat_buffer_2
30671                              <1>
30672                              <1>       ; ESI = Logical DOS Drive Description Table address
30673 0000A4B3 A1[065B0100]        <1>       mov   eax, [FAT_ClusterCounter]
30674 0000A4B8 66BB01FF            <1>       mov   bx, 0FF01h ; add free clusters
30675 0000A4BC E8021B0000          <1>       call  calculate_fat_freespace
30676                              <1>
30677                              <1>       ;inc  eax ; 0FFFFFFFFh -> 0 ; recalculation is needed!
30678                              <1>       ;jnz  short loc_mkdir_save_fat_buffer_2
30679                              <1>
30680                              <1>       ; ecx > 0 -> Recalculation is needed
30681 0000A4C1 09C9                <1>       or    ecx, ecx
30682 0000A4C3 7409                <1>       jz    short loc_mkdir_save_fat_buffer_2
30683                              <1>
30684 0000A4C5 66BB00FF            <1>       mov   bx, 0FF00h ; ; recalculate free space
30685 0000A4C9 E8F51A0000          <1>       call  calculate_fat_freespace
30686                              <1>
30687                              <1> loc_mkdir_save_fat_buffer_2:
30688 0000A4CE C605[675D0100]01    <1>       mov   byte [mkdir_add_new_cluster], 1
30689 0000A4D5 E9C4000000          <1>       jmp   loc_mkdir_upd_parent_dir_lmdt
```

```
30690                                  <1>
30691                                  <1> loc_mkdir_update_sub_dir_cluster:
30692 0000A4DA A1[585D0100]              <1>       mov    eax, [mkdir_FFCluster]
30693 0000A4DF 29C9                      <1>       sub    ecx, ecx ; 0
30694                                    <1>       ; 01/03/2016
30695 0000A4E1 890D[065B0100]            <1>       mov    [FAT_ClusterCounter], ecx ; 0 ; Reset
30696 0000A4E7 49                        <1>       dec    ecx ; 0FFFFFFFFh
30697                                    <1>
30698                                    <1>       ; ESI = Logical DOS Drive Desciption Table address
30699 0000A4E8 E884170000                <1>       call   update_cluster
30700 0000A4ED 7379                      <1>       jnc    short loc_mkdir_save_fat_buffer_3
30701 0000A4EF 09C0                      <1>       or     eax, eax ; EAX = 0 -> cluster value is 0 or eocc
30702 0000A4F1 7475                      <1>       jz     short loc_mkdir_save_fat_buffer_3
30703                                    <1>       ; 01/03/2016
30704 0000A4F3 EB98                      <1>       jmp    short loc_mkdir_save_fat_buffer_stc_retn
30705                                    <1>
30706                                    <1> loc_mkdir_set_ff_dir_entry_2:
30707                                    <1>       ; ESI = Logical DOS Drive Description Table address
30708 0000A4F5 A1[585D0100]              <1>       mov    eax, [mkdir_FFCluster]
30709                                    <1>       ; Load disk sectors as a directory cluster
30710 0000A4FA E82E160000                <1>       call   load_FAT_sub_directory
30711 0000A4FF 7266                      <1>       jc     short retn_make_fat_directory
30712                                    <1>
30713                                    <1>       ; eax = 0
30714                                    <1>       ; ecx =  directory buffer sector count (<= 128)
30715                                    <1>
30716 0000A501 BF40000800                <1>       mov    edi, Directory_Buffer + 64 ; 26/02/2016
30717                                    <1>
30718                                    <1>       ; 02/03/2016
30719 0000A506 668B4611                  <1>       mov    ax, [esi+LD_BPB+BytesPerSec] ; 512
30720 0000A50A 66C1E802                  <1>       shr    ax, 2 ; 'byte count / 4' for 'stosd'
30721 0000A50E F7E1                      <1>       mul    ecx
30722 0000A510 89C1                      <1>       mov    ecx, eax
30723 0000A512 6629C0                    <1>       sub    ax, ax
30724 0000A515 F3AB                      <1>       rep    stosd
30725                                    <1>
30726                                    <1>       ;;mov  al, 128 ; double word
30727                                    <1>       ;;mul  ecx ; ecx =  directory buffer sector count
30728                                    <1>       ;;mov  ecx, eax
30729                                    <1>       ;shl   cx, 7 ; 128 * sector count
30730                                    <1>       ;;sub  eax, eax
30731                                    <1>       ;;sub  al, al ; 0
30732                                    <1>       ;rep   stosd ; clear directory buffer
30733                                    <1>
30734 0000A517 BF00000800                <1>       mov    edi, Directory_Buffer ; 26/02/2016
30735                                    <1>
30736 0000A51C 56                        <1>       push   esi
30737                                    <1>
30738 0000A51D BE[685D0100]              <1>       mov    esi, mkdir_Name
30739 0000A522 66C7062E00                <1>       mov    word [esi], 2Eh ; db '.', '0'
30740                                    <1>
30741 0000A527 A1[585D0100]              <1>       mov    eax, [mkdir_FFCluster]
30742 0000A52C 66B91000                  <1>       mov    cx, 10h ; CL = Directory attribute
30743                                    <1>                      ; CH = 0 -> File size is 0
30744 0000A530 E883000000                <1>       call   make_directory_entry
30745                                    <1>
30746 0000A535 BF20000800                <1>       mov    edi, Directory_Buffer + 32 ; 26/02/2016
30747                                    <1>
30748                                    <1>       ; 03/03/2016
30749                                    <1>       ; Following modification has been done according to
30750                                    <1>       ; 'Microsoft Extensible Firmware Initiative
30751                                    <1>       ; FAT32 File System Specification' document,
30752                                    <1>       ; 'FAT: General Overview of On-Disk Format–Page 25'.
30753                                    <1>       ; "Finally, you set DIR_FstClusLO and DIR_FstClusHI
30754                                    <1>       ; for the dotdot entry (the second entry) to the
30755                                    <1>       ; first cluster number of the directory in which you
30756                                    <1>       ; just created the directory (value is 0 if this directory
30757                                    <1>       ; is the root directory even for FAT32 volumes)."
30758                                    <1>       ; (Correctness of this modification has been verified
30759                                    <1>       ;  by using Windows 98 'scandisk.exe'.)
30760                                    <1>
30761 0000A53A 29C0                      <1>       sub    eax, eax
30762 0000A53C 3805[E4520100]            <1>       cmp    byte [Current_Dir_Level], al ; 0
30763 0000A542 7605                      <1>       jna    short loc_mkdir_set_ff_dir_entry_3
30764 0000A544 A1[E0520100]              <1>       mov    eax, [Current_Dir_FCluster] ; parent dir
30765                                    <1> loc_mkdir_set_ff_dir_entry_3:
30766 0000A549 66C746012E00              <1>       mov    word [esi+1], 2Eh ; db '.', '0'
30767                                    <1>
30768                                    <1>       ;mov   cx, 10h
30769 0000A54F E864000000                <1>       call   make_directory_entry
30770                                    <1>
30771 0000A554 5E                        <1>       pop    esi
30772                                    <1>
30773 0000A555 C605[105B0100]02          <1>       mov    byte [DirBuff_ValidData], 2
30774 0000A55C E834010000                <1>       call   save_directory_buffer
30775 0000A561 0F8373FFFFFF              <1>         jnc     loc_mkdir_update_sub_dir_cluster
30776                                    <1>
30777                                    <1> retn_make_fat_directory:
30778 0000A567 C3                        <1>       retn
30779                                    <1>
30780                                    <1> loc_mkdir_save_fat_buffer_3:
30781                                    <1>       ; 01/03/2016
30782                                    <1>       ; byte [FAT_BuffValidData] = 2
30783 0000A568 E8C1190000                <1>       call   save_fat_buffer
30784 0000A56D 0F821AFFFFFF              <1>         jc      loc_mkdir_save_fat_buffer_stc_retn
30785                                    <1>
30786 0000A573 803D[065B0100]01          <1>       cmp    byte [FAT_ClusterCounter], 1
30787 0000A57A 721B                      <1>       jb     short loc_mkdir_save_fat_buffer_4
30788                                    <1>
30789                                    <1>       ; ESI = Logical DOS Drive Description Table address
30790 0000A57C A1[065B0100]              <1>       mov    eax, [FAT_ClusterCounter]
30791 0000A581 66BB01FF                  <1>       mov    bx, 0FF01h ; add free clusters
30792 0000A585 E8391A0000                <1>       call   calculate_fat_freespace
```

```
30793                                  <1>
30794                                  <1>       ;inc   eax ; 0FFFFFFFFh -> 0 ; recalculation is needed!
30795                                  <1>         ;jnz    short loc_mkdir_save_fat_buffer_4
30796                                  <1>
30797                                  <1>       ; ecx > 0 -> Recalculation is needed
30798 0000A58A 09C9                    <1>       or     ecx, ecx
30799 0000A58C 7409                    <1>         jz      short loc_mkdir_save_fat_buffer_4
30800                                  <1>
30801 0000A58E 66BB00FF                <1>       mov    bx, 0FF00h ; recalculate free space
30802 0000A592 E82C1A0000              <1>       call   calculate_fat_freespace
30803                                  <1>
30804                                  <1> loc_mkdir_save_fat_buffer_4:
30805 0000A597 C605[675D0100]00        <1>       mov    byte [mkdir_add_new_cluster], 0
30806                                  <1>
30807                                  <1> loc_mkdir_upd_parent_dir_lmdt:
30808 0000A59E E88D010000              <1>       call   update_parent_dir_lmdt
30809                                  <1>
30810                                  <1>       ; 01/03/2016
30811 0000A5A3 803D[675D0100]00        <1>       cmp    byte [mkdir_add_new_cluster], 0
30812 0000A5AA 0F8723FEFFFF            <1>         ja      loc_mkdir_gffc_2
30813                                  <1>
30814                                  <1> loc_mkdir_retn_new_dir_cluster:
30815 0000A5B0 A1[585D0100]            <1>       mov    eax, [mkdir_FFCluster]
30816 0000A5B5 31D2                    <1>       xor    edx, edx
30817                                  <1> loc_mkdir_retn:
30818 0000A5B7 C3                      <1>       retn
30819                                  <1>
30820                                  <1> make_directory_entry:
30821                                  <1>       ; 02/03/2016
30822                                  <1>       ; 21/02/2016 (TRDOS 386 = TRDOS v2.0)
30823                                  <1>       ; 09/08/2010 (DIR.ASM, 'proc_make_directory_entry')
30824                                  <1>       ; 17/07/2010
30825                                  <1>       ; INPUT ->
30826                                  <1>       ;      EDI = Directory Entry Address
30827                                  <1>       ;      ESI = Dot File Name Location
30828                                  <1>       ;      EAX = First Cluster
30829                                  <1>       ;      File Size = 0 (Must be set later)
30830                                  <1>       ;      CL = Attributes
30831                                  <1>       ;      CH = 0 (File size = 0)
30832                                  <1>       ;      (If CH>0, File size is in dword [EBX]) (*)
30833                                  <1>       ; OUTPUT ->
30834                                  <1>       ;      EDI = Directory Entry Address
30835                                  <1>       ;      ESI = Dot File Name Location (Capitalized)
30836                                  <1>       ;      If CH input = 0, File Size = 0
30837                                  <1>       ;      Otherwise file size is as dword [EBX] (*)
30838                                  <1>       ;      DX = Date, AX = Time in DOS Dir Entry format
30839                                  <1>       ;      EBX = same
30840                                  <1>       ;      ECX = same
30841                                  <1>
30842 0000A5B8 51                      <1>       push   ecx
30843                                  <1>
30844 0000A5B9 884F0B                  <1>       mov    [edi+11], cl ; Attributes
30845 0000A5BC 6689471A                <1>       mov    [edi+26], ax ; FClusterLw, 26
30846 0000A5C0 C1E810                  <1>       shr    eax, 16
30847 0000A5C3 66894714                <1>       mov    [edi+20], ax ; FClusterHw, 20
30848 0000A5C7 6631C0                  <1>       xor    ax, ax
30849 0000A5CA 6689470C                <1>       mov    [edi+12], ax ; NTReserved, 12
30850                                  <1>                           ; CrtTimeTenth, 13
30851 0000A5CE 08ED                    <1>       or     ch, ch
30852 0000A5D0 7402                    <1>       jz     short loc_make_direntry_set_filesize
30853                                  <1>
30854 0000A5D2 8B03                    <1>       mov    eax, [ebx]
30855                                  <1>
30856                                  <1> loc_make_direntry_set_filesize:
30857 0000A5D4 89471C                  <1>       mov    [edi+28], eax ; FileSize, 28
30858                                  <1>
30859 0000A5D7 E88AFBFFFF              <1>       call   convert_file_name
30860                                  <1>       ;EDI = Dir Entry Format File Name Location
30861                                  <1>       ;ESI = Dot File Name Location (capitalized)
30862                                  <1>
30863 0000A5DC E816000000              <1>       call   convert_current_date_time
30864                                  <1>       ; OUTPUT -> DX = Date in dos dir entry format
30865                                  <1>         ;       AX = Time in dos dir entry format
30866 0000A5E1 6689470E                <1>       mov    [edi+14], ax ; CrtTime, 14
30867 0000A5E5 66895710                <1>       mov    [edi+16], dx ; CrtDate, 16
30868 0000A5E9 66895712                <1>       mov    [edi+18], dx ; LastAccDate, 18
30869 0000A5ED 66894716                <1>       mov    [edi+22], ax ; WrtTime, 14
30870 0000A5F1 66895718                <1>       mov    [edi+24], dx ; WrtDate, 16
30871 0000A5F5 59                      <1>       pop    ecx
30872                                  <1>
30873 0000A5F6 C3                      <1>       retn
30874                                  <1>
30875                                  <1> convert_current_date_time:
30876                                  <1>       ; 21/02/2016 (TRDOS 386 = TRDOS v2.0)
30877                                  <1>       ; 13/06/2010 (DIR.ASM, 'proc_convert_current_date_time')
30878                                  <1>       ; converts date&time to dos dir entry format
30879                                  <1>       ; INPUT -> none
30880                                  <1>       ; OUTPUT -> DX = Date in dos dir entry format
30881                                  <1>       ;           AX = Time in dos dir entry format
30882                                  <1>
30883 0000A5F7 B404                    <1>       mov    ah, 04h ; Return Current Date
30884 0000A5F9 E87CB3FFFF              <1>       call   int1Ah
30885                                  <1>
30886 0000A5FE 88E8                    <1>       mov    al, ch ; <- century BCD
30887 0000A600 240F                    <1>       and    al, 0Fh
30888 0000A602 88EC                    <1>       mov    ah, ch
30889 0000A604 C0EC04                  <1>       shr    ah, 4
30890 0000A607 D50A                    <1>       aad
30891 0000A609 88C5                    <1>       mov    ch, al ; -> century
30892                                  <1>
30893 0000A60B 88C8                    <1>       mov    al, cl ; <- year BCD
30894 0000A60D 240F                    <1>       and    al, 0Fh
30895 0000A60F 88CC                    <1>       mov    ah, cl
```

```
30896 0000A611 C0EC04              <1>       shr   ah, 4
30897 0000A614 D50A                <1>       aad
30898 0000A616 88C1                <1>       mov   cl, al ; -> year
30899                              <1>
30900 0000A618 88E8                <1>       mov   al, ch
30901 0000A61A B464                <1>       mov   ah, 100
30902 0000A61C F6E4                <1>       mul   ah
30903 0000A61E 30ED                <1>       xor   ch, ch
30904 0000A620 6601C8              <1>       add   ax, cx
30905 0000A623 662DBC07            <1>       sub   ax, 1980 ; ms-dos epoch
30906 0000A627 6689C1              <1>       mov   cx, ax
30907                              <1>
30908 0000A62A 88F0                <1>       mov   al, dh ; <- month in bcd
30909 0000A62C 240F                <1>       and   al, 0Fh
30910 0000A62E 88F4                <1>       mov   ah, dh
30911 0000A630 C0EC04              <1>       shr   ah, 4
30912 0000A633 D50A                <1>       aad
30913 0000A635 88C6                <1>       mov   dh, al ; -> month
30914                              <1>
30915 0000A637 88D0                <1>       mov   al, dl ; <- day BCD
30916 0000A639 240F                <1>       and   al, 0Fh
30917 0000A63B 88D4                <1>       mov   ah, dl
30918 0000A63D C0EC04              <1>       shr   ah, 4
30919 0000A640 D50A                <1>       aad
30920 0000A642 88C2                <1>       mov   dl, al ; -> day
30921                              <1>
30922 0000A644 88C8                <1>       mov   al, cl ; count of years from 1980
30923 0000A646 66C1E004            <1>       shl   ax, 4
30924 0000A64A 08F0                <1>       or    al, dh ; month of year, 1 to 12
30925 0000A64C 66C1E005            <1>       shl   ax, 5
30926 0000A650 08D0                <1>       or    al, dl ; day of year, 1 to 31
30927                              <1>
30928 0000A652 6650                <1>       push  ax ; push date
30929                              <1>
30930 0000A654 B402                <1>       mov   ah, 02h ; Return Current Time
30931 0000A656 E81FB3FFFF          <1>       call  int1Ah
30932                              <1>
30933 0000A65B 88E8                <1>       mov   al, ch ; <- hours BCD
30934 0000A65D 240F                <1>       and   al, 0Fh
30935 0000A65F 88EC                <1>       mov   ah, ch
30936 0000A661 C0EC04              <1>       shr   ah, 4
30937 0000A664 D50A                <1>       aad
30938 0000A666 88C5                <1>       mov   ch, al ; -> hours
30939                              <1>
30940 0000A668 88C8                <1>       mov   al, cl ; <- minutes BCD
30941 0000A66A 240F                <1>       and   al, 0Fh
30942 0000A66C 88CC                <1>       mov   ah, cl
30943 0000A66E C0EC04              <1>       shr   ah, 4
30944 0000A671 D50A                <1>       aad
30945 0000A673 88C1                <1>       mov   cl, al ; -> minutes
30946                              <1>
30947 0000A675 88F0                <1>       mov   al, dh ; <- seconds BCD
30948 0000A677 240F                <1>       and   al, 0Fh
30949 0000A679 88F4                <1>       mov   ah, dh
30950 0000A67B C0EC04              <1>       shr   ah, 4
30951 0000A67E D50A                <1>       aad
30952 0000A680 88C6                <1>       mov   dh, al ; -> seconds
30953                              <1>
30954 0000A682 88E8                <1>       mov   al, ch ; hours
30955 0000A684 66C1E006            <1>       shl   ax, 6
30956 0000A688 08C8                <1>       or    al, cl ; minutes
30957 0000A68A 66C1E005            <1>       shl   ax, 5
30958 0000A68E D0EE                <1>       shr   dh, 1 ; 2 seconds
30959                              <1>       ; There is a bug in TRDOS v1 here !
30960                              <1>       ; it was 'or al, dl' !
30961 0000A690 08F0                <1>       or    al, dh ; seconds
30962                              <1>
30963 0000A692 665A                <1>       pop   dx ; pop date
30964                              <1>
30965 0000A694 C3                  <1>       retn
30966                              <1>
30967                              <1> save_directory_buffer:
30968                              <1>       ; 15/10/2016
30969                              <1>       ; 23/03/2016
30970                              <1>       ; 26/02/2016
30971                              <1>       ; 22/02/2016 (TRDOS 386 = TRDOS v2.0)
30972                              <1>       ; 01/08/2011
30973                              <1>       ; 14/03/2010
30974                              <1>       ; INPUT ->
30975                              <1>       ;      none
30976                              <1>       ; OUTPUT ->
30977                              <1>       ;  cf = 0 -> write OK...
30978                              <1>       ;  cf = 1 -> error code in AL (EAX)
30979                              <1>       ;  cf = 1 & AL = 0Dh => CH & CL = FS & FAT type
30980                              <1>       ;  EBX = Directory Buffer Address
30981                              <1>       ;
30982                              <1>       ;  (EAX, ECX, EDX will be modified)
30983                              <1>
30984 0000A695 BB00000800          <1>       mov   ebx, Directory_Buffer
30985 0000A69A 803D[105B0100]02    <1>       cmp   byte [DirBuff_ValidData], 2
30986 0000A6A1 7403                <1>       je    short loc_save_dir_buffer
30987 0000A6A3 31C0                <1>       xor   eax, eax
30988 0000A6A5 C3                  <1>       retn
30989                              <1>
30990                              <1> loc_save_dir_buffer:
30991 0000A6A6 56                  <1>       push  esi
30992 0000A6A7 31DB                <1>       xor   ebx, ebx
30993 0000A6A9 8A3D[0E5B0100]      <1>         mov     bh, [DirBuff_DRV]
30994 0000A6AF 80EF41              <1>       sub   bh, 'A'
30995 0000A6B2 BE00010900          <1>         mov     esi, Logical_DOSDisks
30996 0000A6B7 01DE                <1>       add   esi, ebx
30997 0000A6B9 668B4E03            <1>         mov     cx, [esi+LD_FATType]
30998                              <1>       ; CH = FS Type (A1h for FS)
```

```
                                <1>       ; CL = FAT Type (0 for FS)
0000A6BD 08C9                   <1>       or     cl, cl
0000A6BF 7433                   <1>       jz     short loc_save_dir_buff_stc_retn
                                <1>
                                <1> loc_save_dir_buffer_check_cluster_no:
0000A6C1 A1[155B0100]           <1>       mov    eax, [DirBuff_Cluster]
0000A6C6 28FF                   <1>       sub    bh, bh ; ebx = 0
0000A6C8 09C0                   <1>       or     eax, eax
0000A6CA 7540                   <1>       jnz    short loc_save_sub_dir_buffer
0000A6CC 8A25[0F5B0100]         <1>       mov    ah, [DirBuff_FATType]
0000A6D2 FEC3                   <1>       inc    bl ;  bl = 1
0000A6D4 38DC                   <1>       cmp    ah, bl
0000A6D6 721D                   <1>       jb     short loc_save_dir_buff_inv_data_retn
0000A6D8 FEC3                   <1>       inc    bl ; bl = 2
0000A6DA 38E3                   <1>       cmp    bl, ah
0000A6DC 7217                   <1>       jb     short loc_save_dir_buff_inv_data_retn
                                <1>
                                <1> loc_save_root_dir_buffer:
0000A6DE 668B5E17               <1>       mov    bx, [esi+LD_BPB+RootDirEnts]
0000A6E2 6683C30F               <1>       add    bx, 15
0000A6E6 66C1EB04               <1>       shr    bx, 4 ; 16 dir entries per sector
0000A6EA 6609DB                 <1>       or     bx, bx
0000A6ED 7405                   <1>       jz     short loc_save_dir_buff_stc_retn
                                <1>       ;mov   ecx, ebx
0000A6EF 8B4664                 <1>       mov    eax, [esi+LD_ROOTBegin] ; 26/02/2016
0000A6F2 EB23                   <1>       jmp    short loc_write_directory_to_disk
                                <1>
                                <1> loc_save_dir_buff_stc_retn:
0000A6F4 F9                     <1>       stc
                                <1> loc_save_dir_buff_inv_data_retn:
                                <1>       ; 15/10/2016 (0Dh -> 29)
0000A6F5 B01D                   <1>       mov    al, 29 ; Invalid data !
0000A6F7 C605[105B0100]00       <1>       mov    byte [DirBuff_ValidData], 0
0000A6FE EB05                   <1>       jmp    short loc_save_dir_buff_retn
                                <1>
                                <1> loc_write_directory_to_disk_err:
                                <1>       ; 15/10/2016 (disk write error code, 1Dh -> 18)
0000A700 B812000000             <1>       mov    eax, 18 ; Drive not ready or write error
                                <1>
                                <1> loc_save_dir_buff_retn:
0000A705 BB00000800             <1>       mov    ebx, Directory_Buffer
0000A70A 5E                     <1>       pop    esi
0000A70B C3                     <1>       retn
                                <1>
                                <1> loc_save_sub_dir_buffer:
                                <1>       ; ebx  = 0
0000A70C 83E802                 <1>       sub    eax, 2
0000A70F 8A5E13                 <1>       mov    bl, [esi+LD_BPB+SecPerClust]
0000A712 F7E3                   <1>       mul    ebx
0000A714 034668                 <1>         add     eax, [esi+LD_DATABegin]
                                <1>       ;mov   ecx, ebx
                                <1>
                                <1> loc_write_directory_to_disk:
0000A717 89D9                   <1>       mov    ecx, ebx
0000A719 BB00000800             <1>       mov    ebx, Directory_Buffer
0000A71E E8A64A0000             <1>       call   disk_write
0000A723 72DB                   <1>       jc     short loc_write_directory_to_disk_err
                                <1>
                                <1> loc_save_dir_buff_validate_retn:
0000A725 C605[105B0100]01       <1>       mov    byte [DirBuff_ValidData], 1
0000A72C 31C0                   <1>       xor    eax, eax
                                <1>       ; 26/02/2016
0000A72E EBD5                   <1>       jmp    short loc_save_dir_buff_retn
                                <1>
                                <1> update_parent_dir_lmdt:
                                <1>       ; 22/02/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>       ; 01/08/2011
                                <1>       ; 16/10/2010
                                <1>       ;
                                <1>       ; INPUT ->
                                <1>       ;      none
                                <1>       ; OUTPUT ->
                                <1>       ;      (last modification date & time of the parent dir
                                <1>       ;      will be changed/updated)
                                <1>       ;
                                <1>       ; (EAX, EBX, ECX, EDX, EDI will be changed)
                                <1>
0000A730 29C0                   <1>       sub    eax, eax
0000A732 8A25[E4520100]         <1>       mov    ah, [Current_Dir_Level]
0000A738 A0[E5520100]           <1>       mov    al, [Current_FATType]
0000A73D 3C01                   <1>       cmp    al, 1
0000A73F 723A                   <1>       jb     short loc_UPDLMDT_proc_retn
                                <1>
                                <1> loc_update_parent_dir_lm_date_time:
0000A741 08E4                   <1>       or     ah, ah
0000A743 7436                   <1>       jz     short loc_UPDLMDT_proc_retn
                                <1>
0000A745 56                     <1>       push   esi ; *
0000A746 8825[885D0100]         <1>       mov    [UPDLMDT_CDirLevel], ah
0000A74C 8B15[E0520100]         <1>       mov    edx, [Current_Dir_FCluster]
0000A752 8915[895D0100]         <1>       mov    [UPDLMDT_CDirFCluster], edx
                                <1>
0000A758 FECC                   <1>       dec    ah
0000A75A B90C000000             <1>       mov    ecx, 12
0000A75F BE[475B0100]           <1>         mov     esi, PATH_Array
                                <1>
0000A764 8825[E4520100]         <1>       mov    [Current_Dir_Level], ah
0000A76A 08E4                   <1>       or     ah, ah
0000A76C 750E                   <1>       jnz    short loc_update_parent_dir_lmdt_load_sub_dir_1
0000A76E 803D[E5520100]02       <1>       cmp    byte [Current_FATType], 2
0000A775 770B                   <1>       ja     short loc_update_parent_dir_lmdt_load_sub_dir_2
0000A777 28C0                   <1>       sub    al, al ; eax = 0
0000A779 EB0A                   <1>       jmp    short loc_update_parent_dir_lmdt_load_sub_dir_3
```

```
31102                                  <1>
31103                                  <1> loc_UPDLMDT_proc_retn:
31104 0000A77B C3                      <1>         retn
31105                                  <1>
31106                                  <1> loc_update_parent_dir_lmdt_load_sub_dir_1:
31107 0000A77C B010                    <1>         mov     al, 16
31108 0000A77E F6E4                    <1>         mul     ah
31109 0000A780 01C6                    <1>         add     esi, eax
31110                                  <1>
31111                                  <1> loc_update_parent_dir_lmdt_load_sub_dir_2:
31112 0000A782 8B460C                  <1>         mov     eax, [esi+12] ; Parent Dir First Cluster
31113                                  <1>
31114                                  <1> loc_update_parent_dir_lmdt_load_sub_dir_3:
31115 0000A785 A3[E0520100]            <1>         mov     [Current_Dir_FCluster], eax
31116                                  <1>
31117 0000A78A 83C610                  <1>         add     esi, 16
31118 0000A78D 66BF[6E5C]              <1>         mov     di, Dir_File_Name
31119 0000A791 F3A4                    <1>         rep     movsb
31120                                  <1>
31121 0000A793 BE00010900              <1>         mov     esi, Logical_DOSDisks
31122 0000A798 29DB                    <1>         sub     ebx, ebx
31123 0000A79A 8A3D[E6520100]          <1>         mov     bh, [Current_Drv]
31124 0000A7A0 01DE                    <1>         add     esi, ebx
31125 0000A7A2 E88FF7FFFF              <1>         call    reload_current_directory
31126 0000A7A7 7232                    <1>         jc      short loc_update_parent_dir_lmdt_restore_cdirlevel
31127                                  <1>
31128                                  <1> loc_update_parent_dir_lmdt_locate_dir:
31129 0000A7A9 BE[6E5C0100]            <1>         mov     esi, Dir_File_Name
31130 0000A7AE 6631C9                  <1>         xor     cx, cx
31131 0000A7B1 66B81008                <1>         mov     ax, 0810h ; Only directories
31132 0000A7B5 E8B5F6FFFF              <1>           call    locate_current_dir_file
31133                                  <1>         ; EDI = DirBuff Directory Entry Address
31134 0000A7BA 721F                    <1>         jc short loc_update_parent_dir_lmdt_restore_cdirlevel
31135                                  <1>
31136 0000A7BC E836FEFFFF              <1>         call    convert_current_date_time
31137 0000A7C1 66895712                <1>         mov     [edi+18], dx ; Last Access Date
31138 0000A7C5 66895718                <1>         mov     [edi+24], dx ; Last Write Date
31139 0000A7C9 66894716                <1>         mov     [edi+22], ax ; Last Write Time
31140                                  <1>
31141 0000A7CD C605[105B0100]02        <1>         mov     byte [DirBuff_ValidData], 2
31142 0000A7D4 E8BCFEFFFF              <1>         call    save_directory_buffer
31143 0000A7D9 7200                    <1>         jc      short loc_update_parent_dir_lmdt_restore_cdirlevel
31144                                  <1>         ;xor    al, al
31145                                  <1> loc_update_parent_dir_lmdt_restore_cdirlevel:
31146                                  <1>         ;current directory level restoration
31147 0000A7DB 8A25[885D0100]          <1>         mov     ah, [UPDLMDT_CDirLevel]
31148 0000A7E1 8825[E4520100]          <1>         mov     [Current_Dir_Level], ah
31149 0000A7E7 8B15[895D0100]          <1>           mov     edx, [UPDLMDT_CDirFCluster]
31150 0000A7ED 8915[E0520100]          <1>         mov     [Current_Dir_FCluster], edx
31151                                  <1>
31152 0000A7F3 5E                      <1>         pop     esi ; *
31153 0000A7F4 C3                      <1>         retn
31154                                  <1>
31155                                  <1> delete_longname:
31156                                  <1>         ; 27/02/2016 (TRDOS 386 = TRDOS v2.0)
31157                                  <1>         ; 01/08/2011 (DIR.ASM, 'proc_delete_longname')
31158                                  <1>         ; 14/03/2010
31159                                  <1>         ; INPUT ->
31160                                  <1>         ;       EAX = Directory Entry (Index) Number (< 65536)
31161                                  <1>         ; OUTPUT ->
31162                                  <1>         ;       cf = 0 -> OK  (EAX = 0)
31163                                  <1>         ;       cf = 1 -> error code in EAX (AL)
31164                                  <1>         ;
31165                                  <1>         ; (Modified registers: EAX, EDX, ECX, EBX, EDI)
31166                                  <1>
31167 0000A7F5 66A3[B85D0100]          <1>         mov     [DLN_EntryNumber], ax
31168 0000A7FB C605[BA5D0100]40        <1>           mov     byte [DLN_40h], 40h
31169                                  <1>
31170 0000A802 E858000000              <1>         call    locate_current_dir_entry
31171 0000A807 7308                    <1>         jnc     short loc_dln_check_attributes
31172 0000A809 C3                      <1>         retn
31173                                  <1>
31174                                  <1> loc_dln_longname_not_found:
31175 0000A80A B802000000              <1>         mov     eax, 2
31176 0000A80F F9                      <1>         stc
31177 0000A810 C3                      <1>         retn
31178                                  <1>
31179                                  <1> loc_dln_check_attributes:
31180 0000A811 B00F                    <1>         mov     al, 0Fh  ; long name
31181 0000A813 8A670B                  <1>         mov     ah, [edi+0Bh] ; dir entry attributes
31182 0000A816 38C4                    <1>         cmp     ah, al
31183 0000A818 75F0                    <1>         jne     short loc_dln_longname_not_found
31184 0000A81A 8A27                    <1>         mov     ah, [edi]
31185 0000A81C 2A25[BA5D0100]          <1>         sub     ah, [DLN_40h]
31186 0000A822 76E6                    <1>         jna     short loc_dln_longname_not_found
31187 0000A824 80FC14                  <1>         cmp     ah, 14h ; 84-64=20 -> 20*13=260 bytes
31188 0000A827 77E1                    <1>         ja      short loc_dln_longname_not_found
31189                                  <1>
31190 0000A829 C607E5                  <1>         mov     byte [edi], 0E5h  ; deleted sign
31191 0000A82C C605[105B0100]02        <1>         mov     byte [DirBuff_ValidData], 2 ; changed/write sign
31192 0000A833 C605[BA5D0100]00        <1>         mov     byte [DLN_40h], 0 ; 40h -> 0
31193                                  <1>
31194                                  <1> loc_dln_delete_next_ln_entry:
31195 0000A83A 80FC01                  <1>         cmp     ah, 1
31196 0000A83D 7616                    <1>         jna     short loc_dln_longname_retn
31197                                  <1> loc_dln_delete_next_ln_entry_0:
31198 0000A83F 66FF05[B85D0100]        <1>         inc     word [DLN_EntryNumber]
31199 0000A846 0FB705[B85D0100]        <1>         movzx   eax, word [DLN_EntryNumber]
31200 0000A84D E80D000000              <1>         call    locate_current_dir_entry
31201 0000A852 73BD                    <1>         jnc     short loc_dln_check_attributes
31202                                  <1>
31203                                  <1> loc_dln_longname_stc_retn:
31204 0000A854 C3                      <1>         retn
```

```
31205                                  <1>
31206                                  <1> loc_dln_longname_retn:
31207                                  <1>       ;cmp   byte [DirBuff_ValidData], 2
31208                                  <1>       ;jne   short loc_dln_longname_retn_xor_eax
31209 0000A855 E83BFEFFFF               <1>       call   save_directory_buffer
31210 0000A85A 72F8                     <1>       jc     short loc_dln_longname_stc_retn
31211                                  <1>
31212                                  <1> loc_dln_longname_retn_xor_eax:
31213 0000A85C 31C0                     <1>       xor    eax, eax
31214 0000A85E C3                       <1>       retn
31215                                  <1>
31216                                  <1> locate_current_dir_entry:
31217                                  <1>       ; 16/10/2016
31218                                  <1>       ; 15/10/2016
31219                                  <1>       ; 23/03/2016
31220                                  <1>       ; 27/02/2016 (TRDOS 386 = TRDOS v2.0)
31221                                  <1>       ; 01/08/2011 (DIR.ASM, 'proc_locate_current_dir_entry')
31222                                  <1>       ; 07/03/2010
31223                                  <1>       ; INPUT ->
31224                                  <1>       ;     EAX = Directory Entry (Index) Number (< 65536)
31225                                  <1>       ; OUTPUT ->
31226                                  <1>       ;     EDI = Directory Entry Address
31227                                  <1>       ;     EAX = Cluster Number of Directory Buffer
31228                                  <1>       ;     EBX = Directory Buffer Entry Offset
31229                                  <1>       ;     ECX = DirBuff Valid Data identifier (CL)
31230                                  <1>       ;     If CF = 0 and CL = 2 then
31231                                  <1>       ;        directory buffer modified and
31232                                  <1>       ;        must be written to disk.
31233                                  <1>       ;     If CF = 0  and CL = 1 then
31234                                  <1>       ;        dir buffer has been written to disk, already.
31235                                  <1>       ;     CF = 1 -> Error code in EAX (AL)
31236                                  <1>       ;
31237                                  <1>       ; (Modified registers: EAX, EDX, ECX, EBX, EDI)
31238                                  <1>
31239                                  <1> loc_locate_current_dir_entry:
31240 0000A85F 56                       <1>       push   esi
31241 0000A860 89C1                     <1>       mov    ecx, eax
31242 0000A862 BA20000000               <1>       mov    edx, 32
31243 0000A867 F7E2                     <1>       mul    edx
31244 0000A869 A3[C45D0100]             <1>       mov    [LCDE_ByteOffset], eax
31245 0000A86E 31DB                     <1>       xor    ebx, ebx
31246 0000A870 8A3D[E6520100]           <1>       mov    bh, [Current_Drv]
31247 0000A876 A0[0E5B0100]             <1>         mov    al, [DirBuff_DRV]
31248 0000A87B 2C41                     <1>       sub    al, 'A'
31249 0000A87D BE00010900               <1>         mov    esi, Logical_DOSDisks
31250 0000A882 01DE                     <1>       add    esi, ebx
31251 0000A884 38C7                     <1>       cmp    bh, al
31252 0000A886 0F8592000000             <1>         jne    loc_lcde_reload_current_directory
31253                                  <1> loc_lcde_cdl_check:
31254 0000A88C 803D[E4520100]00         <1>       cmp    byte [Current_Dir_Level], 0
31255 0000A893 772A                     <1>       ja     short loc_lcde_calc_dirbuff_cluster_offset
31256                                  <1>       ; 27/02/2016
31257                                  <1>       ; TRDOS v1 has bug here for FAT32 fs !
31258                                  <1>       ; (Root Directory Entries for FAT32 = 0)
31259 0000A895 807E0303                 <1>       cmp    byte [esi+LD_FATType], 3  ; FAT32
31260 0000A899 7324                     <1>       jnb    short loc_lcde_calc_dirbuff_cluster_offset
31261                                  <1>
31262                                  <1> loc_lcde_cdl_check_FAT12_16:
31263 0000A89B 668B4617                 <1>       mov    ax, [esi+LD_BPB+RootDirEnts]
31264 0000A89F 6648                     <1>       dec    ax
31265                                  <1>       ;xor   dx, dx
31266 0000A8A1 6639C8                   <1>       cmp    ax, cx ; cx = Directory Entry (Index) Number
31267 0000A8A4 720E                     <1>       jb     short loc_lcde_stc_12h_retn
31268 0000A8A6 66890D[BC5D0100]         <1>       mov    [LCDE_EntryIndex], cx
31269 0000A8AD 31C0                     <1>       xor    eax, eax
31270 0000A8AF E993000000               <1>         jmp    loc_lcde_check_dir_buffer_cluster
31271                                  <1>
31272                                  <1> loc_lcde_stc_12h_retn:
31273 0000A8B4 5E                       <1>       pop    esi
31274 0000A8B5 89CB                     <1>       mov    ebx, ecx
31275 0000A8B7 89D1                     <1>       mov    ecx, edx
31276                                  <1>       ; 16/10/2016 (12h -> 12)
31277 0000A8B9 B80C000000               <1>       mov    eax, 12 ; No more files
31278 0000A8BE C3                       <1>       retn
31279                                  <1>
31280                                  <1> loc_lcde_calc_dirbuff_cluster_offset:
31281 0000A8BF 8A5E13                   <1>       mov    bl, [esi+LD_BPB+SecPerClust]
31282 0000A8C2 30FF                     <1>       xor    bh, bh
31283 0000A8C4 668B4611                 <1>       mov    ax, [esi+LD_BPB+BytesPerSec]
31284 0000A8C8 66F7E3                   <1>       mul    bx
31285 0000A8CB 6609D2                   <1>       or     dx, dx ; If bytes per cluster > 32KB it is invalid
31286 0000A8CE 755D                     <1>       jnz    short loc_lcde_invalid_format
31287                                  <1>       ;mov   ecx, eax
31288 0000A8D0 6689C1                   <1>       mov    cx, ax ; BYTES PER CLUSTER
31289 0000A8D3 A1[C45D0100]             <1>       mov    eax, [LCDE_ByteOffset]
31290                                  <1>       ;sub   edx, edx
31291 0000A8D8 F7F1                     <1>       div    ecx
31292 0000A8DA 3DFFFF0000               <1>       cmp    eax, 65535
31293 0000A8DF 774C                     <1>       ja     short loc_lcde_invalid_format
31294                                  <1>
31295                                  <1>       ; cluster sequence number of directory (< 65536)
31296 0000A8E1 66A3[BE5D0100]           <1>       mov    [LCDE_ClusterSN], ax
31297                                  <1>
31298 0000A8E7 6689D0                   <1>       mov    ax, dx ; byte offset in cluster (directory buffer)
31299 0000A8EA 66BB2000                 <1>       mov    bx, 32 ; ; 1 dir entry = 32 bytes
31300 0000A8EE 6629D2                   <1>         sub    dx, dx  ; 0
31301 0000A8F1 66F7F3                   <1>       div    bx
31302 0000A8F4 66A3[BC5D0100]           <1>       mov    [LCDE_EntryIndex], ax ; dir entry index/sequence number
31303                                  <1>                                      ; (in directory buffer/cluster)
31304                                  <1> loc_lcde_get_current_sub_dir_fcluster:
31305 0000A8FA A1[E0520100]             <1>       mov    eax, [Current_Dir_FCluster]
31306                                  <1>
31307                                  <1> loc_lcde_get_next_cluster:
```

```
31308 0000A8FF 66833D[BE5D0100]00  <1>       cmp    word [LCDE_ClusterSN], 0
31309 0000A907 763E                <1>       jna    short loc_lcde_check_dir_buffer_cluster
31310 0000A909 A3[C05D0100]        <1>       mov    [LCDE_Cluster], eax
31311 0000A90E E834100000          <1>       call   get_next_cluster
31312 0000A913 7220                <1>       jc     short loc_lcde_check_gnc_error
31313 0000A915 66FF0D[BE5D0100]    <1>       dec    word [LCDE_ClusterSN]
31314 0000A91C EBE1                <1>       jmp    short loc_lcde_get_next_cluster
31315                              <1>
31316                              <1> loc_lcde_reload_current_directory:
31317 0000A91E 51                  <1>       push   ecx
31318 0000A91F E812F6FFFF          <1>       call   reload_current_directory
31319 0000A924 59                  <1>       pop    ecx
31320 0000A925 0F8361FFFFFF        <1>         jnc     loc_lcde_cdl_check
31321 0000A92B 5E                  <1>       pop    esi
31322 0000A92C C3                  <1>       retn
31323                              <1>
31324                              <1> loc_lcde_invalid_format:
31325                              <1>       ; 15/10/2016 (0Bh -> 28)
31326 0000A92D B81C000000          <1>       mov    eax, 28 ; Invalid Format !
31327                              <1> loc_lcde_drive_not_ready_read_err:
31328 0000A932 F9                  <1>       stc
31329 0000A933 5E                  <1>       pop    esi
31330 0000A934 C3                  <1>       retn
31331                              <1>
31332                              <1> loc_lcde_check_gnc_error:
31333 0000A935 09C0                <1>       or     eax, eax
31334 0000A937 75F9                <1>       jnz    short loc_lcde_drive_not_ready_read_err
31335 0000A939 66FF0D[BE5D0100]    <1>       dec    word [LCDE_ClusterSN]
31336 0000A940 75EB                <1>       jnz    short loc_lcde_invalid_format
31337 0000A942 A1[C05D0100]        <1>       mov    eax, [LCDE_Cluster]
31338                              <1>
31339                              <1> loc_lcde_check_dir_buffer_cluster:
31340 0000A947 3B05[155B0100]      <1>       cmp    eax, [DirBuff_Cluster]
31341 0000A94D 755C                <1>       jne    short loc_lcde_load_dir_cluster
31342 0000A94F 803D[105B0100]00    <1>       cmp    byte [DirBuff_ValidData], 0
31343 0000A956 7727                <1>       ja     short lcde_check_dir_buffer_cluster_next
31344 0000A958 803D[E4520100]00    <1>       cmp    byte [Current_Dir_Level], 0
31345 0000A95F 775F                <1>       ja     short loc_lcde_load_dir_cluster_0
31346                              <1>       ; 27/02/2016
31347                              <1>       ; TRDOS v1 has bug here for FAT32 fs !
31348 0000A961 807E0303            <1>       cmp    byte [esi+LD_FATType], 3  ; FAT32
31349 0000A965 7359                <1>       jnb    short loc_lcde_load_dir_cluster_0
31350                              <1>       ;
31351 0000A967 0FB74E17            <1>       movzx  ecx, word [esi+LD_BPB+RootDirEnts]
31352 0000A96B 6683C10F            <1>       add    cx, 15 ; round up (16 entries per sector)
31353 0000A96F 66C1E904            <1>       shr    cx, 4 ; 1 sector contains 16 dir entries
31354                              <1>
31355 0000A973 8B4664              <1>         mov     eax, [esi+LD_ROOTBegin]
31356 0000A976 EB54                <1>       jmp    short loc_lcde_load_dir_cluster_1
31357                              <1>
31358                              <1> loc_lcde_validate_dirBuff:
31359 0000A978 C605[105B0100]01    <1>       mov    byte [DirBuff_ValidData], 1
31360                              <1>
31361                              <1> lcde_check_dir_buffer_cluster_next:
31362 0000A97F 0FB71D[BC5D0100]    <1>       movzx  ebx, word [LCDE_EntryIndex]
31363 0000A986 663B1D[135B0100]    <1>       cmp    bx, [DirBuff_LastEntry]
31364 0000A98D 779E                <1>       ja     short loc_lcde_invalid_format
31365 0000A98F B820000000          <1>       mov    eax, 32
31366 0000A994 F7E3                <1>       mul    ebx
31367                              <1>       ;or    edx, edx
31368                              <1>       ;jnz   short loc_lcde_invalid_format
31369                              <1>
31370 0000A996 BF00000800          <1>       mov    edi, Directory_Buffer
31371 0000A99B 01C7                <1>       add    edi, eax ; add entry offset to buffer address
31372                              <1>
31373                              <1> loc_lcde_dir_buffer_last_check:
31374 0000A99D A1[155B0100]        <1>       mov    eax, [DirBuff_Cluster]
31375 0000A9A2 0FB60D[105B0100]    <1>       movzx  ecx, byte [DirBuff_ValidData]
31376                              <1>
31377                              <1> loc_lcde_retn:
31378 0000A9A9 5E                  <1>       pop    esi
31379 0000A9AA C3                  <1>       retn
31380                              <1>
31381                              <1> loc_lcde_load_dir_cluster:
31382                              <1>       ;cmp   byte [DirBuff_ValidData], 2
31383                              <1>       ;jne   short loc_lcde_load_dir_cluster_n2
31384 0000A9AB 50                  <1>       push   eax
31385 0000A9AC E8E4FCFFFF          <1>       call   save_directory_buffer
31386 0000A9B1 58                  <1>       pop    eax
31387 0000A9B2 72F5                <1>       jc     short loc_lcde_retn
31388                              <1>
31389                              <1> loc_lcde_load_dir_cluster_n2:
31390 0000A9B4 C605[105B0100]00    <1>       mov    byte [DirBuff_ValidData], 0
31391 0000A9BB A3[155B0100]        <1>       mov    [DirBuff_Cluster], eax
31392                              <1>
31393                              <1> loc_lcde_load_dir_cluster_0:
31394 0000A9C0 83E802              <1>       sub    eax, 2
31395 0000A9C3 0FB64E13            <1>       movzx  ecx, byte [esi+LD_BPB+SecPerClust]
31396 0000A9C7 F7E1                <1>       mul    ecx
31397 0000A9C9 034668              <1>         add     eax, [esi+LD_DATABegin]
31398                              <1>
31399                              <1> loc_lcde_load_dir_cluster_1:
31400 0000A9CC BB00000800          <1>       mov    ebx, Directory_Buffer
31401                              <1>       ; ecx = sector count
31402 0000A9D1 E802480000          <1>       call   disk_read
31403 0000A9D6 73A0                <1>       jnc    short loc_lcde_validate_dirBuff
31404                              <1>
31405                              <1>       ; 15/10/2016
31406                              <1>       ; (Disk read error instead of drv not ready err)
31407 0000A9D8 B811000000          <1>       mov    eax, 17 ; Drive not ready or read error !
31408 0000A9DD EBCA                <1>       jmp    short loc_lcde_retn
31409                              <1>
31410                              <1>
```

```
                            <1> remove_file:
                            <1>       ; 15/10/2016
                            <1>       ; 28/02/2016 (TRDOS 386 = TRDOS v2.0)
                            <1>       ; 10/04/2011 (FILE.ASM, 'proc_delete_file')
                            <1>       ; 09/08/2010
                            <1>       ; INPUT ->
                            <1>       ;      EDI = Directory Buffer Entry Address
                            <1>       ;       CX = Directory Buffer Entry Counter/Index
                            <1>       ;       BL = Longname Entry Length
                            <1>       ;       BH = Logical DOS Drive Number
                            <1>
0000A9DF 29C0               <1>       sub    eax, eax
0000A9E1 88FC               <1>       mov    ah, bh
0000A9E3 BE00010900         <1>       mov    esi, Logical_DOSDisks
0000A9E8 01C6               <1>       add    esi, eax
                            <1>
0000A9EA 807E0301           <1>       cmp    byte [esi+LD_FATType], 1
0000A9EE 7312               <1>       jnb    short loc_del_fat_file
                            <1>
0000A9F0 807E04A1           <1>       cmp    byte [esi+LD_FSType], 0A1h
0000A9F4 7406               <1>       je     short loc_del_fs_file
                            <1>
                            <1> loc_del_file_invalid_format:
0000A9F6 30E4               <1>       xor    ah, ah
                            <1>       ; 15/10/2016 (0Bh -> 28)
0000A9F8 B01C               <1>       mov    al, 28  ; Invalid Format
0000A9FA F9                 <1>       stc
0000A9FB C3                 <1>       retn
                            <1>
                            <1> loc_del_fs_file:
0000A9FC E83F0F0000         <1>       call   delete_fs_file
0000AA01 C3                 <1>       retn
                            <1>
                            <1> loc_del_fat_file:
0000AA02 E808000000         <1>       call   delete_directory_entry
0000AA07 7205               <1>       jc     short loc_del_file_err_retn
                            <1>
                            <1> loc_delfile_unlink_cluster_chain:
0000AA09 E863170000         <1>       call   truncate_cluster_chain
                            <1>       ;jc    short loc_del_file_err_retn
                            <1>
                            <1> loc_delfile_return:
                            <1> loc_del_file_err_retn:
0000AA0E C3                 <1>       retn
                            <1>
                            <1> delete_directory_entry:
                            <1>       ; 15/10/2016
                            <1>       ; 28/02/2016 (TRDOS 386 = TRDOS v2.0)
                            <1>       ; 01/08/2011 (DIR.ASM, 'proc_delete_directory_entry')
                            <1>       ; 10/04/2011
                            <1>       ; INPUT ->
                            <1>       ;      ESI = Logical Dos Drive Descripton Table Address
                            <1>       ;      EDI = Directory Buffer Entry Address
                            <1>       ;       CX = Directory Buffer Entry Counter/Index
                            <1>       ;       BL = Longname Entry Length
                            <1>       ; OUTPUT ->
                            <1>       ;      ESI = Logical dos drive descripton table address
                            <1>       ;      EAX = First cluster to be truncated/unlinked
                            <1>       ;       CF = 1 -> Error code in EAX (AL)
                            <1>       ;       CF = 0 & BH <> 0 -> LMDT write error  (BH = 1)
                            <1>       ;       CF = 0 & BL <> 0 -> Long name delete error (BL = FFh)
                            <1>       ;
                            <1>       ;  (EDI, EBX, ECX register contents will be changed)
                            <1>
0000AA0F 881D[525D0100]     <1>       mov    [DelFile_LNEL], bl
0000AA15 66890D[505D0100]   <1>       mov    [DelFile_EntryCounter], cx
                            <1>
0000AA1C 668B4714           <1>       mov    ax, [edi+20] ; First Cluster High Word
0000AA20 C1E010             <1>       shl    eax, 16
0000AA23 668B471A           <1>       mov    ax, [edi+26] ; First Cluster Low Word
                            <1>
0000AA27 A3[4C5D0100]       <1>       mov    [DelFile_FCluster], eax
                            <1>
                            <1> loc_del_short_name:
0000AA2C C607E5             <1>       mov    byte [edi], 0E5h  ; Deleted sign
                            <1>
0000AA2F C605[105B0100]02   <1>       mov    byte [DirBuff_ValidData], 2
0000AA36 E85AFCFFFF         <1>       call   save_directory_buffer
0000AA3B 723D               <1>       jc     short loc_delete_direntry_err_return
                            <1>
                            <1> loc_del_long_name:
0000AA3D 0FB615[525D0100]   <1>       movzx  edx, byte [DelFile_LNEL]
0000AA44 08D2               <1>       or     dl, dl
0000AA46 7416               <1>       jz     short loc_del_dir_entry_update_parent_dir_lm_date
                            <1>
0000AA48 8835[525D0100]     <1>       mov    byte [DelFile_LNEL], dh ; 0
                            <1>
0000AA4E 0FB705[505D0100]   <1>       movzx  eax,  word [DelFile_EntryCounter]
0000AA55 29D0               <1>       sub    eax, edx
                            <1>       ;jnc   short loc_del_long_name_continue
0000AA57 7205               <1>       jc     short loc_del_dir_entry_update_parent_dir_lm_date
                            <1>
                            <1> ;loc_del_direntry_inv_data_return: ; 15/10/2016 (0Dh -> 29)
                            <1> ;     mov    eax, 29 ; 0Dh (TRDOS 8086) ; Invalid data
                            <1> ;     retn
                            <1>
                            <1> loc_del_long_name_continue:
                            <1>       ; AX = Directory Entry Number of the long name last entry
0000AA59 E897FDFFFF         <1>       call   delete_longname
                            <1>       ;jc    short loc_delete_direntry_err_return
                            <1>
                            <1> loc_del_dir_entry_update_parent_dir_lm_date:
0000AA5E 801D[525D0100]00   <1>       sbb    byte [DelFile_LNEL], 0 ; 0FFh if cf = 1
```

```
                                <1>
0000AA65 E8C6FCFFFF             <1>       call   update_parent_dir_lmdt
0000AA6A B700                   <1>       mov    bh, 0
0000AA6C 80D700                 <1>       adc    bh, 0
                                <1>
0000AA6F 8A1D[525D0100]         <1>       mov    bl, byte [DelFile_LNEL]
                                <1>
                                <1> loc_delete_direntry_return:
0000AA75 A1[4C5D0100]           <1>       mov    eax, [DelFile_FCluster]
                                <1> loc_delete_direntry_err_return:
0000AA7A C3                     <1>       retn
                                <1>
                                <1> rename_directory_entry:
                                <1>       ; 13/11/2017
                                <1>       ; 15/10/2016
                                <1>       ; 06/03/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>       ; 01/08/2011 (DIR.ASM, 'proc_rename_directory_entry')
                                <1>       ; 19/11/2010
                                <1>       ; INPUT -> (Current Directory)
                                <1>       ;      CX = Directory Entry Number
                                <1>       ;      EAX = First Cluster number of file or directory
                                <1>       ;      EBX = Longname Length (dir entry count) (< 256)
                                <1>       ;      ESI = New file (or directory) name (no path).
                                <1>       ;           (ASCIIZ string)
                                <1>       ; OUTPUT ->
                                <1>       ;      CF = 0 -> successfull
                                <1>       ;      CF = 1 -> error code in EAX (AL)
                                <1>       ;
                                <1>       ; (EAX, EBX, ECX, EDX, ESI, EDI will be changed)
                                <1>
0000AA7B 803D[E5520100]00       <1>       cmp    byte [Current_FATType], 0
0000AA82 7706                   <1>       ja     short loc_rename_directory_entry
                                <1>
0000AA84 E8B80E0000             <1>       call   rename_fs_file_or_directory
0000AA89 C3                     <1>       retn
                                <1>
                                <1> loc_rename_directory_entry:
0000AA8A 881D[525D0100]         <1>       mov    [DelFile_LNEL], bl
0000AA90 66890D[505D0100]       <1>       mov    [DelFile_EntryCounter], cx
0000AA97 A3[4C5D0100]           <1>       mov    [DelFile_FCluster], eax
                                <1>
0000AA9C 0FB7C1                 <1>       movzx  eax, cx
0000AA9F E8BBFDFFFF             <1>       call   locate_current_dir_entry
0000AAA4 7308                   <1>       jnc    short loc_rename_direntry_check_fcluster
                                <1>
                                <1> loc_rename_direntry_pop_retn:
0000AAA6 C3                     <1>       retn
                                <1>
                                <1> loc_rename_direntry_pop_invd_retn:
0000AAA7 F9                     <1>       stc
                                <1> loc_rename_direntry_invd_retn:
                                <1>       ; 15/10/2016 (0Dh -> 29)
0000AAA8 B81D000000             <1>       mov    eax, 29 ; Invalid data
                                <1> loc_rename_retn:
0000AAAD C3                     <1>       retn
                                <1>
                                <1> loc_rename_direntry_check_fcluster:
0000AAAE 668B5714               <1>       mov    dx, [edi+20] ; First Cluster HW
0000AAB2 C1E210                 <1>       shl    edx, 16 ; 13/11/2017
0000AAB5 668B571A               <1>       mov    dx, [edi+26] ; First Cluster LW
0000AAB9 3B15[4C5D0100]         <1>       cmp    edx, [DelFile_FCluster]
0000AABF 75E6                   <1>       jne    short loc_rename_direntry_pop_invd_retn
                                <1>       ; ESI = New file (or directory) name. (ASCIIZ string)
                                <1>       ; 06/03/2016
                                <1>       ; TRDOS v2 - NOTE: 'convert_file_name' procedure
                                <1>       ; has been modified for eliminating following situation.
                                <1>       ;
                                <1>       ; TRDOS v1 - NOTE: If file/dir name is more than 11 bytes
                                <1>       ; without a dot, attributes (edi+11) byte will be overwritten !
                                <1>       ; (Dot file name input must be proper for 11 byte dir entry
                                <1>       ;  type file name output.)
0000AAC1 E8A0F6FFFF             <1>       call   convert_file_name
                                <1>
0000AAC6 C605[105B0100]02       <1>         mov     byte [DirBuff_ValidData], 2
0000AACD E8C3FBFFFF             <1>       call   save_directory_buffer
0000AAD2 72D9                   <1>       jc     short loc_rename_retn
                                <1>
                                <1> loc_rename_direntry_del_ln:
0000AAD4 0FB615[525D0100]       <1>       movzx  edx, byte [DelFile_LNEL]
0000AADB 08D2                   <1>       or     dl, dl
0000AADD 7410                   <1>       jz     short loc_rename_direntry_update_parent_dir_lm_date
                                <1>
0000AADF 0FB705[505D0100]       <1>       movzx  eax, word [DelFile_EntryCounter]
0000AAE6 29D0                   <1>       sub    eax, edx
0000AAE8 72BE                   <1>       jc     short loc_rename_direntry_invd_retn
                                <1>
                                <1> loc_rename_direntry_del_ln_continue:
                                <1>       ; EAX = Directory Entry Number of the long name last entry
0000AAEA E806FDFFFF             <1>       call   delete_longname
                                <1>
                                <1> loc_rename_direntry_update_parent_dir_lm_date:
0000AAEF E83CFCFFFF             <1>       call   update_parent_dir_lmdt
0000AAF4 31C0                   <1>       xor    eax, eax
0000AAF6 C3                     <1>       retn
                                <1>
                                <1> move_source_file_to_destination_file:
                                <1>       ; 15/10/2016
                                <1>       ; 11/03/2016
                                <1>       ; 10/03/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>       ; 01/08/2011 (FILE.ASM)
                                <1>       ; 04/08/2010
                                <1>       ;
                                <1>       ;   Phase 1 -> Check destination file,
```

```
31617                                  <1>       ;              'not found' is required
31618                                  <1>       ;   Phase 2 -> Check source file
31619                                  <1>       ;              'found' and proper attributes is required
31620                                  <1>       ;   Phase 3 -> Make destination directory entry,
31621                                  <1>       ;             add new dir cluster or section if it is required
31622                                  <1>       ;   Phase 4 -> Delete source directory entry.
31623                                  <1>       ;       cf = 1 causes to return before the phase 4.
31624                                  <1>       ;    (source file protection against any possible errors)
31625                                  <1>       ;
31626                                  <1>       ; 08/05/2011 major modification
31627                                  <1>       ;            -> destination file deleting is removed
31628                                  <1>       ;            for msdos move/rename compatibility.
31629                                  <1>       ;            (Access denied error will return if
31630                                  <1>       ;            the destination file is found...)
31631                                  <1>       ; INPUT ->
31632                                  <1>       ;      ESI = Source File Pathname (Asciiz)
31633                                  <1>       ;       EDI = Destination File Pathname (Asciiz)
31634                                  <1>       ;       AL = 0 --> Interrupt (System call)
31635                                  <1>       ;       AL > 0 --> Command Interpreter (Question)
31636                                  <1>       ;       AL = 1 --> Question Phase
31637                                  <1>       ;       AL = 2 --> Progress Phase
31638                                  <1>       ; OUTPUT ->
31639                                  <1>       ;      cf = 0 -> OK
31640                                  <1>       ;       EAX = Destination directory first cluster
31641                                  <1>       ;       ESI = Logical DOS drive description table
31642                                  <1>       ;       EBX = Destination file structure offset
31643                                  <1>       ;       CX = 0 (CX > 0 --> calculate free space error)
31644                                  <1>       ;       cf = 1 -> Error code in EAX (AL)
31645                                  <1>       ;
31646                                  <1>       ;  (EDX, ECX, EBX, ESI, EDI will be changed)
31647                                  <1>
31648 0000AAF7 3C02                   <1>       cmp    al, 2
31649 0000AAF9 0F847F010000           <1>       je     msftdf_df2_check_directory
31650 0000AAFF A2[D25E0100]           <1>       mov    [move_cmd_phase], al
31651                                  <1>
31652                                  <1> msftdf_parse_sf_path:
31653                                  <1>       ; ESI = ASCIIZ pathname (Source)
31654 0000AB04 57                     <1>       push   edi
31655 0000AB05 BF[D05D0100]           <1>       mov    edi, SourceFile_Drv
31656 0000AB0A E822F7FFFF             <1>       call   parse_path_name
31657 0000AB0F 5E                     <1>       pop    esi
31658 0000AB10 7211                   <1>       jc     short msftdf_psf_retn
31659                                  <1>
31660                                  <1> msftdf_parse_df_path:
31661                                  <1>       ; ESI = ASCIIZ pathname   (Destination)
31662 0000AB12 BF[505E0100]           <1>       mov    edi, DestinationFile_Drv
31663 0000AB17 E815F7FFFF             <1>       call   parse_path_name
31664 0000AB1C 7306                   <1>       jnc    short msftdf_check_sf_drv
31665                                  <1>
31666 0000AB1E 3C01                   <1>       cmp    al, 1 ; File or directory name is not existing
31667 0000AB20 7602                   <1>       jna    short msftdf_check_sf_drv
31668                                  <1>
31669                                  <1> msftdf_stc_retn:
31670 0000AB22 F9                     <1>       stc
31671                                  <1> msftdf_psf_retn:
31672 0000AB23 C3                     <1>       retn
31673                                  <1>
31674                                  <1> msftdf_check_sf_drv:
31675 0000AB24 A0[D05D0100]           <1>       mov    al, [SourceFile_Drv]
31676                                  <1>
31677                                  <1> msftdf_check_df_drv:
31678 0000AB29 8A15[505E0100]         <1>       mov    dl, [DestinationFile_Drv]
31679                                  <1>
31680                                  <1> msftdf_compare_sf_df_drv:
31681 0000AB2F 29DB                   <1>       sub    ebx, ebx
31682 0000AB31 8A3D[E6520100]         <1>       mov    bh, [Current_Drv]
31683 0000AB37 38C2                   <1>       cmp    dl, al
31684 0000AB39 7409                   <1>       je     short msftdf_check_sf_df_drv_ok
31685                                  <1>
31686                                  <1> msftdf_not_same_drv:
31687                                  <1>         ; DL = source file's drive number
31688 0000AB3B 88C6                   <1>       mov    dh, al ; destination file's drive number
31689                                  <1>       ; 15/10/2016 (11h -> 21)
31690 0000AB3D B815000000             <1>       mov    eax, 21 ; Not the same drive
31691 0000AB42 F9                     <1>       stc
31692 0000AB43 C3                     <1>       retn
31693                                  <1>
31694                                  <1> msftdf_check_sf_df_drv_ok:
31695 0000AB44 8815[D35E0100]         <1>       mov    [msftdf_sf_df_drv], dl
31696                                  <1>
31697 0000AB4A 29C0                   <1>         sub  eax, eax
31698 0000AB4C 88D4                   <1>       mov    ah, dl
31699 0000AB4E 0500010900             <1>       add    eax, Logical_DOSDisks
31700 0000AB53 A3[D45E0100]           <1>       mov    [msftdf_drv_offset], eax
31701                                  <1>
31702 0000AB58 38FA                   <1>       cmp    dl, bh ; byte [Current_Drv]
31703 0000AB5A 7407                   <1>       je     short msftdf_df_check_directory
31704                                  <1>
31705                                  <1> msftdf_change_drv:
31706 0000AB5C E80FC1FFFF             <1>       call   change_current_drive
31707 0000AB61 726D                   <1>       jc     short msftdf_df_error_retn
31708                                  <1>
31709                                  <1> msftdf_check_destination_file:
31710                                  <1> msftdf_df_check_directory:
31711 0000AB63 BE[515E0100]           <1>       mov    esi, DestinationFile_Directory
31712 0000AB68 803E20                 <1>       cmp    byte [esi], 20h
31713 0000AB6B 760F                   <1>       jna    short msftdf_df_find_1
31714                                  <1>
31715                                  <1> msftdf_df_change_directory:
31716 0000AB6D FE05[D3060100]         <1>       inc    byte [Restore_CDIR]
31717 0000AB73 30E4                   <1>       xor    ah, ah ; CD_COMMAND sign -> 0
31718 0000AB75 E8A1F0FFFF             <1>       call   change_current_directory
31719 0000AB7A 7254                   <1>       jc     short msftdf_df_error_retn
```

```
                                <1>
                                <1> ;msftdf_df_change_prompt_dir_string:
                                <1> ;       call    change_prompt_dir_string
                                <1>
                                <1> msftdf_df_find_1:
0000AB7C BE[925E0100]           <1>         mov     esi, DestinationFile_Name
0000AB81 803E20                 <1>         cmp     byte [esi], 20h
0000AB84 7631                   <1>         jna     short msftdf_df_copy_sf_name
                                <1>
                                <1> msftdf_df_find_2:
0000AB86 6631C0                 <1>         xor     ax, ax ; DestinationFile_AttributesMask -> any/zero
0000AB89 E87DD4FFFF             <1>         call    find_first_file
0000AB8E 0F838D000000           <1>         jnc     msftdf_permission_denied_retn
                                <1>
                                <1> msftdf_df_check_error_code:
                                <1>         ;cmp    eax, 2 ; File not found error
0000AB94 3C02                   <1>         cmp     al, 2
0000AB96 7537                   <1>         jne     short msftdf_df_stc_retn
                                <1>
                                <1> msftdf_df_check_fname:
                                <1>         ; 15/10/2016
0000AB98 BE[925E0100]           <1>         mov     esi, DestinationFile_Name ; *
0000AB9D E829D8FFFF             <1>         call    check_filename
0000ABA2 7307                   <1>         jnc     short msftdf_convert_df_direntry_name
                                <1>         ; invalid file name chars !
0000ABA4 B81A000000             <1>         mov     eax, ERR_INV_FILE_NAME  ; 26
0000ABA9 EB24                   <1>         jmp     short msftdf_df_stc_retn
                                <1>
                                <1> msftdf_convert_df_direntry_name:
                                <1>         ; mov  esi, DestinationFile_Name ; *
0000ABAB BF[A25E0100]           <1>         mov     edi, DestinationFile_DirEntry
0000ABB0 E8B1F5FFFF             <1>         call    convert_file_name
0000ABB5 EB1A                   <1>         jmp     short msftdf_restore_current_dir_1
                                <1>
                                <1> msftdf_df_copy_sf_name:
0000ABB7 89F7                   <1>         mov     edi, esi
0000ABB9 57                     <1>         push    edi
0000ABBA BE[125E0100]           <1>         mov     esi, SourceFile_Name
0000ABBF B90C000000             <1>         mov     ecx, 12
                                <1> msftdf_df_copy_sf_name_loop:
0000ABC4 AC                     <1>         lodsb
0000ABC5 AA                     <1>         stosb
0000ABC6 08C0                   <1>         or      al, al
0000ABC8 7402                   <1>         jz      short msftdf_df_copy_sf_name_ok
0000ABCA E2F8                   <1>         loop    msftdf_df_copy_sf_name_loop
                                <1> msftdf_df_copy_sf_name_ok:
0000ABCC 5E                     <1>         pop     esi
0000ABCD EBB7                   <1>         jmp     short msftdf_df_find_2
                                <1>
                                <1> msftdf_df_stc_retn:
0000ABCF F9                     <1>         stc
                                <1> msftdf_restore_cdir_failed:
                                <1> msftdf_df_error_retn:
0000ABD0 C3                     <1>         retn
                                <1>
                                <1> msftdf_restore_current_dir_1:
0000ABD1 803D[D3060100]00       <1>         cmp     byte [Restore_CDIR], 0
0000ABD8 760D                   <1>         jna     short msftdf_sf_check_directory
0000ABDA 8B35[D45E0100]         <1>         mov     esi, [msftdf_drv_offset]
0000ABE0 E842C1FFFF             <1>         call    restore_current_directory
0000ABE5 72E9                   <1>         jc      short msftdf_restore_cdir_failed
                                <1>
                                <1> msftdf_sf_check_directory:
0000ABE7 BE[D15D0100]           <1>         mov     esi, SourceFile_Directory
0000ABEC 803E20                 <1>         cmp     byte [esi], 20h
0000ABEF 760F                   <1>         jna     short msftdf_sf_find
                                <1> msftdf_sf_change_directory:
0000ABF1 FE05[D3060100]         <1>         inc     byte [Restore_CDIR]
0000ABF7 30E4                   <1>         xor     ah, ah ; CD_COMMAND sign -> 0
0000ABF9 E81DF0FFFF             <1>         call    change_current_directory
0000ABFE 7227                   <1>         jc      short msftdf_return
                                <1>
                                <1> ;msftdf_sf_change_prompt_dir_string:
                                <1> ;       call    change_prompt_dir_string
                                <1>
                                <1> msftdf_sf_find:
0000AC00 BE[125E0100]           <1>         mov     esi, SourceFile_Name  ; Offset 66
0000AC05 66B80018               <1>         mov     ax, 1800h ; Only files
0000AC09 E8FDD3FFFF             <1>         call    find_first_file
0000AC0E 7217                   <1>         jc      short msftdf_return
                                <1>
                                <1> msftdf_sf_ambgfn_check:
0000AC10 6609D2                 <1>         or      dx, dx ; Ambiguous filename chars used sign (DX>0)
0000AC13 7407                   <1>         jz      short msftdf_sf_found
                                <1>
                                <1> msftdf_ambiguous_file_name_error:
0000AC15 B802000000             <1>         mov     eax, 2 ; File not found error
0000AC1A F9                     <1>         stc
0000AC1B C3                     <1>         retn
                                <1>
                                <1> msftdf_sf_found:
0000AC1C 80E31F                 <1>         and     bl, 1Fh ; Attributes, D-V-S-H-R
0000AC1F 7416                   <1>         jz      short msftdf_save_sf_structure
                                <1>
                                <1> msftdf_permission_denied_retn:
0000AC21 B805000000             <1>         mov     eax, 05h ; Access (Permission) denied !
0000AC26 F9                     <1>         stc
                                <1> msftdf_rest_cdir_err_retn:
                                <1> msftdf_return:
0000AC27 C3                     <1>         retn
                                <1>
                                <1> msftdf_phase_1_return:
0000AC28 31C0                   <1>         xor     eax, eax
```

```
31823 0000AC2A A2[D25E0100]       <1>       mov    [move_cmd_phase], al ; 0
31824 0000AC2F FEC0               <1>       inc    al ; mov al, 1
31825 0000AC31 BB[7EAC0000]       <1>       mov    ebx, msftdf_df2_check_directory
31826                             <1>       ;mov   edx, 0FFFFFFFFh
31827 0000AC36 C3                 <1>       retn
31828                             <1>
31829                             <1> msftdf_save_sf_structure:
31830 0000AC37 BE[DC5C0100]       <1>       mov    esi, FindFile_DirEntry
31831 0000AC3C BF[225E0100]       <1>       mov    edi, SourceFile_DirEntry
31832 0000AC41 B908000000         <1>       mov    ecx, 8
31833 0000AC46 F3A5               <1>       rep    movsd
31834                             <1>
31835                             <1> msftdf_df_copy_sf_parameters:
31836 0000AC48 BE0B000000         <1>       mov    esi, 11
31837 0000AC4D 89F7               <1>       mov    edi, esi
31838 0000AC4F 81C6[225E0100]     <1>       add    esi, SourceFile_DirEntry
31839 0000AC55 81C7[A25E0100]     <1>       add    edi, DestinationFile_DirEntry
31840                             <1>       ;mov   ecx, 21
31841 0000AC5B B115               <1>       mov    cl, 21
31842 0000AC5D F3A4               <1>       rep    movsb
31843                             <1>
31844                             <1> msftdf_restore_current_dir_2:
31845 0000AC5F 803D[D3060100]00   <1>       cmp    byte [Restore_CDIR], 0
31846 0000AC66 760D               <1>       jna    short msftdf_df2_check_move_cmd_phase
31847 0000AC68 8B35[D45E0100]     <1>       mov    esi, [msftdf_drv_offset]
31848 0000AC6E E8B4C0FFFF         <1>       call   restore_current_directory
31849 0000AC73 72B2               <1>       jc     short msftdf_rest_cdir_err_retn
31850                             <1>
31851                             <1> msftdf_df2_check_move_cmd_phase:
31852 0000AC75 803D[D25E0100]01   <1>       cmp    byte [move_cmd_phase], 1
31853 0000AC7C 74AA               <1>       je     short msftdf_phase_1_return
31854                             <1>
31855                             <1> msftdf_df2_check_directory:
31856 0000AC7E BE[515E0100]       <1>       mov    esi, DestinationFile_Directory
31857 0000AC83 803E20             <1>       cmp    byte [esi], 20h
31858 0000AC86 760F               <1>       jna    short msftdf_make_dfde_locate_ffe_on_directory
31859                             <1> msftdf_df2_change_directory:
31860 0000AC88 FE05[D3060100]     <1>       inc    byte [Restore_CDIR]
31861 0000AC8E 30E4               <1>       xor    ah, ah ; CD_COMMAND sign -> 0
31862 0000AC90 E886EFFFFF         <1>       call   change_current_directory
31863 0000AC95 7290               <1>       jc     short msftdf_return
31864                             <1>
31865                             <1> ;msftdf_df2_change_prompt_dir_string:
31866                             <1> ;     call   change_prompt_dir_string
31867                             <1>
31868                             <1> msftdf_make_dfde_locate_ffe_on_directory:
31869                             <1>       ; Current directory fcluster <> Directory buffer cluster
31870                             <1>       ; Current directory will be reloaded by
31871                             <1>       ; 'locate_current_dir_file' procedure
31872                             <1>       ;
31873                             <1>       ;xor   ax, ax
31874 0000AC97 31C0               <1>       xor    eax, eax
31875 0000AC99 89C1               <1>       mov    ecx, eax
31876 0000AC9B 6649               <1>       dec    cx ; FFFFh
31877                             <1>              ; CX = FFFFh -> find first deleted or free entry
31878                             <1>              ; ESI would be ASCIIZ filename address if the call
31879                             <1>              ; would not be for first free or deleted dir entry
31880 0000AC9D E8CDF1FFFF         <1>       call   locate_current_dir_file
31881 0000ACA2 733F               <1>       jnc    msftdf_make_dfde_set_ff_dir_entry
31882                             <1>
31883                             <1>       ;cmp   eax, 2
31884 0000ACA4 3C02               <1>         cmp  al, 2
31885 0000ACA6 7537               <1>       jne    short msftdf_error_retn
31886                             <1>
31887                             <1> msftdf_add_new_dir_entry_check_fs:
31888 0000ACA8 8B35[D45E0100]     <1>       mov    esi, [msftdf_drv_offset]
31889 0000ACAE A1[155B0100]       <1>       mov    eax, [DirBuff_Cluster]
31890 0000ACB3 807E0300           <1>       cmp    byte [esi+LD_FATType], 0
31891 0000ACB7 7711               <1>       ja     short msftdf_add_new_subdir_cluster
31892                             <1>
31893                             <1> msftdf_add_new_fs_subdir_section:
31894                             <1>       ;CL=0, CH=E5h --> deleted entry, CH=0 --> free entry
31895                             <1>         ;xor cx, cx
31896 0000ACB9 30ED               <1>       xor    ch, ch ; cx = 0 --> add a new subdir section
31897 0000ACBB E8830C0000         <1>       call   add_new_fs_section
31898 0000ACC0 721E               <1>         jc   short msftdf_dsfde_error_retn
31899                             <1>       ;mov   [createfile_LastDirCluster], eax
31900                             <1>
31901 0000ACC2 E8A30E0000         <1>       call   load_FS_sub_directory
31902                             <1>       ;mov   ebx, Directory_Buffer
31903 0000ACC7 7318               <1>       jnc    short msftdf_add_new_fs_subdir_section_ok
31904 0000ACC9 C3                 <1>       retn
31905                             <1>
31906                             <1> msftdf_add_new_subdir_cluster:
31907 0000ACCA E881150000         <1>       call   add_new_cluster
31908 0000ACCF 720F               <1>       jc     short msftdf_dsfde_error_retn
31909                             <1>
31910                             <1>       ;mov   [createfile_LastDirCluster], eax
31911                             <1>
31912 0000ACD1 E8570E0000         <1>       call   load_FAT_sub_directory
31913 0000ACD6 7309               <1>       jnc    short msftdf_add_new_subdir_cluster_ok
31914                             <1>       ; EBX = Directory buffer address
31915                             <1>
31916                             <1> msftdf_ansdc_update_parent_dir_lmdt:
31917                             <1> msftdf_make_dfde_err_upd_pdir_lmdt:
31918 0000ACD8 50                 <1>       push   eax
31919 0000ACD9 E852FAFFFF         <1>       call   update_parent_dir_lmdt
31920 0000ACDE 58                 <1>       pop    eax
31921                             <1>
31922                             <1> msftdf_error_retn:
31923 0000ACDF F9                 <1>       stc
31924                             <1> msftdf_dsfde_restore_cdir_failed:
31925                             <1> msftdf_dsfde_error_retn:
```

```
31926 0000ACE0 C3                  <1>       retn
31927                              <1>
31928                              <1> msftdf_add_new_fs_subdir_section_ok:
31929                              <1> msftdf_add_new_subdir_cluster_ok:
31930 0000ACE1 89DF                <1>       mov    edi, ebx ; Directory buffer address
31931                              <1>
31932                              <1> msftdf_make_dfde_set_ff_dir_entry:
31933 0000ACE3 8B15[E0520100]      <1>       mov    edx, [Current_Dir_FCluster]
31934 0000ACE9 8915[385F0100]      <1>       mov    [createfile_FFCluster], edx
31935                              <1>       ; EDI = Directory entry offset
31936 0000ACEF BE[A25E0100]        <1>       mov    esi, DestinationFile_DirEntry
31937 0000ACF4 B908000000          <1>       mov    ecx, 8
31938 0000ACF9 F3A5                <1>       rep    movsd
31939                              <1>
31940 0000ACFB C605[105B0100]02    <1>       mov    byte [DirBuff_ValidData], 2
31941 0000AD02 E88EF9FFFF          <1>       call   save_directory_buffer
31942 0000AD07 72CF                <1>       jc     short msftdf_make_dfde_err_upd_pdir_lmdt
31943                              <1>
31944                              <1> msftdf_make_dfde_update_pdir_lmdt:
31945 0000AD09 E822FAFFFF          <1>       call   update_parent_dir_lmdt
31946                              <1>
31947                              <1> msftdf_dsfde_restore_current_dir_1:
31948 0000AD0E 803D[D3060100]00    <1>       cmp    byte [Restore_CDIR], 0
31949 0000AD15 760D                <1>       jna    short msftdf_dsfde_check_directory
31950 0000AD17 8B35[D45E0100]      <1>       mov    esi, [msftdf_drv_offset]
31951 0000AD1D E805C0FFFF          <1>       call   restore_current_directory
31952 0000AD22 72BC                <1>       jc     short msftdf_dsfde_restore_cdir_failed
31953                              <1>
31954                              <1> msftdf_dsfde_check_directory:
31955 0000AD24 BE[D15D0100]        <1>       mov    esi, SourceFile_Directory
31956 0000AD29 803E20              <1>       cmp    byte [esi], 20h
31957 0000AD2C 760F                <1>       jna    short msftdf_dsfde_find_file
31958                              <1>
31959                              <1> msftdf_dsfde_change_directory:
31960 0000AD2E FE05[D3060100]      <1>       inc    byte [Restore_CDIR]
31961 0000AD34 28E4                <1>       sub    ah, ah ; CD_COMMAND sign -> 0
31962 0000AD36 E8E0EEFFFF          <1>       call   change_current_directory
31963 0000AD3B 72A3                <1>       jc     short msftdf_dsfde_error_retn
31964                              <1>
31965                              <1> ;msftdf_dsfde_sf_change_prompt_dir_string:
31966                              <1> ;     call   change_prompt_dir_string
31967                              <1>
31968                              <1> msftdf_dsfde_find_file:
31969 0000AD3D BE[125E0100]        <1>       mov    esi, SourceFile_Name  ; Offset 66
31970 0000AD42 668B460E            <1>       mov    ax, [esi+14] ; 80 -> SourceFile_AttributesMask
31971 0000AD46 E8C0D2FFFF          <1>       call   find_first_file
31972 0000AD4B 7293                <1>       jc     short msftdf_dsfde_error_retn
31973                              <1>
31974                              <1> msftdf_dsfde_delete_direntry:
31975 0000AD4D 8B35[D45E0100]      <1>       mov    esi, [msftdf_drv_offset]
31976                              <1>
31977 0000AD53 807E0300            <1>       cmp    byte [esi+LD_FATType], 0
31978 0000AD57 770A                <1>       ja     short msftdf_delete_FAT_direntry
31979                              <1>
31980 0000AD59 30DB                <1>       xor    bl, bl
31981                              <1>       ; BL = 0 -> File
31982                              <1>       ; EDI -> Directory buffer entry offset/address
31983 0000AD5B E8E40B0000          <1>       call   delete_fs_directory_entry
31984 0000AD60 7315                <1>       jnc    short msftdf_dsfde_restore_current_dir_2
31985 0000AD62 C3                  <1>       retn
31986                              <1>
31987                              <1> msftdf_delete_FAT_direntry:
31988 0000AD63 8A1D[D95C0100]      <1>       mov    bl, [FindFile_LongNameEntryLength]
31989 0000AD69 668B0D[045D0100]    <1>       mov    cx, [FindFile_DirEntryNumber]
31990                              <1>       ; ESI = Logical DOS drive description table address
31991                              <1>       ; EDI = Directory buffer entry offset/address
31992 0000AD70 E89AFCFFFF          <1>       call   delete_directory_entry
31993 0000AD75 721C                <1>       jc     short msftdf_retn
31994                              <1>
31995                              <1> msftdf_dsfde_restore_current_dir_2:
31996 0000AD77 803D[D3060100]00    <1>       cmp    byte [Restore_CDIR], 0
31997 0000AD7E 7607                <1>       jna    short msftdf_new_dir_fcluster_retn
31998                              <1>       ;mov   esi, [msftdf_drv_offset]
31999 0000AD80 E8A2BFFFFF          <1>       call   restore_current_directory
32000 0000AD85 720C                <1>       jc     short msftdf_retn
32001                              <1>
32002                              <1> msftdf_new_dir_fcluster_retn:
32003 0000AD87 31C9                <1>       xor    ecx, ecx
32004 0000AD89 A1[385F0100]        <1>       mov    eax, [createfile_FFCluster]
32005 0000AD8E BB[505E0100]        <1>       mov    ebx, DestinationFile_Drv
32006                              <1>
32007                              <1> msftdf_retn:
32008 0000AD93 C3                  <1>       retn
32009                              <1>
32010                              <1>
32011                              <1> copy_source_file_to_destination_file:
32012                              <1>       ; 17/10/2016
32013                              <1>       ; 16/10/2016
32014                              <1>       ; 15/10/2016
32015                              <1>       ; 30/03/2016, 31/03/2016
32016                              <1>       ; 24/03/2016, 25/03/2016, 28/03/2016
32017                              <1>       ; 21/03/2016, 22/03/2016, 23/03/2016
32018                              <1>       ; 16/03/2016, 17/03/2016, 18/03/2016
32019                              <1>       ; 15/03/2016 (TRDOS 386 = TRDOS v2.0)
32020                              <1>       ; 02/09/2011 (FILE.ASM 'copy_source_file_to_destination_file')
32021                              <1>       ; 01/08/2010 - 18/05/2011
32022                              <1>       ;
32023                              <1>       ;   Command Interpreter phase 1 enter ->
32024                              <1>       ;           AL = 1 -> Caller is command interpreter
32025                              <1>       ;           AL = 2 -> The second call, re-enter/continue
32026                              <1>       ;   Phase 1 -> Check source file
32027                              <1>       ;              'found' is required
32028                              <1>       ;   Phase 2 -> Check destination file,
```

```
32029                                  <1>       ;             save 'found' or 'not found' status
32030                                  <1>       ;             'permission denied' error will be return
32031                                  <1>       ;             if attributes have not for ordinary file
32032                                  <1>       ;             without readonly attribute
32033                                  <1>       ;   Command Interpreter phase 1 return ->
32034                                  <1>       ;             DH = Source file attributes
32035                                  <1>       ;             DL = Destination file found status
32036                                  <1>       ;             EAX = 0
32037                                  <1>       ;   Command Interpreter phase 2 enter ->
32038                                  <1>       ;             AL = 2 -> Continue from the last position
32039                                  <1>       ;             AH =
32040                                  <1>       ;   Phase 3 -> Load source file or use read/write cluster method
32041                                  <1>       ;   Phase 4 -> Create destination file if it is not found
32042                                  <1>       ;   Phase 5 -> Open destination file
32043                                  <1>       ;   Phase 6 -> Read from source and write to destination
32044                                  <1>       ;   Phase 7 -> Unload source file, if it is loaded at memory
32045                                  <1>       ;       cf = 1 causes to return before the phase 7
32046                                  <1>       ;             but loaded file will be unloaded
32047                                  <1>       ;            (allocated memory block will be deallocated)
32048                                  <1>       ;
32049                                  <1>       ; INPUT ->
32050                                  <1>       ;      ESI = Source File Pathname (Asciiz)
32051                                  <1>       ;       EDI = Destination File Pathname (Asciiz)
32052                                  <1>       ;       AL = 0 --> Interrupt (System call)
32053                                  <1>       ;       AL > 0 --> Command Interpreter (Question)
32054                                  <1>       ;       AL = 1 --> Question Phase
32055                                  <1>       ;       AL = 2 --> Progress Phase
32056                                  <1>       ;
32057                                  <1>       ; OUTPUT ->
32058                                  <1>       ;     cf = 0 -> OK
32059                                  <1>       ;     EAX = Destination file first cluster
32060                                  <1>       ;
32061                                  <1>       ;       CL > 0 if there is file reading error before EOF
32062                                  <1>       ;            (incomplete copy)
32063                                  <1>       ;       CH > 0 if file is (full) loaded at memory
32064                                  <1>       ;
32065                                  <1>       ;     cf = 1 -> Error code in AL (EAX)
32066                                  <1>       ;
32067                                  <1>       ; (EBX, ECX, ESI, EDI register contents will be changed)
32068                                  <1>
32069                                  <1>
32070 0000AD94 3C02                   <1>       cmp   al, 2
32071 0000AD96 0F845A020000           <1>       je    csftdf2_check_cdrv
32072                                  <1>
32073                                  <1> ; Phase 1
32074                                  <1>
32075 0000AD9C A2[F85E0100]           <1>       mov   byte [copy_cmd_phase], al
32076                                  <1>
32077 0000ADA1 57                     <1>       push  edi ; *
32078                                  <1>
32079                                  <1> csftdf_parse_sf_path:
32080 0000ADA2 BF[D05D0100]           <1>       mov   edi, SourceFile_Drv
32081 0000ADA7 E885F4FFFF             <1>       call  parse_path_name
32082 0000ADAC 721C                   <1>       jc    short csftdf_parse_sf_path_failed
32083                                  <1>
32084                                  <1> csftdf_parse_df_path:
32085 0000ADAE 5E                     <1>       pop   esi ; * (pushed edi)
32086                                  <1>
32087                                  <1> csftdf_sf_check_filename_exists:
32088 0000ADAF 803D[125E0100]21       <1>       cmp   byte [SourceFile_Name], 21h
32089 0000ADB6 7215                   <1>       jb    short csftdf_sf_file_not_found_error
32090                                  <1>
32091 0000ADB8 BF[505E0100]           <1>       mov   edi, DestinationFile_Drv
32092 0000ADBD E86FF4FFFF             <1>       call  parse_path_name
32093 0000ADC2 7310                   <1>       jnc   short csftdf_check_sf_cdrv
32094                                  <1>
32095 0000ADC4 3C01                   <1>       cmp   al, 1 ; File or directory name is not existing
32096 0000ADC6 760C                   <1>       jna   short csftdf_check_sf_cdrv
32097                                  <1>
32098                                  <1> csftdf_parse_df_path_failed:
32099 0000ADC8 F9                     <1>       stc
32100                                  <1> csftdf_sf_error_retn:
32101 0000ADC9 C3                     <1>       retn
32102                                  <1>
32103                                  <1> csftdf_parse_sf_path_failed:
32104 0000ADCA 5F                     <1>       pop   edi ; *
32105 0000ADCB EBFC                   <1>       jmp   short csftdf_sf_error_retn
32106                                  <1>
32107                                  <1> csftdf_sf_file_not_found_error:
32108 0000ADCD B802000000             <1>       mov   eax, 2 ; File not found
32109 0000ADD2 EBF5                   <1>       jmp   short csftdf_sf_error_retn
32110                                  <1>
32111                                  <1> csftdf_check_sf_cdrv:
32112 0000ADD4 8A3D[E6520100]         <1>       mov   bh, [Current_Drv]
32113                                  <1>
32114 0000ADDA 883D[FB5E0100]         <1>       mov   [csftdf_cdrv], bh ; 23/03/2016
32115                                  <1>
32116 0000ADE0 8A15[D05D0100]         <1>       mov   dl, [SourceFile_Drv]
32117 0000ADE6 38FA                   <1>       cmp   dl, bh ; byte [Current_Drv]
32118 0000ADE8 7407                   <1>       je    short csftdf_sf_check_directory
32119                                  <1>
32120 0000ADEA E881BEFFFF             <1>       call  change_current_drive
32121 0000ADEF 72D8                   <1>       jc    short csftdf_sf_error_retn
32122                                  <1>
32123                                  <1> csftdf_sf_check_directory:
32124 0000ADF1 BE[D15D0100]           <1>       mov   esi, SourceFile_Directory
32125 0000ADF6 803E20                 <1>       cmp   byte [esi], 20h
32126 0000ADF9 760F                   <1>       jna   short csftdf_find_sf
32127                                  <1>
32128                                  <1> csftdf_sf_change_directory:
32129 0000ADFB FE05[D3060100]         <1>       inc   byte [Restore_CDIR]
32130 0000AE01 30E4                   <1>       xor   ah, ah ; CD_COMMAND sign -> 0
32131 0000AE03 E813EEFFFF             <1>       call  change_current_directory
```

```
32132 0000AE08 72BF                <1>       jc     short csftdf_sf_error_retn
32133                              <1>
32134                              <1> ;csftdf_sf_change_prompt_dir_string:
32135                              <1> ;     call   change_prompt_dir_string
32136                              <1>
32137                              <1> csftdf_find_sf:
32138 0000AE0A BE[125E0100]        <1>       mov    esi, SourceFile_Name
32139 0000AE0F 66B80018            <1>       mov    ax, 1800h ; Except volume label and dirs
32140 0000AE13 E8F3D1FFFF          <1>       call   find_first_file
32141 0000AE18 72AF                <1>       jc     short csftdf_sf_error_retn
32142                              <1>
32143                              <1> csftdf_sf_ambgfn_check:
32144 0000AE1A 6621D2              <1>       and    dx, dx ; Ambiguous filename chars used sign (DX>0)
32145 0000AE1D 7407                <1>       jz     short csftdf_sf_found
32146                              <1>
32147                              <1> csftdf_ambiguous_file_name_error:
32148 0000AE1F B802000000          <1>       mov    eax, 2 ; File not found error
32149 0000AE24 F9                  <1>       stc
32150 0000AE25 C3                  <1>       retn
32151                              <1>
32152                              <1> csftdf_sf_found:
32153 0000AE26 A3[FC5E0100]        <1>       mov    [csftdf_filesize], eax
32154                              <1>
32155 0000AE2B 09C0                <1>       or     eax, eax
32156 0000AE2D 7507                <1>       jnz    short csftdf_set_source_file_direntry
32157                              <1>
32158                              <1> csftdf_sf_file_size_zero:
32159 0000AE2F B814000000          <1>       mov    eax, 20 ; TRDOS zero length (file size) error
32160 0000AE34 F9                  <1>       stc
32161 0000AE35 C3                  <1>       retn
32162                              <1>
32163                              <1> csftdf_set_source_file_direntry:
32164 0000AE36 BE[DC5C0100]        <1>       mov    esi, FindFile_DirEntry
32165 0000AE3B BF[225E0100]        <1>       mov    edi, SourceFile_DirEntry
32166 0000AE40 B908000000          <1>       mov    ecx, 8
32167 0000AE45 F3A5                <1>       rep    movsd
32168                              <1>
32169                              <1> csftdf_sf_restore_cdrv:
32170                              <1>       ; 22/03/2016
32171 0000AE47 8A15[FB5E0100]      <1>       mov    dl, [csftdf_cdrv]
32172 0000AE4D 3A15[E6520100]      <1>       cmp    dl, [Current_Drv]
32173 0000AE53 7407                <1>       je     short csftdf_sf_restore_cdir
32174 0000AE55 E816BEFFFF          <1>       call   change_current_drive
32175 0000AE5A 724F                <1>       jc     short csftdf_df_error_retn ; 30/03/2016
32176                              <1>
32177                              <1> csftdf_sf_restore_cdir:
32178 0000AE5C 803D[D3060100]00    <1>       cmp    byte [Restore_CDIR], 0
32179 0000AE63 7612                <1>       jna    short csftdf_df_check_filename_exists
32180 0000AE65 29C0                <1>       sub    eax, eax
32181 0000AE67 BE00010900          <1>       mov    esi, Logical_DOSDisks
32182 0000AE6C 88D4                <1>       mov    ah, dl ; byte [csftdf_cdrv]
32183 0000AE6E 01C6                <1>       add    esi, eax
32184 0000AE70 E8B2BEFFFF          <1>       call   restore_current_directory
32185 0000AE75 7234                <1>       jc     short csftdf_df_error_retn
32186                              <1>
32187                              <1> csftdf_df_check_filename_exists:
32188 0000AE77 803D[925E0100]20    <1>       cmp    byte [DestinationFile_Name], 20h
32189 0000AE7E 7716                <1>       ja     short csftdf_check_df_cdrv
32190                              <1>
32191                              <1> csftdf_copy_sf_name:
32192 0000AE80 BF[925E0100]        <1>       mov    edi, DestinationFile_Name
32193 0000AE85 BE[125E0100]        <1>       mov    esi, SourceFile_Name
32194 0000AE8A B10C                <1>       mov    cl, 12
32195                              <1>
32196                              <1> csftdf_df_copy_sf_name_loop:
32197 0000AE8C AC                  <1>       lodsb
32198 0000AE8D AA                  <1>       stosb
32199 0000AE8E 08C0                <1>       or     al, al
32200 0000AE90 7404                <1>       jz     short csftdf_check_df_cdrv
32201 0000AE92 FEC9                <1>       dec    cl
32202 0000AE94 75F6                <1>       jnz    csftdf_df_copy_sf_name_loop
32203                              <1>
32204                              <1> csftdf_check_df_cdrv:
32205 0000AE96 8A15[505E0100]      <1>       mov    dl, [DestinationFile_Drv]
32206 0000AE9C 3A15[E6520100]      <1>       cmp    dl, [Current_Drv]
32207 0000AEA2 7408                <1>       je     short csftdf_df_check_directory
32208                              <1>
32209 0000AEA4 E8C7BDFFFF          <1>       call   change_current_drive
32210 0000AEA9 7301                <1>       jnc    short csftdf_df_check_directory
32211                              <1>
32212                              <1> csftdf_df_error_retn:
32213 0000AEAB C3                  <1>       retn
32214                              <1>
32215                              <1> csftdf_df_check_directory:
32216 0000AEAC BE[515E0100]        <1>       mov    esi, DestinationFile_Directory
32217 0000AEB1 803E20              <1>         cmp     byte [esi], 20h
32218 0000AEB4 760F                <1>       jna    short csftdf_find_df
32219                              <1>
32220                              <1> csftdf_df_change_directory:
32221 0000AEB6 FE05[D3060100]      <1>       inc    byte [Restore_CDIR]
32222 0000AEBC 28E4                <1>       sub    ah, ah ; CD_COMMAND sign -> 0
32223 0000AEBE E858EDFFFF          <1>       call   change_current_directory
32224 0000AEC3 72E6                <1>       jc     short csftdf_df_error_retn
32225                              <1>
32226                              <1> ;csftdf_df_change_prompt_dir_string:
32227                              <1> ;     call   change_prompt_dir_string
32228                              <1>
32229                              <1> csftdf_find_df:
32230                              <1>       ; 23/03/2016
32231 0000AEC5 29DB                <1>       sub    ebx, ebx
32232 0000AEC7 8A3D[505E0100]      <1>       mov    bh,  [DestinationFile_Drv]
32233 0000AECD 81C300010900        <1>       add    ebx, Logical_DOSDisks
32234 0000AED3 891D[285F0100]      <1>       mov    [csftdf_df_drv_dt], ebx
```

```
32235                                  <1>
32236 0000AED9 BE[925E0100]            <1>       mov    esi, DestinationFile_Name
32237 0000AEDE 6631C0                  <1>       xor    ax, ax
32238                                  <1>              ; DestinationFile_AttributesMask -> any/zero
32239 0000AEE1 E825D1FFFF              <1>       call   find_first_file
32240 0000AEE6 7218                    <1>       jc     short csftdf_df_check_error_code
32241                                  <1>
32242                                  <1> csftdf_df_ambgfn_check:
32243 0000AEE8 6609D2                  <1>       or     dx, dx ; Ambiguous filename chars used sign (DX>0)
32244 0000AEEB 752A                    <1>       jnz    short csftdf_df_error_inv_fname
32245                                  <1>
32246                                  <1> csftdf_df_found:
32247 0000AEED C605[FA5E0100]01        <1>       mov    byte [DestinationFileFound], 1
32248                                  <1>       ; 17/10/2016 (cl -> bl)
32249 0000AEF4 80E31F                  <1>       and    bl, 1Fh ; Attributes, D-V-S-H-R
32250 0000AEF7 745F                    <1>       jz     short csftdf_df_save_first_cluster
32251                                  <1>
32252                                  <1> csftdf_df_permission_denied_retn:
32253 0000AEF9 B805000000              <1>       mov    eax, 05h ; Access/Permisson denied.
32254                                  <1> csftdf_df_error_stc_retn:
32255 0000AEFE F9                      <1>       stc
32256 0000AEFF C3                      <1>       retn
32257                                  <1>
32258                                  <1> csftdf_df_check_error_code:
32259                                  <1>       ;cmp   eax, 2
32260 0000AF00 3C02                    <1>       cmp    al, 2
32261 0000AF02 75FA                    <1>       jne    short csftdf_df_error_stc_retn
32262                                  <1>
32263 0000AF04 C605[FA5E0100]00        <1>       mov    byte [DestinationFileFound], 0
32264                                  <1>
32265                                  <1>       ; 15/10/2016
32266 0000AF0B BE[CC5C0100]            <1>       mov    esi, FindFile_Name ; *
32267 0000AF10 E8B6D4FFFF              <1>       call   check_filename
32268 0000AF15 7307                    <1>       jnc    short csftdf_df_valid_fname
32269                                  <1> csftdf_df_error_inv_fname: ; 'invalid file name !'
32270 0000AF17 B81A000000              <1>       mov    eax, ERR_INV_FILE_NAME  ; 26
32271 0000AF1C F9                      <1>       stc
32272 0000AF1D C3                      <1>       retn
32273                                  <1>
32274                                  <1> csftdf_df_valid_fname:
32275                                  <1>       ; 21/03/2016
32276                                  <1>       ; (Capitalized file name)
32277                                  <1>       ;mov   esi, FindFile_Name ; * ; 15/10/2016
32278 0000AF1E BF[925E0100]            <1>       mov    edi, DestinationFile_Name
32279 0000AF23 A5                      <1>       movsd
32280 0000AF24 A5                      <1>       movsd
32281 0000AF25 A5                      <1>       movsd
32282                                  <1>       ;movsb
32283                                  <1>
32284                                  <1> csftdf_check_disk_free_size_0:
32285 0000AF26 A1[3E5E0100]            <1>       mov    eax, [SourceFile_DirEntry+DirEntry_FileSize]
32286                                  <1>
32287                                  <1> csftdf_check_disk_free_size_1:
32288                                  <1>       ;sub   ebx, ebx
32289                                  <1>       ;mov   esi, Logical_DOSDisks
32290                                  <1>       ;mov   bh,  [DestinationFile_Drv]
32291                                  <1>       ;add   esi, ebx
32292                                  <1>
32293 0000AF2B 8B35[285F0100]          <1>       mov    esi, [csftdf_df_drv_dt] ; 23/03/2016
32294                                  <1>
32295 0000AF31 0FB74E11                <1>       movzx  ecx, word [esi+LD_BPB+BytesPerSec] ; 17, LD_BPB + 0Bh
32296 0000AF35 01C8                    <1>       add    eax, ecx
32297 0000AF37 48                      <1>       dec    eax  ; file size (additional bytes) + 511 (round up)
32298                                  <1> csftdf_check_disk_free_size_3: ; 16/03/2016
32299 0000AF38 29D2                    <1>       sub    edx, edx
32300 0000AF3A F7F1                    <1>       div    ecx ; bytes per sector
32301                                  <1>
32302                                  <1> csftdf_check_disk_free_size:
32303 0000AF3C 3B4674                  <1>       cmp    eax, [esi+LD_FreeSectors]
32304 0000AF3F 0F8294000000            <1>         jb     csftdf_check_disk_free_size_ok
32305 0000AF45 770A                    <1>       ja     short csftdf_df_insufficient_disk_space
32306                                  <1>
32307 0000AF47 807E0300                <1>       cmp    byte [esi+LD_FATType], 0 ; FS needs FDT sector also.
32308 0000AF4B 0F8788000000            <1>         ja     csftdf_check_disk_free_size_ok
32309                                  <1>
32310                                  <1> csftdf_df_insufficient_disk_space:
32311 0000AF51 B827000000              <1>       mov    eax, 27h ; insufficient disk space
32312 0000AF56 EBA6                    <1>       jmp    short csftdf_df_error_stc_retn
32313                                  <1>
32314                                  <1> csftdf_df_save_first_cluster:
32315                                  <1>       ; ESI = FindFile_DirEntry (for the old destination file)
32316                                  <1>       ; EAX = Old destination file size
32317                                  <1>       ; 24/03/2016
32318                                  <1>       ; EDI = Directory entry address (within Dir Buffer boundaries)
32319 0000AF58 81EF00000800            <1>       sub    edi, Directory_Buffer  ; (<65536)
32320 0000AF5E 66C1EF05                <1>       shr    di, 5 ; Convert entry offset to entry index/number
32321 0000AF62 66893D[CA5E0100]        <1>       mov    [DestinationFile_DirEntryNumber], di ; (<2048)
32322                                  <1>
32323                                  <1> csftdf_df_check_sf_df_fcluster:
32324 0000AF69 668B5614                <1>       mov    dx, [esi+DirEntry_FstClusHI]
32325 0000AF6D C1E210                  <1>       shl    edx, 16
32326 0000AF70 668B561A                <1>       mov    dx, [esi+DirEntry_FstClusLO]
32327 0000AF74 8915[0C5F0100]          <1>       mov    [csftdf_df_cluster], edx
32328                                  <1> csftdf_df_check_sf_df_fcluster_1:
32329 0000AF7A 668B15[365E0100]        <1>       mov    dx, [SourceFile_DirEntry+DirEntry_FstClusHI]
32330 0000AF81 C1E210                  <1>       shl    edx, 16
32331 0000AF84 668B15[3C5E0100]        <1>       mov    dx, [SourceFile_DirEntry+DirEntry_FstClusLO]
32332 0000AF8B 3B15[0C5F0100]          <1>       cmp    edx, [csftdf_df_cluster]
32333 0000AF91 7512                    <1>       jne    short csftdf_df_check_sf_df_fcluster_ok
32334                                  <1> csftdf_df_check_sf_df_drv:
32335 0000AF93 8A15[D05D0100]          <1>       mov    dl, [SourceFile_Drv]
32336 0000AF99 3A15[505E0100]          <1>       cmp    dl, [DestinationFile_Drv]
32337 0000AF9F 7504                    <1>       jne    short csftdf_df_check_sf_df_fcluster_ok
```

```
                                <1>
                                <1>       ; source and destination files are same !
                                <1>       ; (they have same first cluster value on same logical disk)
                                <1>
0000AFA1 31C0                   <1>       xor    eax, eax ; mov eax, 0 -> Bad command or file name !
0000AFA3 F9                     <1>       stc
0000AFA4 C3                     <1>       retn
                                <1>
                                <1> csftdf_df_check_sf_df_fcluster_ok:
                                <1> csftdf_df_move_findfile_struct:
                                <1>       ; mov  esi, FindFile_DirEntry
0000AFA5 BF[A25E0100]           <1>       mov    edi, DestinationFile_DirEntry
0000AFAA B908000000             <1>       mov    ecx, 8
0000AFAF F3A5                   <1>       rep    movsd
                                <1>
                                <1> csftdf_check_disk_free_size_2:
0000AFB1 89C2                   <1>       mov    edx, eax ; Old destination file size
                                <1>
                                <1>       ;mov   eax, [SourceFile_DirEntry+DirEntry_FileSize]
0000AFB3 A1[FC5E0100]           <1>       mov    eax, [csftdf_filesize] ; 23/03/2016
                                <1>
                                <1>       ;;sub  ecx, ecx ; 0
                                <1>       ;mov   esi, Logical_DOSDisks
                                <1>       ;mov   ch,  [DestinationFile_Drv]
                                <1>       ;add   esi, ecx
                                <1>       ;
                                <1>       ;mov   [csftdf_df_drv_dt], esi
                                <1>
0000AFB8 8B35[285F0100]         <1>       mov    esi, [csftdf_df_drv_dt] ; 23/03/2016
                                <1>
0000AFBE 668B4E11               <1>       mov    cx, [esi+LD_BPB+BytesPerSec] ; 17, LD_BPB + 0Bh
0000AFC2 01CA                   <1>       add    edx, ecx ; + 512
0000AFC4 01C8                   <1>       add    eax, ecx ; + 512
0000AFC6 4A                     <1>       dec    edx ; old file size + 511 (round up)
0000AFC7 48                     <1>       dec    eax ; new file size + 511 (round up)
0000AFC8 F7D9                   <1>       neg    ecx ; -512 ; 0FFFFFE00h
0000AFCA 21CA                   <1>       and    edx, ecx ; = old sector count * 512
0000AFCC 21C8                   <1>       and    eax, ecx ; = new sector count * 512
                                <1>
0000AFCE 29D0                   <1>       sub    eax, edx ; new file size - old file size (on disk)
0000AFD0 7607                   <1>       jna    short csftdf_check_disk_free_size_ok
                                <1>
0000AFD2 F7D9                   <1>       neg    ecx ; 512 (bytes per sector) ; 200h
                                <1>       ; check free space for additional sectors
                                <1>       ; eax = number of additional sectors * bytes per sector
                                <1>       ; esi = Logical DOS drive number (of destination disk)
0000AFD4 E95FFFFFFF             <1>         jmp     csftdf_check_disk_free_size_3
                                <1>
                                <1> csftdf_check_disk_free_size_ok:
                                <1>       ; 18/03/2016
                                <1> csftdf_df_check_copy_cmd_phase:
0000AFD9 A0[F85E0100]           <1>       mov    al, [copy_cmd_phase]
0000AFDE 3C01                   <1>       cmp    al, 1
0000AFE0 7514                   <1>       jne    short csftdf2_check_cdrv
                                <1>
0000AFE2 31C0                   <1>       xor    eax, eax
0000AFE4 A2[F85E0100]           <1>       mov    [copy_cmd_phase], al ; 0
                                <1>
0000AFE9 8A15[FA5E0100]         <1>       mov    dl, [DestinationFileFound]
0000AFEF 8A35[2D5E0100]         <1>       mov    dh, [SourceFile_DirEntry+11] ; Attributes
                                <1>
                                <1> csftdf_return:
0000AFF5 C3                     <1>       retn
                                <1>
                                <1> ; Phase 2
                                <1>
                                <1> csftdf2_check_cdrv:
                                <1>       ; 18/03/2016
                                <1>       ; Here, destination drive and directory are ready !
                                <1>       ; (checking/restoring is not needed)
                                <1>       ; (Since at the end of the phase 1)
                                <1>
                                <1> ;     mov    dl, [DestinationFile_Drv]
                                <1> ;     cmp    dl, [Current_Drv]
                                <1> ;     je     short csftdf2_df_check_directory
                                <1> ;
                                <1> ;     call   change_current_drive
                                <1> ;     jc     short csftdf2_read_error
                                <1> ;
                                <1> ;csftdf2_df_check_directory:
                                <1> ;     mov    esi, DestinationFile_Directory
                                <1> ;     cmp    byte [esi], 20h
                                <1> ;     jna    short csftdf2_df_check_found_or_not
                                <1> ;
                                <1> ;csftdf2_df_change_directory:
                                <1> ;     inc    byte [Restore_CDIR]
                                <1> ;     xor    ah, ah ; CD_COMMAND sign -> 0
                                <1> ;     call   change_current_directory
                                <1> ;     jc     short csftdf2_stc_return
                                <1> ;
                                <1> ;;csftdf2_df_change_prompt_dir_string:
                                <1> ;;    call   change_prompt_dir_string
                                <1>
                                <1> csftdf2_df_check_found_or_not:
                                <1>       ; 21/03/2016
0000AFF6 803D[FA5E0100]00       <1>       cmp    byte [DestinationFileFound], 0
0000AFFD 7739                   <1>       ja     short csftdf2_set_sf_percentage
                                <1>
                                <1> csftdf2_create_file:
0000AFFF BE[925E0100]           <1>       mov    esi, DestinationFile_Name
0000B004 A1[FC5E0100]           <1>       mov    eax, [csftdf_filesize]
0000B009 30C9                   <1>       xor    cl, cl ; 0
                                <1>
```

```
32441 0000B00B 31DB                <1>       xor    ebx, ebx ; 0
32442 0000B00D 4B                  <1>       dec    ebx ; 0FFFFFFFFh
32443                              <1>
32444                              <1>       ; INPUT ->
32445                              <1>       ;      EAX -> File Size
32446                              <1>       ;      ESI = ASCIIZ File name
32447                              <1>       ;       CL = File attributes
32448                              <1>       ;      EBX = FFFFFFFFh -> empty file sign for FAT fs
32449                              <1>       ;      EBX <> FFFFFFFFh -> use file size for FAT fs
32450                              <1>       ;
32451                              <1>       ; OUTPUT ->
32452                              <1>       ;      EAX = New file's first cluster
32453                              <1>       ;      ESI = Logical Dos Drv Descr. Table Addr.
32454                              <1>       ;      EBX = CreateFile_Size address
32455                              <1>       ;      ECX = Sectors per cluster (<256)
32456                              <1>       ;      EDX = Directory Entry Index/Number (<65536)
32457                              <1>       ;
32458                              <1>       ;      cf = 1 -> error code in AL (EAX)
32459                              <1>
32460 0000B00E E8EC050000          <1>       call   create_file
32461                              <1>       ;pop   esi
32462 0000B013 0F82A3050000        <1>         jc      csftdf2_rw_error
32463                              <1>
32464                              <1> csftdf2_create_file_OK:
32465 0000B019 A3[0C5F0100]        <1>       mov    [csftdf_df_cluster], eax
32466                              <1>
32467                              <1>       ; 24/03/2016
32468 0000B01E 668915[CA5E0100]    <1>       mov    [DestinationFile_DirEntryNumber], dx
32469                              <1>
32470                              <1>       ; 21/03/2016
32471 0000B025 BE00000800          <1>       mov    esi, Directory_Buffer
32472 0000B02A C1E205              <1>       shl    edx, 5 ; 32 * index number
32473 0000B02D 01D6                <1>       add    esi, edx
32474 0000B02F BF[A25E0100]        <1>       mov    edi, DestinationFile_DirEntry
32475 0000B034 B108                <1>       mov    cl, 8 ; 32 bytes
32476 0000B036 F3A5                <1>       rep    movsd
32477                              <1>
32478                              <1> csftdf2_set_sf_percentage:
32479                              <1>       ; 17/03/2016
32480 0000B038 31C0                <1>       xor    eax, eax
32481 0000B03A A2[205F0100]        <1>       mov    [csftdf_percentage], al ; 0, reset
32482                              <1>
32483 0000B03F A3[185F0100]        <1>       mov    [csftdf_sf_rbytes], eax ; 0, reset
32484 0000B044 A3[1C5F0100]        <1>       mov    [csftdf_df_wbytes], eax ; 0, reset
32485                              <1>
32486 0000B049 8A25[D05D0100]      <1>       mov    ah, [SourceFile_Drv]
32487 0000B04F BE00010900          <1>       mov    esi, Logical_DOSDisks
32488 0000B054 01C6                <1>       add    esi, eax
32489                              <1>
32490 0000B056 8935[245F0100]      <1>       mov    [csftdf_sf_drv_dt], esi ; 23/03/2016
32491                              <1>
32492 0000B05C 668B15[365E0100]    <1>       mov    dx, [SourceFile_DirEntry+DirEntry_FstClusHI]
32493 0000B063 C1E210              <1>       shl    edx, 16
32494 0000B066 668B15[3C5E0100]    <1>       mov    dx, [SourceFile_DirEntry+DirEntry_FstClusLO]
32495 0000B06D 8915[085F0100]      <1>       mov    [csftdf_sf_cluster], edx
32496                              <1>
32497                              <1>       ; 16/03/2016
32498                              <1>       ; Note: Singlix FS boot sector parameters (for cluster
32499                              <1>       ;      related calculations) has same offset
32500                              <1>       ;      values from LD_BPB as in FAT file system.
32501                              <1>       ;      [esi+LD_BPB+SecPerClust] is 1 for Singlix FS.
32502                              <1>       ;
32503 0000B073 0FB64E13            <1>       movzx  ecx, byte [esi+LD_BPB+SecPerClust]
32504 0000B077 880D[4E5E0100]      <1>       mov    [SourceFile_SecPerClust], cl
32505                              <1>
32506                              <1>       ; 17/03/2016
32507 0000B07D 386E03              <1>       cmp    [esi+LD_FATType], ch ; 0
32508 0000B080 7707                <1>       ja     short csftdf2_set_sf_percent_rsize1
32509                              <1>
32510 0000B082 B800000100          <1>       mov    eax, 65536 ; read/write buffer size for Singlix FS
32511 0000B087 EB06                <1>       jmp    short csftdf2_set_sf_percent_rsize2
32512                              <1>
32513                              <1> csftdf2_set_sf_percent_rsize1:
32514 0000B089 668B4611            <1>       mov    ax, [esi+LD_BPB+BytesPerSec]
32515 0000B08D F7E1                <1>       mul    ecx
32516                              <1>       ;sub   edx, edx
32517                              <1> csftdf2_set_sf_percent_rsize2:
32518 0000B08F A3[105F0100]        <1>       mov    [csftdf_r_size], eax
32519                              <1>
32520                              <1> csftdf2_set_df_percentage:
32521                              <1>       ;sub   eax, eax
32522                              <1>       ;mov   ah, [DestinationFile_Drv]
32523                              <1>       ;mov   edi, Logical_DOSDisks
32524                              <1>       ;add   edi, eax
32525                              <1>       ;mov   [csftdf_df_drv_dt], edi ; 17/03/2016
32526                              <1>
32527 0000B094 8B3D[285F0100]      <1>       mov    edi, [csftdf_df_drv_dt] ; 23/03/2016
32528                              <1>
32529                              <1>       ; 16/03/2016
32530                              <1>       ; Note: Singlix FS boot sector parameters (for cluster
32531                              <1>       ;      related calculations) has same offset
32532                              <1>       ;      values from LD_BPB as in FAT file system.
32533                              <1>       ;      [edi+LD_BPB+SecPerClust] is 1 for Singlix FS.
32534                              <1>       ;
32535                              <1>       ;movzx ecx, byte [edi+LD_BPB+SecPerClust]
32536 0000B09A 8A4F13              <1>       mov    cl, [edi+LD_BPB+SecPerClust]
32537 0000B09D 880D[CE5E0100]      <1>       mov    [DestinationFile_SecPerClust], cl
32538                              <1>
32539                              <1>       ; 17/03/2016
32540 0000B0A3 386F03              <1>       cmp    [edi+LD_FATType], ch ; 0
32541 0000B0A6 7707                <1>       ja     short csftdf2_set_df_percent_wsize1
32542                              <1>
32543 0000B0A8 B800000100          <1>       mov    eax, 65536 ; read/write buffer size for Singlix FS
```

```
32544 0000B0AD EB06               <1>       jmp    short csftdf2_set_df_percent_wsize2
32545                             <1>
32546                             <1> csftdf2_set_df_percent_wsize1:
32547 0000B0AF 0FB74711           <1>       movzx  eax, word [edi+LD_BPB+BytesPerSec]
32548 0000B0B3 F7E1               <1>       mul    ecx
32549                             <1>       ;sub   edx, edx
32550                             <1> csftdf2_set_df_percent_wsize2:
32551 0000B0B5 A3[145F0100]       <1>       mov    [csftdf_w_size], eax
32552                             <1>
32553 0000B0BA A1[FC5E0100]       <1>       mov    eax, [csftdf_filesize]
32554                             <1>
32555 0000B0BF 3D00000100         <1>       cmp    eax, 65536 ; 64KB   ; small file
32556 0000B0C4 721F               <1>       jb     short csftdf2_load_file ; do not display percentage
32557                             <1>
32558                             <1> csftdf2_reset_wf_percent_ptr_chk_64k:
32559 0000B0C6 B201               <1>       mov    dl, 1 ; 25/03/2016
32560                             <1>
32561 0000B0C8 3D00000400         <1>       cmp    eax, 65536*4 ; 256KB
32562 0000B0CD 7310               <1>       jnb    short csftdf2_enable_percentage_display ; big file
32563                             <1>
32564                             <1>       ; 64-128KB file size for floppy disks
32565 0000B0CF 3815[D05D0100]     <1>       cmp    byte [SourceFile_Drv], dl ; 1 ; read from floppy disk ?
32566 0000B0D5 7608               <1>       jna    short csftdf2_enable_percentage_display
32567                             <1>
32568 0000B0D7 3815[505E0100]     <1>       cmp    byte [DestinationFile_Drv], dl ; 1 ; write to floppy disk ?
32569 0000B0DD 7706               <1>       ja     short csftdf2_load_file
32570                             <1>
32571                             <1> csftdf2_enable_percentage_display:
32572 0000B0DF 8815[205F0100]     <1>       mov    [csftdf_percentage], dl ; 1
32573                             <1>
32574                             <1> csftdf2_load_file:
32575                             <1>       ; 13/05/2016
32576                             <1>       ; 19/03/2016
32577                             <1>       ; 18/03/2016
32578                             <1>       ; 17/03/2016
32579 0000B0E5 B40F               <1>       mov    ah, 0Fh
32580 0000B0E7 E8AE63FFFF         <1>       call   _int10h
32581                             <1>       ; 13/05/2016
32582 0000B0EC 883D[215F0100]     <1>       mov    [csftdf_videopage], bh ; active video page
32583 0000B0F2 B403               <1>       mov    ah, 03h
32584 0000B0F4 E8A163FFFF         <1>       call   _int10h
32585 0000B0F9 668915[225F0100]   <1>       mov    [csftdf_cursorpos], dx
32586                             <1>
32587 0000B100 29C0               <1>       sub    eax, eax
32588 0000B102 A2[F95E0100]       <1>       mov    [csftdf_rw_err], al ; 0
32589                             <1>
32590                             <1> ; ///
32591                             <1> csftdf_sf_amb: ; 15/03/2016
32592 0000B107 8B0D[FC5E0100]     <1>       mov    ecx, [csftdf_filesize]    ; 23/03/2016
32593                             <1>
32594                             <1>       ; TRDOS 386 (TRDOS v2.0)
32595                             <1>       ; Allocate contiguous memory block for loading the file
32596                             <1>
32597                             <1>       ;mov   ecx, [SourceFile_DirEntry+DirEntry_FileSize]
32598                             <1>
32599                             <1>       ;sub   eax, eax ; First free memory aperture
32600                             <1>
32601                             <1>       ; eax = 0 (Allocate memory from the beginning)
32602                             <1>       ; ecx = File (Allocation) size in bytes
32603                             <1>
32604 0000B10D E812A3FFFF         <1>       call   allocate_memory_block
32605 0000B112 7304               <1>       jnc    short loc_check_sf_save_loading_parms
32606                             <1>
32607 0000B114 29C0               <1>       sub    eax, eax
32608 0000B116 29C9               <1>       sub    ecx, ecx
32609                             <1>
32610                             <1> loc_check_sf_save_loading_parms:
32611 0000B118 A3[005F0100]       <1>       mov    [csftdf_sf_mem_addr], eax ; loading address
32612 0000B11D 890D[045F0100]     <1>       mov    [csftdf_sf_mem_bsize], ecx ; block size
32613                             <1> ; ///
32614                             <1>       ; 19/03/2016
32615 0000B123 8B35[245F0100]     <1>       mov    esi, [csftdf_sf_drv_dt] ; logical dos drv desc. tbl.
32616                             <1>
32617                             <1>       ; 17/03/2016
32618 0000B129 09C0               <1>       or     eax, eax ; contiguous free memory block address
32619 0000B12B 0F845B010000       <1>         jz      csftdf2_read_sf_cluster
32620                             <1>
32621                             <1>       ; 18/03/2016
32622 0000B131 8B1D[005F0100]     <1>       mov    ebx, [csftdf_sf_mem_addr] ; memory block address
32623                             <1>
32624 0000B137 807E0300           <1>       cmp    byte [esi+LD_FATType], 0
32625 0000B13B 0F8605020000       <1>         jna     csftdf2_load_fs_file
32626                             <1>
32627                             <1> csftdf2_load_fat_file:
32628 0000B141 53                 <1>       push   ebx ; *
32629                             <1>
32630                             <1> csftdf2_load_fat_file_next:
32631 0000B142 BE[230D0100]       <1>       mov    esi, msg_reading
32632 0000B147 E811B2FFFF         <1>       call   print_msg
32633                             <1>
32634 0000B14C 803D[205F0100]00   <1>       cmp    byte [csftdf_percentage], 0
32635 0000B153 7605               <1>       jna    short csftdf2_load_fat_file_1
32636                             <1>
32637 0000B155 E87C000000         <1>       call   csftdf2_print_percentage ; 19/03/2016
32638                             <1>
32639                             <1> csftdf2_load_fat_file_1:
32640 0000B15A 8B35[245F0100]     <1>       mov    esi, [csftdf_sf_drv_dt]
32641 0000B160 5B                 <1>       pop    ebx ; *
32642                             <1>
32643                             <1> csftdf2_load_fat_file_2:
32644 0000B161 E8B8000000         <1>       call   csftdf2_read_fat_file_sectors ; 19/03/2016
32645 0000B166 0F8250040000       <1>         jc      csftdf2_rw_error ; eocc! or disk error!
32646                             <1>
```

```
32647 0000B16C 09D2              <1>       or    edx, edx ; edx > 0 -> EOF
32648 0000B16E 7520              <1>       jnz   short csftdf2_load_fat_file_ok
32649                            <1>
32650 0000B170 803D[205F0100]00  <1>       cmp   byte [csftdf_percentage], 0
32651 0000B177 76E8              <1>       jna   short csftdf2_load_fat_file_2
32652                            <1>
32653 0000B179 53                <1>       push  ebx ; *
32654                            <1>
32655                            <1>       ; Set cursor position
32656                            <1>       ; AH= 02h, BH= Page Number, DH= Row, DL= Column
32657 0000B17A 8A3D[215F0100]    <1>       mov   bh, [csftdf_videopage]
32658 0000B180 668B15[225F0100]  <1>       mov   dx, [csftdf_cursorpos]
32659 0000B187 B402              <1>       mov   ah, 2
32660 0000B189 E80C63FFFF        <1>       call  _int10h
32661 0000B18E EBB2              <1>       jmp   short csftdf2_load_fat_file_next
32662                            <1>
32663                            <1> csftdf2_load_fat_file_ok:
32664 0000B190 803D[205F0100]00  <1>       cmp   byte [csftdf_percentage], 0
32665 0000B197 0F8651020000      <1>         jna     csftdf2_save_file ; 25/03/2016
32666                            <1>
32667                            <1>       ; "Reading... 100%"
32668 0000B19D BF[3B0D0100]      <1>       mov   edi, percentagestr
32669 0000B1A2 B031              <1>       mov   al, '1'
32670 0000B1A4 AA                <1>       stosb
32671 0000B1A5 B030              <1>       mov   al, '0'
32672 0000B1A7 AA                <1>       stosb
32673 0000B1A8 AA                <1>       stosb
32674                            <1>
32675 0000B1A9 8A3D[215F0100]    <1>       mov   bh, [csftdf_videopage]
32676 0000B1AF 668B15[225F0100]  <1>       mov   dx, [csftdf_cursorpos]
32677 0000B1B6 B402              <1>       mov   ah, 2
32678 0000B1B8 E8DD62FFFF        <1>       call  _int10h
32679                            <1>
32680 0000B1BD BE[230D0100]      <1>       mov   esi, msg_reading
32681 0000B1C2 E896B1FFFF        <1>       call  print_msg
32682                            <1>
32683 0000B1C7 BE[3B0D0100]      <1>       mov   esi, percentagestr
32684 0000B1CC E88CB1FFFF        <1>       call  print_msg
32685                            <1>
32686 0000B1D1 E918020000        <1>         jmp     csftdf2_save_file ; 25/03/2016
32687                            <1>
32688                            <1> csftdf2_print_percentage:
32689                            <1>       ; 09/12/2017
32690                            <1>       ; 19/03/2016
32691                            <1>       ; 18/03/2016
32692 0000B1D6 B020              <1>       mov   al, 20h
32693 0000B1D8 BF[3B0D0100]      <1>       mov   edi, percentagestr
32694 0000B1DD AA                <1>       stosb
32695 0000B1DE AA                <1>       stosb
32696 0000B1DF A1[185F0100]      <1>       mov   eax, [csftdf_sf_rbytes]
32697 0000B1E4 BA64000000        <1>       mov   edx, 100
32698 0000B1E9 F7E2              <1>       mul   edx
32699 0000B1EB 8B0D[FC5E0100]    <1>       mov   ecx, [csftdf_filesize]
32700 0000B1F1 F7F1              <1>       div   ecx
32701 0000B1F3 B10A              <1>       mov   cl, 10
32702 0000B1F5 F6F1              <1>       div   cl
32703 0000B1F7 80C430            <1>       add   ah, '0'
32704 0000B1FA 8827              <1>       mov   [edi], ah
32705 0000B1FC 20C0              <1>       and   al, al
32706 0000B1FE 740A              <1>       jz    short csftdf2_print_percent_1
32707 0000B200 4F                <1>       dec   edi
32708                            <1>       ;cbw
32709 0000B201 28E4              <1>       sub   ah, ah ; 09/12/2017
32710 0000B203 F6F1              <1>       div   cl
32711 0000B205 80C430            <1>       add   ah, '0'
32712 0000B208 8827              <1>       mov   [edi], ah
32713                            <1>       ;and  al, al
32714                            <1>       ;jz   short csftdf2_print_percent_1
32715                            <1>       ;dec  edi
32716                            <1>       ;mov  [edi], '1' ; 100%
32717                            <1>
32718                            <1> csftdf2_print_percent_1:
32719 0000B20A BE[3B0D0100]      <1>       mov   esi, percentagestr
32720                            <1>       ;call print_msg
32721                            <1>       ;retn
32722 0000B20F E949B1FFFF        <1>       jmp   print_msg
32723                            <1>
32724                            <1> csftdf2_read_file_sectors:
32725                            <1>       ; 19/03/2016
32726 0000B214 807E0300          <1>       cmp   byte [esi+LD_FATType], 0
32727 0000B218 0F8627070000      <1>         jna     csftdf2_read_fs_file_sectors
32728                            <1>
32729                            <1> csftdf2_read_fat_file_sectors:
32730                            <1>       ; 19/03/2016
32731                            <1>       ; 18/03/2016
32732                            <1>       ; return:
32733                            <1>       ;   CF = 0 & EDX > 0 -> END OF FILE
32734                            <1>       ;   CF = 0 & EDX = 0 -> not EOF
32735                            <1>       ;   CF = 1 -> read error (error code in AL)
32736                            <1>
32737                            <1> csftdf2_read_fat_file_secs_0:
32738 0000B21E 8B15[FC5E0100]    <1>       mov   edx, [csftdf_filesize]
32739 0000B224 2B15[185F0100]    <1>       sub   edx, [csftdf_sf_rbytes]
32740 0000B22A 3B15[105F0100]    <1>       cmp   edx, [csftdf_r_size]
32741 0000B230 7306              <1>       jnb   short csftdf2_read_fat_file_secs_1
32742 0000B232 8915[105F0100]    <1>       mov   [csftdf_r_size], edx
32743                            <1>
32744                            <1> csftdf2_read_fat_file_secs_1:
32745 0000B238 A1[105F0100]      <1>       mov   eax, [csftdf_r_size]
32746 0000B23D 29D2              <1>       sub   edx, edx
32747 0000B23F 0FB74E11          <1>       movzx ecx, word [esi+LD_BPB+BytesPerSec]
32748 0000B243 01C8              <1>       add   eax, ecx
32749 0000B245 48                <1>       dec   eax
```

```
32750 0000B246 F7F1              <1>       div    ecx
32751 0000B248 89C1              <1>       mov    ecx, eax ; sector count
32752 0000B24A A1[085F0100]      <1>       mov    eax, [csftdf_sf_cluster]
32753                            <1>
32754                            <1>       ; EBX = memory block address (current)
32755                            <1>
32756 0000B24F E821090000        <1>       call   read_fat_file_sectors
32757 0000B254 7235              <1>       jc     short csftdf2_read_fat_file_secs_3
32758                            <1>
32759                            <1>       ; EBX = next memory address
32760                            <1>
32761 0000B256 A1[185F0100]      <1>       mov    eax, [csftdf_sf_rbytes]
32762 0000B25B 0305[105F0100]    <1>       add    eax, [csftdf_r_size]
32763 0000B261 8B15[FC5E0100]    <1>       mov    edx, [csftdf_filesize]
32764 0000B267 39D0              <1>       cmp    eax, edx
32765 0000B269 7320              <1>       jnb    short csftdf2_read_fat_file_secs_3 ; edx > 0
32766 0000B26B A3[185F0100]      <1>       mov    [csftdf_sf_rbytes], eax
32767                            <1>
32768 0000B270 53                <1>       push   ebx ; *
32769                            <1>       ; get next cluster (csftdf_r_size! bytes)
32770 0000B271 A1[085F0100]      <1>       mov    eax, [csftdf_sf_cluster]
32771 0000B276 E8CC060000        <1>       call   get_next_cluster
32772 0000B27B 5B                <1>       pop    ebx ; *
32773 0000B27C 7306              <1>       jnc    short csftdf2_read_fat_file_secs_2
32774                            <1>
32775                            <1>       ; 15/10/2016
32776                            <1>       ;Disk read error instad of drv not ready err
32777 0000B27E B811000000        <1>       mov    eax, 17 ; Read error !
32778 0000B283 C3                <1>       retn
32779                            <1>
32780                            <1> csftdf2_read_fat_file_secs_2:
32781 0000B284 29D2              <1>       sub    edx, edx ; 0
32782 0000B286 A3[085F0100]      <1>       mov    [csftdf_sf_cluster], eax ; next cluster
32783                            <1>
32784                            <1> csftdf2_read_fat_file_secs_3:
32785 0000B28B C3                <1>       retn
32786                            <1>
32787                            <1> csftdf2_read_sf_cluster:
32788                            <1>       ; 19/03/2016
32789 0000B28C BB00000700        <1>       mov    ebx, Cluster_Buffer ; buffer address (64KB)
32790                            <1>
32791 0000B291 803D[205F0100]00  <1>       cmp    byte [csftdf_percentage], 0
32792 0000B298 760D              <1>       jna    short csftdf2_read_sf_clust_2
32793                            <1>
32794 0000B29A 53                <1>       push   ebx ; *
32795                            <1>
32796                            <1> csftdf2_read_sf_clust_next:
32797 0000B29B E836FFFFFF        <1>       call   csftdf2_print_percentage
32798                            <1>
32799                            <1> csftdf2_read_sf_clust_0:
32800 0000B2A0 8B35[245F0100]    <1>       mov    esi, [csftdf_sf_drv_dt]
32801                            <1> csftdf2_read_sf_clust_1:
32802 0000B2A6 5B                <1>       pop    ebx ; *
32803                            <1>
32804                            <1> csftdf2_read_sf_clust_2:
32805 0000B2A7 89DA              <1>       mov    edx, ebx
32806 0000B2A9 0315[105F0100]    <1>       add    edx, [csftdf_r_size]
32807 0000B2AF 81FA00000800      <1>       cmp    edx, Cluster_Buffer + 65536
32808 0000B2B5 772F              <1>       ja     short csftdf2_write_df_cluster
32809                            <1>
32810 0000B2B7 E858FFFFFF        <1>       call   csftdf2_read_file_sectors ; 19/03/2016
32811 0000B2BC 0F8280020000      <1>         jc     csftdf2_save_fat_file_err2 ; eocc! or disk error!
32812                            <1>
32813 0000B2C2 09D2              <1>       or     edx, edx ; edx > 0 -> EOF
32814 0000B2C4 7520              <1>       jnz    short csftdf2_write_df_cluster
32815                            <1>
32816 0000B2C6 803D[205F0100]00  <1>       cmp    byte [csftdf_percentage], 0
32817 0000B2CD 76D8              <1>       jna    short csftdf2_read_sf_clust_2
32818                            <1>
32819 0000B2CF 53                <1>       push   ebx ; *
32820                            <1>
32821                            <1>       ; Set cursor position
32822                            <1>       ; AH= 02h, BH= Page Number, DH= Row, DL= Column
32823 0000B2D0 8A3D[215F0100]    <1>       mov    bh, [csftdf_videopage]
32824 0000B2D6 668B15[225F0100]  <1>       mov    dx, [csftdf_cursorpos]
32825 0000B2DD B402              <1>       mov    ah, 2
32826 0000B2DF E8B661FFFF        <1>       call   _int10h
32827 0000B2E4 EBB5              <1>       jmp    short csftdf2_read_sf_clust_next
32828                            <1>
32829                            <1> csftdf2_write_df_cluster:
32830                            <1>       ; 19/03/2016
32831 0000B2E6 8B35[285F0100]    <1>       mov    esi, [csftdf_df_drv_dt]
32832 0000B2EC BB00000700        <1>       mov    ebx, Cluster_Buffer ; buffer address (64KB)
32833                            <1>
32834                            <1> csftdf2_write_df_clust_next:
32835 0000B2F1 E855000000        <1>       call   csftdf2_write_file_sectors ; 19/03/2016
32836 0000B2F6 0F8246020000      <1>         jc     csftdf2_save_fat_file_err2 ; eocc! or disk error!
32837                            <1>
32838 0000B2FC 09D2              <1>       or     edx, edx ; edx > 0 -> EOF
32839 0000B2FE 750A              <1>       jnz    short csftdf2_rw_f_clust_ok
32840                            <1>
32841 0000B300 81FB00000800      <1>       cmp    ebx, Cluster_Buffer + 65536
32842 0000B306 72E9              <1>       jb     short csftdf2_write_df_clust_next
32843                            <1>
32844 0000B308 EB82              <1>       jmp    short csftdf2_read_sf_cluster
32845                            <1>
32846                            <1> csftdf2_rw_f_clust_ok:
32847 0000B30A 803D[205F0100]00  <1>       cmp    byte [csftdf_percentage], 0
32848 0000B311 0F86B2010000      <1>         jna    csftdf2_save_fat_file_4 ; 25/03/2016
32849                            <1>
32850                            <1>       ; "100%"
32851 0000B317 BF[3B0D0100]      <1>       mov    edi, percentagestr
32852 0000B31C B031              <1>       mov    al, '1'
```

```
32853 0000B31E AA                <1>       stosb
32854 0000B31F B030              <1>       mov    al, '0'
32855 0000B321 AA                <1>       stosb
32856 0000B322 AA                <1>       stosb
32857                            <1>
32858 0000B323 8A3D[215F0100]    <1>       mov    bh, [csftdf_videopage]
32859 0000B329 668B15[225F0100]  <1>       mov    dx, [csftdf_cursorpos]
32860 0000B330 B402              <1>       mov    ah, 2
32861 0000B332 E86361FFFF        <1>       call   _int10h
32862                            <1>
32863 0000B337 BE[3B0D0100]      <1>       mov    esi, percentagestr
32864 0000B33C E81CB0FFFF        <1>       call   print_msg
32865                            <1>
32866 0000B341 E983010000        <1>         jmp     csftdf2_save_fat_file_4
32867                            <1>
32868                            <1> csftdf2_load_fs_file:
32869                            <1>       ; temporary - 18/03/2016
32870 0000B346 E96F020000        <1>         jmp     csftdf2_read_error
32871                            <1>
32872                            <1> csftdf2_write_file_sectors:
32873                            <1>       ; 19/03/2016
32874 0000B34B 807E0300          <1>       cmp    byte [esi+LD_FATType], 0
32875 0000B34F 0F86F1050000      <1>         jna     csftdf2_write_fs_file_sectors
32876                            <1>
32877                            <1> csftdf2_write_fat_file_sectors:
32878                            <1>       ; 19/03/2016
32879                            <1>       ; 18/03/2016
32880                            <1>       ; return:
32881                            <1>       ;   CF = 0 & EDX > 0 -> END OF FILE
32882                            <1>       ;   CF = 0 & EDX = 0 -> not EOF
32883                            <1>       ;   CF = 1 -> write error (error code in AL)
32884                            <1>
32885                            <1> csftdf2_write_fat_file_secs_0:
32886 0000B355 8B15[FC5E0100]    <1>       mov    edx, [csftdf_filesize]
32887 0000B35B 2B15[1C5F0100]    <1>       sub    edx, [csftdf_df_wbytes]
32888 0000B361 3B15[145F0100]    <1>       cmp    edx, [csftdf_w_size]
32889 0000B367 7306              <1>       jnb    short csftdf2_write_fat_file_secs_1
32890 0000B369 8915[145F0100]    <1>       mov    [csftdf_w_size], edx
32891                            <1>
32892                            <1> csftdf2_write_fat_file_secs_1:
32893 0000B36F A1[145F0100]      <1>       mov    eax, [csftdf_w_size]
32894 0000B374 29D2              <1>       sub    edx, edx
32895 0000B376 0FB74E11          <1>       movzx  ecx, word [esi+LD_BPB+BytesPerSec]
32896 0000B37A 01C8              <1>       add    eax, ecx
32897 0000B37C 48                <1>       dec    eax
32898 0000B37D F7F1              <1>       div    ecx
32899 0000B37F 89C1              <1>       mov    ecx, eax ; sector count
32900 0000B381 A1[0C5F0100]      <1>       mov    eax, [csftdf_df_cluster]
32901                            <1>
32902                            <1>       ; EBX = memory block address (current)
32903                            <1>
32904 0000B386 E8A20F0000        <1>       call   write_fat_file_sectors
32905 0000B38B 7259              <1>       jc     short csftdf2_write_fat_file_secs_4
32906                            <1>
32907                            <1>       ; EBX = next memory address
32908                            <1>
32909 0000B38D A1[1C5F0100]      <1>       mov    eax, [csftdf_df_wbytes]
32910 0000B392 0305[145F0100]    <1>       add    eax, [csftdf_w_size]
32911 0000B398 8B15[FC5E0100]    <1>       mov    edx, [csftdf_filesize]
32912 0000B39E 39D0              <1>       cmp    eax, edx
32913 0000B3A0 7344              <1>       jnb    short csftdf2_write_fat_file_secs_4
32914 0000B3A2 A3[1C5F0100]      <1>       mov    [csftdf_df_wbytes], eax
32915                            <1>       ;
32916 0000B3A7 A3[BE5E0100]      <1>       mov    [DestinationFile_DirEntry+DirEntry_FileSize], eax
32917                            <1>
32918 0000B3AC 53                <1>       push   ebx ; *
32919                            <1>
32920 0000B3AD 803D[FA5E0100]01  <1>       cmp    byte [DestinationFileFound], 1
32921 0000B3B4 7210              <1>       jb     short csftdf2_write_fat_file_secs_2
32922                            <1>
32923                            <1>       ; get next cluster (csftdf_w_size! bytes)
32924 0000B3B6 A1[0C5F0100]      <1>       mov    eax, [csftdf_df_cluster]
32925 0000B3BB E887050000        <1>       call   get_next_cluster
32926 0000B3C0 731C              <1>       jnc    short csftdf2_write_fat_file_secs_3
32927                            <1>
32928 0000B3C2 21C0              <1>       and    eax, eax ; end of cluster chain!?
32929 0000B3C4 7521              <1>       jnz    short csftdf2_write_fat_file_secs_5 ; disk error !
32930                            <1>
32931                            <1> csftdf2_write_fat_file_secs_2:
32932 0000B3C6 A1[0C5F0100]      <1>       mov    eax, [csftdf_df_cluster] ; last cluster
32933 0000B3CB E8800E0000        <1>       call   add_new_cluster
32934 0000B3D0 7215              <1>       jc     short csftdf2_write_fat_file_secs_5
32935                            <1>
32936                            <1>       ; NOTE: Destination file size may be bigger than
32937                            <1>       ; source file size when the last reading fails after here.
32938                            <1>       ; (The last -empty- cluster of destination file must be
32939                            <1>       ; truncated and LMDT must be current date&time for partial
32940                            <1>       ; copy result!)
32941 0000B3D2 8B15[145F0100]    <1>       mov    edx, [csftdf_w_size] ; bytes per cluster
32942 0000B3D8 0115[BE5E0100]    <1>       add    [DestinationFile_DirEntry+DirEntry_FileSize], edx
32943                            <1>
32944                            <1> csftdf2_write_fat_file_secs_3:
32945 0000B3DE 5B                <1>       pop    ebx ; *
32946 0000B3DF 29D2              <1>       sub    edx, edx ; 0
32947 0000B3E1 A3[0C5F0100]      <1>       mov    [csftdf_df_cluster], eax ; next cluster
32948                            <1>
32949                            <1> csftdf2_write_fat_file_secs_4:
32950 0000B3E6 C3                <1>       retn
32951                            <1>
32952                            <1> csftdf2_write_fat_file_secs_5:
32953 0000B3E7 5B                <1>       pop    ebx ; *
32954                            <1>       ; 16/10/2016 (1Dh -> 18)
32955 0000B3E8 B812000000        <1>       mov    eax, 18 ; Write error !
```

```
32956 0000B3ED C3                 <1>       retn
32957                             <1>
32958                             <1> csftdf2_save_file:
32959                             <1>       ; 09/12/2017
32960                             <1>       ; 25/03/2016
32961                             <1>       ; 19/03/2016
32962                             <1>       ; 18/03/2016
32963 0000B3EE 8B35[285F0100]     <1>       mov    esi, [csftdf_df_drv_dt] ; logical dos drv desc. tbl.
32964                             <1>
32965 0000B3F4 8B1D[005F0100]     <1>       mov    ebx, [csftdf_sf_mem_addr] ; memory block address
32966                             <1>
32967 0000B3FA 807E0300           <1>       cmp    byte [esi+LD_FATType], 0
32968 0000B3FE 0F86F4010000       <1>         jna     csftdf2_save_fs_file
32969                             <1>
32970                             <1> csftdf2_save_fat_file:
32971 0000B404 53                 <1>       push   ebx; *
32972                             <1>
32973 0000B405 803D[205F0100]00   <1>       cmp    byte [csftdf_percentage], 0
32974 0000B40C 7724               <1>       ja     short csftdf2_save_fat_file_0
32975                             <1>
32976                             <1>       ; Set cursor position
32977                             <1>       ; AH= 02h, BH= Page Number, DH= Row, DL= Column
32978 0000B40E 8A3D[215F0100]     <1>       mov    bh, [csftdf_videopage]
32979 0000B414 668B15[225F0100]   <1>       mov    dx, [csftdf_cursorpos]
32980 0000B41B B402               <1>       mov    ah, 2
32981 0000B41D E87860FFFF         <1>       call   _int10h
32982                             <1>
32983 0000B422 BE[2F0D0100]       <1>       mov    esi, msg_writing
32984 0000B427 E831AFFFFF         <1>       call   print_msg
32985                             <1>
32986                             <1> csftdf2_save_fat_file_next:
32987 0000B42C 8B35[285F0100]     <1>       mov    esi, [csftdf_df_drv_dt] ; 25/03/2016
32988                             <1>
32989                             <1> csftdf2_save_fat_file_0:
32990 0000B432 5B                 <1>       pop    ebx ; *
32991                             <1>
32992                             <1> csftdf2_save_fat_file_1:
32993 0000B433 E813FFFFFF         <1>       call   csftdf2_write_file_sectors ; 19/03/2016
32994 0000B438 0F827E010000       <1>         jc      csftdf2_rw_error ; eocc! or disk error!
32995                             <1>
32996 0000B43E 09D2               <1>       or     edx, edx ; edx > 0 -> EOF
32997 0000B440 756D               <1>         jnz     short csftdf2_save_fat_file_3 ; 25/03/2016
32998                             <1>
32999 0000B442 803D[205F0100]00   <1>       cmp    byte [csftdf_percentage], 0
33000 0000B449 76E8               <1>       jna    short csftdf2_save_fat_file_1
33001                             <1>
33002 0000B44B B020               <1>       mov    al, 20h
33003 0000B44D BF[3B0D0100]       <1>       mov    edi, percentagestr
33004 0000B452 AA                 <1>       stosb
33005 0000B453 AA                 <1>       stosb
33006 0000B454 A1[1C5F0100]       <1>       mov    eax, [csftdf_df_wbytes]
33007 0000B459 BA64000000         <1>       mov    edx, 100
33008 0000B45E F7E2               <1>       mul    edx
33009 0000B460 8B0D[FC5E0100]     <1>       mov    ecx, [csftdf_filesize]
33010 0000B466 F7F1               <1>       div    ecx
33011 0000B468 B10A               <1>       mov    cl, 10
33012 0000B46A F6F1               <1>       div    cl
33013 0000B46C 80C430             <1>       add    ah, '0'
33014 0000B46F 8827               <1>       mov    [edi], ah
33015 0000B471 20C0               <1>       and    al, al
33016 0000B473 740A               <1>       jz     short csftdf2_save_fat_file_2
33017 0000B475 4F                 <1>       dec    edi
33018                             <1>       ;cbw
33019 0000B476 30E4               <1>       xor    ah, ah ; 09/12/2017
33020 0000B478 F6F1               <1>       div    cl
33021 0000B47A 80C430             <1>       add    ah, '0'
33022 0000B47D 8827               <1>       mov    [edi], ah
33023                             <1>       ;and   al, al
33024                             <1>       ;jz    short csftdf2_save_fat_file_2
33025                             <1>       ;dec   edi
33026                             <1>       ;mov   [edi], '1' ; 100%
33027                             <1>
33028                             <1> csftdf2_save_fat_file_2:
33029 0000B47F 53                 <1>       push   ebx ; *
33030                             <1>
33031 0000B480 E802000000         <1>       call   csftdf2_print_wr_percentage ; 25/03/2016
33032                             <1>
33033 0000B485 EBA5               <1>         jmp     csftdf2_save_fat_file_next
33034                             <1>
33035                             <1> csftdf2_print_wr_percentage:
33036                             <1>       ; Set cursor position
33037                             <1>       ; AH= 02h, BH= Page Number, DH= Row, DL= Column
33038 0000B487 8A3D[215F0100]     <1>       mov    bh, [csftdf_videopage]
33039 0000B48D 668B15[225F0100]   <1>       mov    dx, [csftdf_cursorpos]
33040 0000B494 B402               <1>       mov    ah, 2
33041 0000B496 E8FF5FFFFF         <1>       call   _int10h
33042                             <1>
33043 0000B49B BE[2F0D0100]       <1>       mov    esi, msg_writing
33044 0000B4A0 E8B8AEFFFF         <1>       call   print_msg
33045                             <1>
33046 0000B4A5 BE[3B0D0100]       <1>       mov    esi, percentagestr
33047                             <1>       ;call  print_msg
33048                             <1>       ;retn
33049 0000B4AA E9AEAEFFFF         <1>       jmp    print_msg
33050                             <1>
33051                             <1> csftdf2_save_fat_file_3:
33052 0000B4AF 803D[205F0100]00   <1>       cmp    byte [csftdf_percentage], 0
33053 0000B4B6 7611               <1>         jna     csftdf2_save_fat_file_4 ; 25/03/2016
33054                             <1>
33055                             <1>       ; "100%"
33056 0000B4B8 BF[3B0D0100]       <1>       mov    edi, percentagestr
33057 0000B4BD B031               <1>       mov    al, '1'
33058 0000B4BF AA                 <1>       stosb
```

```
33059 0000B4C0 B030                <1>       mov    al, '0'
33060 0000B4C2 AA                  <1>       stosb
33061 0000B4C3 AA                  <1>       stosb
33062                              <1>
33063 0000B4C4 E8BEFFFFFF          <1>       call   csftdf2_print_wr_percentage
33064                              <1>
33065                              <1> csftdf2_save_fat_file_4:
33066 0000B4C9 803D[FA5E0100]00    <1>       cmp    byte [DestinationFileFound], 0
33067 0000B4D0 7647                <1>       jna    short csftdf2_save_fat_file_6
33068                              <1>
33069 0000B4D2 8B35[285F0100]      <1>       mov    esi, [csftdf_df_drv_dt] ; 31/03/2016
33070                              <1>
33071 0000B4D8 A1[0C5F0100]        <1>       mov    eax, [csftdf_df_cluster] ; last cluster
33072 0000B4DD E865040000          <1>       call   get_next_cluster
33073 0000B4E2 7235                <1>       jc     short csftdf2_save_fat_file_6 ; eocc! or disk error!
33074                              <1>
33075 0000B4E4 A1[0C5F0100]        <1>       mov    eax, [csftdf_df_cluster] ; last cluster
33076                              <1>       ;xor   ecx, ecx
33077                              <1>       ;mov   [FAT_ClusterCounter], ecx ; 0 ; reset
33078                              <1>       ;dec   ecx ; 0FFFFFFFFh
33079                              <1>       ;shr   ecx, 4 ; 28 bit ; 0FFFFFFFh
33080 0000B4E9 B9FFFFFF0F          <1>       mov    ecx, 0FFFFFFFh
33081 0000B4EE E87E070000          <1>       call   update_cluster
33082 0000B4F3 7224                <1>       jc     short csftdf2_save_fat_file_6 ; really last cluster!?
33083                              <1>
33084 0000B4F5 A3[0C5F0100]        <1>       mov    [csftdf_df_cluster], eax ; next cluster
33085                              <1>
33086                              <1>       ; byte [FAT_BuffValidData] = 2
33087 0000B4FA E82F0A0000          <1>       call   save_fat_buffer
33088 0000B4FF 730E                <1>       jnc    short csftdf2_save_fat_file_5
33089                              <1>
33090 0000B501 8B15[FC5E0100]      <1>       mov    edx, [csftdf_filesize]
33091 0000B507 8915[BE5E0100]      <1>       mov    [DestinationFile_DirEntry+DirEntry_FileSize], edx
33092 0000B50D EB58                <1>       jmp    short csftdf2_save_fat_file_err3
33093                              <1>
33094                              <1> csftdf2_save_fat_file_5:
33095 0000B50F A1[0C5F0100]        <1>       mov    eax, [csftdf_df_cluster]
33096                              <1>
33097                              <1>       ; EAX = First cluster to be truncated/unlinked
33098                              <1>       ; ESI = Logical dos drive description table address
33099 0000B514 E8580C0000          <1>       call   truncate_cluster_chain
33100                              <1>
33101                              <1> csftdf2_save_fat_file_6:
33102                              <1>       ; 28/03/2016
33103 0000B519 BE[2D5E0100]        <1>       mov    esi, SourceFile_DirEntry+DirEntry_Attr ; +11 to + 18
33104 0000B51E BF[AD5E0100]        <1>       mov    edi, DestinationFile_DirEntry+DirEntry_Attr ; +11 to + 18
33105 0000B523 A4                  <1>       movsb ; +11
33106 0000B524 A5                  <1>       movsd ; +12 .. +15
33107 0000B525 66A5                <1>       movsw ; +16 .. +17
33108                              <1>             ; + 18
33109 0000B527 83C604              <1>       add    esi, 4
33110 0000B52A 83C704              <1>       add    edi, 4
33111 0000B52D A5                  <1>       movsd  ; DirEntry_WrtTime ; +22 .. +25
33112                              <1>
33113 0000B52E 8B15[FC5E0100]      <1>       mov    edx, [csftdf_filesize]
33114 0000B534 8915[BE5E0100]      <1>       mov    [DestinationFile_DirEntry+DirEntry_FileSize], edx
33115                              <1>
33116 0000B53A E8B8F0FFFF          <1>       call   convert_current_date_time
33117                              <1>       ; DX = Date in dos dir entry format
33118                              <1>       ; AX = Time in dos dir entry format
33119 0000B53F EB4D                <1>       jmp    short csftdf2_save_fat_file_7
33120                              <1>
33121                              <1> csftdf2_save_fat_file_err1:
33122 0000B541 5B                  <1>       pop    ebx ; *
33123                              <1> csftdf2_save_fat_file_err2:
33124 0000B542 A1[1C5F0100]        <1>       mov    eax, [csftdf_df_wbytes]
33125 0000B547 8B15[BE5E0100]      <1>       mov    edx, [DestinationFile_DirEntry+DirEntry_FileSize]
33126 0000B54D 39C2                <1>       cmp    edx, eax
33127 0000B54F 7616                <1>       jna    short csftdf2_save_fat_file_err3
33128 0000B551 A1[0C5F0100]        <1>       mov    eax, [csftdf_df_cluster] ; last (empty) cluster
33129                              <1>       ; ESI = Logical dos drive description table address
33130 0000B556 E8160C0000          <1>       call   truncate_cluster_chain
33131 0000B55B 720A                <1>       jc     short csftdf2_save_fat_file_err3
33132 0000B55D A1[1C5F0100]        <1>       mov    eax, [csftdf_df_wbytes]
33133 0000B562 A3[BE5E0100]        <1>       mov    [DestinationFile_DirEntry+DirEntry_FileSize], eax
33134                              <1> csftdf2_save_fat_file_err3:
33135 0000B567 E88BF0FFFF          <1>       call   convert_current_date_time
33136                              <1>       ; DX = Date in dos dir entry format
33137                              <1>       ; AX = Time in dos dir entry format
33138 0000B56C C605[AF5E0100]00    <1>       mov    byte [DestinationFile_DirEntry+DirEntry_CrtTimeTenth], 0
33139 0000B573 66A3[B05E0100]      <1>       mov    [DestinationFile_DirEntry+DirEntry_CrtTime], ax
33140 0000B579 668915[B25E0100]    <1>       mov    [DestinationFile_DirEntry+DirEntry_CrtDate], dx
33141 0000B580 66A3[B85E0100]      <1>       mov    [DestinationFile_DirEntry+DirEntry_WrtTime], ax
33142 0000B586 668915[BA5E0100]    <1>       mov    [DestinationFile_DirEntry+DirEntry_WrtDate], dx
33143 0000B58D F9                  <1>       stc
33144                              <1> csftdf2_save_fat_file_7:
33145 0000B58E 9C                  <1>       pushf
33146 0000B58F 668915[B45E0100]    <1>       mov    [DestinationFile_DirEntry+DirEntry_LastAccDate], dx
33147 0000B596 BE[A25E0100]        <1>       mov    esi, DestinationFile_DirEntry
33148 0000B59B BF00000800          <1>       mov    edi, Directory_Buffer
33149 0000B5A0 0FB70D[CA5E0100]    <1>       movzx  ecx, word [DestinationFile_DirEntryNumber] ; (<2048)
33150 0000B5A7 66C1E105            <1>       shl    cx, 5 ; 32 * directory entry number
33151 0000B5AB 01CF                <1>       add    edi, ecx
33152                              <1>       ;mov   ecx, 8
33153 0000B5AD 66B90800            <1>       mov    cx, 8
33154 0000B5B1 F3A5                <1>       rep    movsd
33155 0000B5B3 9D                  <1>       popf
33156 0000B5B4 730B                <1>       jnc    short csftdf2_write_file_OK
33157                              <1>
33158                              <1> csftdf2_write_error:
33159                              <1>       ; 18/03/2016
33160 0000B5B6 B01D                <1>       mov    al, 1Dh ; write error
33161 0000B5B8 EB02                <1>       jmp    short csftdf2_rw_error
```

```
                                <1>
                                <1>       ; 16/03/2016
                                <1> csftdf2_read_error:
0000B5BA B011                   <1>       mov    al, 17 ; ; Drive not ready or read error!
                                <1> csftdf2_rw_error:
0000B5BC A2[F95E0100]           <1>       mov    [csftdf_rw_err], al
                                <1>
                                <1> csftdf2_write_file_OK:
                                <1>       ; 18/03/2016
0000B5C1 C605[105B0100]02       <1>       mov    byte [DirBuff_ValidData], 2
0000B5C8 E8C8F0FFFF             <1>       call   save_directory_buffer
                                <1>
                                <1>       ; Update last modification date&time of destination
                                <1>       ; file's (parent) directory
0000B5CD E85EF1FFFF             <1>       call   update_parent_dir_lmdt
                                <1>       ;
0000B5D2 A1[005F0100]           <1>       mov    eax, [csftdf_sf_mem_addr] ; start address
                                <1>
0000B5D7 21C0                   <1>       and    eax, eax
0000B5D9 750E                   <1>       jnz    short csftdf2_dealloc_mblock
                                <1>
0000B5DB 88C5                   <1>       mov    ch, al ; 0 (Cluster r/w, not full loading)
                                <1> csftdf2_dealloc_retn:
0000B5DD 8A0D[F95E0100]         <1>       mov    cl, [csftdf_rw_err]
0000B5E3 A1[0C5F0100]           <1>       mov    eax, [csftdf_df_cluster]
0000B5E8 C3                     <1>       retn
                                <1>
                                <1> csftdf2_dealloc_mblock:
0000B5E9 8B0D[045F0100]         <1>       mov    ecx, [csftdf_sf_mem_bsize] ; block size
0000B5EF E83DA0FFFF             <1>       call   deallocate_memory_block
0000B5F4 B5FF                   <1>         mov     ch, 0FFh ; (File was full loaded at memory)
0000B5F6 EBE5                   <1>       jmp    short csftdf2_dealloc_retn
                                <1>
                                <1> csftdf2_save_fs_file:
                                <1>       ; 16/10/2016 (1Dh -> 18)
                                <1>       ; temporary - (21/03/2016)
0000B5F8 B812000000             <1>       mov    eax, 18 ; write error
0000B5FD F9                     <1>       stc
0000B5FE C3                     <1>       retn
                                <1>
                                <1> create_file:
                                <1>       ; 16/10/2016
                                <1>       ; 24/03/2016, 31/03/2016
                                <1>       ; 20/03/2016, 21/03/2016, 23/03/2016
                                <1>       ; 19/03/2016 (TRDOS 396 = TRDOS v2.0)
                                <1>       ; 03/09/2011 (FILE.ASM, 'proc_create_file')
                                <1>       ; 09/08/2010
                                <1>       ;
                                <1>       ; INPUT ->
                                <1>       ;     EAX = File Size
                                <1>       ;     ESI = ASCIIZ File Name
                                <1>       ;     CL = File Attributes
                                <1>       ;     EBX = FFFFFFFFh -> create empty file
                                <1>       ;                        (only for FAT fs)
                                <1>       ; OUTPUT ->
                                <1>       ;     CF = 0 ->
                                <1>       ;     EAX = New file's first cluster
                                <1>       ;     ESI = Logical Dos Drv Descr. Table Addr.
                                <1>       ;     EBX = offset CreateFile_Size
                                <1>       ;     ECX = Sectors per cluster (<256)
                                <1>       ;     EDX = Directory entry index/number (<65536)
                                <1>       ;     CF = 1 -> error code in AL
                                <1>
                                <1> ;     test   cl, 18h (directory or volume name)
                                <1> ;     jnz    short loc_createfile_access_denied
0000B5FF 80E107                 <1>       and    cl, 07h ; S, H, R
0000B602 880D[485F0100]         <1>         mov     [createfile_attrib], cl
                                <1>
0000B608 89D9                   <1>       mov    ecx, ebx
0000B60A 89F3                   <1>       mov    ebx, esi ; ASCIIZ File Name address
0000B60C 29D2                   <1>       sub    edx, edx
0000B60E 8A35[E6520100]         <1>         mov     dh, [Current_Drv]
0000B614 BE00010900             <1>         mov     esi, Logical_DOSDisks
0000B619 01D6                   <1>       add    esi, edx
                                <1>
0000B61B 8815[535F0100]         <1>       mov    [createfile_UpdatePDir], dl ; 0 ; 31/03/2016
                                <1>
                                <1>       ; LD_DiskType = 0 for write protection (read only)
0000B621 807E0101               <1>       cmp    byte [esi+LD_DiskType], 1 ; 0 = Invalid
0000B625 730A                   <1>       jnb    short loc_createfile_check_file_sytem
                                <1>       ; 16/10/2016 (TRDOS Error code: 30, disk write protected)
0000B627 B81E000000             <1>       mov    eax, 30 ; 13h, MSDOS err : Disk write-protected
0000B62C 66BA0000               <1>       mov    dx, 0
                                <1>       ; err retn: EDX = 0, EBX = File name offset
                                <1>       ; ESI -> Dos drive description table address
0000B630 C3                     <1>       retn
                                <1>
                                <1> ;loc_createfile_access_denied:
                                <1> ;     mov    eax, 05h ; access denied (invalid attributes input)
                                <1> ;     stc
                                <1> ;     retn
                                <1>
                                <1> loc_createfile_check_file_sytem:
0000B631 807E0301               <1>       cmp    byte [esi+LD_FATType], 1
0000B635 730A                   <1>       jnb    short loc_createfile_chk_empty_FAT_file_sign1
                                <1>
0000B637 A3[345F0100]           <1>       mov    [createfile_size], eax
                                <1>       ; ESI = Logical Dos Drive Description Table address
                                <1>       ; EBX = ASCIIZ File Name address
0000B63C E9FE020000             <1>       jmp    create_fs_file
                                <1>
                                <1> loc_createfile_chk_empty_FAT_file_sign1:
                                <1>       ; ECX = FFFFFFFFh -> create empty file if drive has FAT fs
```

```
33265 0000B641 41                <1>       inc    ecx
33266 0000B642 7506              <1>       jnz    short loc_createfile_chk_empty_FAT_file_sign2
33267 0000B644 890D[345F0100]    <1>       mov    [createfile_size], ecx ; 0 ; empty file
33268                            <1>
33269                            <1> loc_createfile_chk_empty_FAT_file_sign2:
33270                            <1>       ; 23/03/2016
33271 0000B64A 668B4E11          <1>       mov    cx, [esi+LD_BPB+BytesPerSec]
33272 0000B64E 66890D[505F0100]  <1>       mov    [createfile_BytesPerSec], cx
33273                            <1>
33274                            <1>       ; EBX = ASCIIZ File Name address
33275 0000B655 0FB65613          <1>       movzx  edx, byte [esi+LD_BPB+SecPerClust]
33276 0000B659 8815[495F0100]    <1>       mov    [createfile_SecPerClust], dl
33277 0000B65F 8B4E74            <1>       mov    ecx, [esi+LD_FreeSectors]
33278 0000B662 39D1              <1>       cmp    ecx, edx ; byte [createfile_SecPerClust]
33279 0000B664 7306              <1>       jnb    short loc_create_fat_file
33280                            <1>
33281                            <1> loc_createfile_insufficient_disk_space:
33282 0000B666 B827000000        <1>       mov    eax, 27h
33283                            <1> loc_createfile_gffc_retn:
33284 0000B66B C3                <1>       retn
33285                            <1>
33286                            <1> loc_create_fat_file:
33287 0000B66C 891D[2C5F0100]    <1>       mov    [createfile_Name_Offset], ebx
33288 0000B672 890D[305F0100]    <1>       mov    [createfile_FreeSectors], ecx
33289                            <1>
33290                            <1> loc_createfile_gffc_1:
33291 0000B678 E821050000        <1>       call   get_first_free_cluster
33292 0000B67D 72EC              <1>       jc     short loc_createfile_gffc_retn
33293                            <1>
33294 0000B67F A3[385F0100]      <1>       mov    [createfile_FFCluster], eax
33295                            <1>
33296                            <1> loc_createfile_locate_ffe_on_directory:
33297                            <1>       ; Current directory fcluster <> Directory buffer cluster
33298                            <1>       ; Current directory will be reloaded by
33299                            <1>       ; 'locate_current_dir_file' procedure
33300                            <1>       ;
33301                            <1>       ; ESI = Logical Dos Drv Desc. Table Adress
33302 0000B684 56                <1>       push   esi ; *
33303 0000B685 31C0              <1>       xor    eax, eax
33304                            <1>
33305 0000B687 A3[065B0100]      <1>       mov    dword [FAT_ClusterCounter], eax ; 0
33306                            <1>       ; 21/03/2016
33307 0000B68C A2[525F0100]      <1>       mov    byte [createfile_wfc], al ; 0
33308                            <1>
33309 0000B691 89C1              <1>       mov    ecx, eax
33310 0000B693 6649              <1>       dec    cx ; FFFFh
33311                            <1>       ; CX = FFFFh -> find first deleted or free entry
33312                            <1>       ; ESI would be ASCIIZ filename address if the call
33313                            <1>       ; would not be for first free or deleted dir entry
33314 0000B695 E8D5E7FFFF        <1>       call   locate_current_dir_file
33315 0000B69A 0F83EE000000      <1>       jnc    loc_createfile_set_ff_dir_entry
33316 0000B6A0 5E                <1>       pop    esi ; *
33317                            <1>        ; ESI = Logical DOS Drv. Description Table Address
33318 0000B6A1 83F802            <1>       cmp    eax, 2
33319 0000B6A4 7402              <1>       je     short loc_createfile_add_new_cluster
33320                            <1> loc_createfile_locate_file_stc_retn:
33321 0000B6A6 F9                <1>       stc
33322 0000B6A7 C3                <1>       retn
33323                            <1>
33324                            <1> loc_createfile_add_new_cluster:
33325 0000B6A8 803D[E5520100]02  <1>       cmp    byte [Current_FATType], 2
33326                            <1>       ;cmp   byte [esi+LD_FATType], 2
33327 0000B6AF 770C              <1>       ja     short loc_createfile_add_new_cluster_check_fsc
33328 0000B6B1 803D[E4520100]01  <1>       cmp    byte [Current_Dir_Level], 1
33329                            <1>       ;cmp   byte [esi+LD_CDirLevel], 1
33330 0000B6B8 7303              <1>       jnb    short loc_createfile_add_new_cluster_check_fsc
33331                            <1>
33332                            <1>       ;mov   eax, 12
33333 0000B6BA B00C              <1>       mov    al, 12 ; No more files
33334                            <1>
33335                            <1> loc_createfile_anc_retn:
33336 0000B6BC C3                <1>       retn
33337                            <1>
33338                            <1> loc_createfile_add_new_cluster_check_fsc:
33339 0000B6BD 8B0D[305F0100]    <1>       mov    ecx, [createfile_FreeSectors]
33340 0000B6C3 0FB605[495F0100]  <1>       movzx  eax, byte [createfile_SecPerClust]
33341 0000B6CA 66D1E0            <1>       shl    ax, 1 ; AX = 2 * AX
33342 0000B6CD 39C1              <1>       cmp    ecx, eax
33343 0000B6CF 7295              <1>         jb   short loc_createfile_insufficient_disk_space
33344                            <1>
33345                            <1> loc_createfile_add_new_subdir_cluster:
33346 0000B6D1 8B15[155B0100]    <1>       mov    edx, [DirBuff_Cluster]
33347 0000B6D7 8915[3C5F0100]    <1>       mov    [createfile_LastDirCluster], edx
33348                            <1>
33349 0000B6DD A1[385F0100]      <1>       mov    eax, [createfile_FFCluster]
33350 0000B6E2 E846040000        <1>       call   load_FAT_sub_directory
33351 0000B6E7 72D3              <1>       jc     short loc_createfile_anc_retn
33352                            <1>
33353                            <1> pass_createfile_add_new_subdir_cluster:
33354                            <1>       ;movzx eax, word [esi+LD_BPB+BytesPerSec]
33355 0000B6E9 0FB705[505F0100]  <1>       movzx  eax, word [createfile_BytesPerSec] ; 23/03/2016
33356 0000B6F0 F7E1              <1>       mul    ecx ; ecx = directory buffer sector count
33357 0000B6F2 89C1              <1>       mov    ecx, eax
33358 0000B6F4 C1E902            <1>       shr    ecx, 2 ; dword count
33359 0000B6F7 29C0              <1>       sub    eax, eax ; 0
33360 0000B6F9 F3AB              <1>       rep    stosd
33361                            <1>       ;
33362 0000B6FB C605[105B0100]02  <1>       mov    byte [DirBuff_ValidData], 2
33363 0000B702 E88EEFFFFF        <1>       call   save_directory_buffer
33364 0000B707 72B3              <1>       jc     short loc_createfile_anc_retn
33365                            <1>
33366                            <1> loc_createfile_save_added_subdir_cluster:
33367 0000B709 A1[3C5F0100]      <1>       mov    eax, [createfile_LastDirCluster]
```

```
33368 0000B70E 8B0D[385F0100]      <1>       mov    ecx, [createfile_FFCluster]
33369 0000B714 E858050000          <1>       call   update_cluster
33370 0000B719 7304                <1>       jnc    short loc_createfile_save_fat_buffer_0
33371 0000B71B 09C0                <1>       or     eax, eax ; EAX = 0 -> cluster value is 0 or eocc
33372 0000B71D 751A                <1>       jnz    short loc_createfile_save_fat_buffer_stc_retn
33373                              <1>
33374                              <1> loc_createfile_save_fat_buffer_0:
33375 0000B71F A1[385F0100]        <1>       mov    eax, [createfile_FFCluster]
33376 0000B724 A3[3C5F0100]        <1>       mov    [createfile_LastDirCluster], eax
33377 0000B729 B9FFFFFF0F          <1>       mov    ecx, 0FFFFFFFh ; 28 bit
33378 0000B72E E83E050000          <1>       call   update_cluster
33379 0000B733 7306                <1>       jnc    short loc_createfile_save_fat_buffer_1
33380 0000B735 09C0                <1>       or     eax, eax ; Was it free cluster
33381 0000B737 7402                <1>       jz     short loc_createfile_save_fat_buffer_1
33382                              <1>
33383                              <1> loc_createfile_save_fat_buffer_stc_retn:
33384 0000B739 F9                  <1>       stc
33385                              <1> loc_createfile_save_fat_buffer_retn:
33386                              <1> loc_createfile_gffc_2_stc_retn:
33387 0000B73A C3                  <1>       retn
33388                              <1>
33389                              <1> loc_createfile_save_fat_buffer_1:
33390                              <1>       ; byte [FAT_BuffValidData] = 2
33391 0000B73B E8EE070000          <1>       call   save_fat_buffer
33392 0000B740 72F8                <1>       jc     short loc_createfile_save_fat_buffer_retn
33393                              <1>
33394 0000B742 803D[065B0100]01    <1>       cmp    byte [FAT_ClusterCounter], 1
33395 0000B749 7222                <1>       jb     short loc_createfile_save_fat_buffer_2
33396                              <1>
33397                              <1>       ; ESI = Logical DOS Drive Description Table address
33398 0000B74B A1[065B0100]        <1>       mov    eax, [FAT_ClusterCounter]
33399                              <1>
33400 0000B750 C605[065B0100]00    <1>       mov    byte [FAT_ClusterCounter], 0 ; 21/03/2016
33401                              <1>
33402 0000B757 66BB01FF            <1>       mov    bx, 0FF01h ; add free clusters
33403 0000B75B E863080000          <1>       call   calculate_fat_freespace
33404                              <1>
33405                              <1>       ;inc   eax ; 0FFFFFFFFh -> 0 ; recalculation is needed!
33406                              <1>       ;jnz   short loc_createfile_save_fat_buffer_2
33407                              <1>
33408                              <1>       ; ecx > 0 -> Recalculation is needed
33409 0000B760 09C9                <1>       or     ecx, ecx
33410 0000B762 7409                <1>       jz     short loc_createfile_save_fat_buffer_2
33411                              <1>
33412 0000B764 66BB00FF            <1>       mov    bx, 0FF00h ; ; recalculate free space
33413 0000B768 E856080000          <1>       call   calculate_fat_freespace
33414                              <1>
33415                              <1> loc_createfile_save_fat_buffer_2:
33416                              <1>       ;call  update_parent_dir_lmdt
33417                              <1>
33418                              <1> loc_createfile_gffc_2:
33419 0000B76D E82C040000          <1>       call   get_first_free_cluster
33420 0000B772 72C6                <1>       jc     short loc_createfile_gffc_2_stc_retn
33421                              <1>
33422 0000B774 A3[385F0100]        <1>       mov    [createfile_FFCluster], eax
33423                              <1>
33424 0000B779 A1[3C5F0100]        <1>       mov    eax, [createfile_LastDirCluster]
33425                              <1>
33426 0000B77E E8AA030000          <1>       call   load_FAT_sub_directory
33427 0000B783 72B5                <1>       jc     short loc_createfile_gffc_2_stc_retn
33428                              <1>
33429 0000B785 BF00000800          <1>       mov    edi, Directory_Buffer
33430                              <1>
33431 0000B78A 6629DB              <1>       sub    bx, bx ; directory entry index/number = 0
33432                              <1>
33433 0000B78D 56                  <1>       push   esi ; * ; 23/03/2016
33434                              <1>
33435                              <1> loc_createfile_set_ff_dir_entry:
33436 0000B78E 66891D[4A5F0100]    <1>       mov    [createfile_DirIndex], bx
33437                              <1>
33438                              <1>         ; EDI = Directory entry address
33439 0000B795 8B35[2C5F0100]      <1>       mov    esi, [createfile_Name_Offset]
33440 0000B79B A1[385F0100]        <1>       mov    eax, [createfile_FFCluster]
33441 0000B7A0 A3[405F0100]        <1>       mov    [createfile_Cluster], eax ; 24/03/2016
33442 0000B7A5 B5FF                <1>       mov    ch, 0FFh
33443 0000B7A7 8A0D[485F0100]      <1>         mov  cl, [createfile_attrib] ; file attributes
33444                              <1>       ; CH > 0 -> File size is in [EBX]
33445 0000B7AD BB[345F0100]        <1>       mov    ebx, createfile_size
33446                              <1>
33447 0000B7B2 E801EEFFFF          <1>       call   make_directory_entry
33448                              <1>
33449 0000B7B7 5E                  <1>       pop    esi ; * ; ESI = Logical Dos Drv Desc. Table address
33450                              <1>
33451 0000B7B8 C605[105B0100]02    <1>       mov    byte [DirBuff_ValidData], 2
33452 0000B7BF E8D1EEFFFF          <1>       call   save_directory_buffer
33453 0000B7C4 7221                <1>       jc     short loc_createfile_set_ff_dir_entry_retn
33454                              <1>
33455 0000B7C6 C605[535F0100]01    <1>       mov    byte [createfile_UpdatePDir], 1 ; 31/03/2016
33456                              <1>
33457                              <1> loc_createfile_get_set_write_file_cluster:
33458 0000B7CD A1[345F0100]        <1>       mov    eax, [createfile_size]
33459 0000B7D2 09C0                <1>       or     eax, eax
33460 0000B7D4 7570                <1>       jnz    short loc_createfile_get_set_wfc_cont
33461 0000B7D6 40                  <1>       inc    eax
33462                              <1>       ; 23/03/2016
33463 0000B7D7 0FB61D[495F0100]    <1>       movzx  ebx, byte [createfile_SecPerClust]
33464                              <1>       ;movzx ecx, word [esi+LD_BPB+BytesPerSec] ; 512
33465 0000B7DE 0FB70D[505F0100]    <1>         movzx   ecx, word [createfile_BytesPerSec] ; 512
33466 0000B7E5 EB7C                <1>       jmp    loc_createfile_set_cluster_count
33467                              <1>
33468                              <1> loc_createfile_set_ff_dir_entry_retn:
33469 0000B7E7 C3                  <1>       retn
33470                              <1>
```

```
33471                                  <1> loc_createfile_write_fcluster_to_disk:
33472 0000B7E8 034668                  <1>       add   eax, [esi+LD_DATABegin] ; convert to physical address
33473 0000B7EB BB00000700              <1>       mov   ebx, Cluster_Buffer
33474                                  <1>       ; ESI = Logical DOS Drv. Desc. Tbl. address
33475                                  <1>       ; EAX = Disk address
33476                                  <1>       ; EBX = Sector Buffer
33477                                  <1>       ; ECX = sectors per cluster
33478 0000B7F0 E8D4390000              <1>       call  disk_write
33479 0000B7F5 7211                    <1>       jc    short loc_createfile_dsk_wr_err
33480                                  <1>
33481                                  <1> loc_createfile_update_fat_cluster:
33482                                  <1>       ; 21/03/2016
33483 0000B7F7 803D[525F0100]00        <1>       cmp   byte [createfile_wfc], 0
33484 0000B7FE 7712                    <1>       ja    short loc_createfile_update_fat_cluster_n1
33485                                  <1>
33486 0000B800 FE05[525F0100]          <1>       inc   byte [createfile_wfc] ; 1
33487 0000B806 EB24                    <1>       jmp   short loc_createfile_update_fat_cluster_n2
33488                                  <1>
33489                                  <1> loc_createfile_dsk_wr_err:
33490                                  <1>       ; 16/10/2016 (1Dh -> 18)
33491                                  <1>       ; 23/03/2016
33492 0000B808 B812000000              <1>       mov   eax, 18 ; Drive not ready or write error !
33493 0000B80D E9BD000000              <1>       jmp   loc_createfile_stc_retn
33494                                  <1>
33495                                  <1> loc_createfile_update_fat_cluster_n1:
33496 0000B812 A1[445F0100]            <1>       mov   eax, [createfile_PCluster]
33497 0000B817 8B0D[405F0100]          <1>       mov   ecx, [createfile_Cluster]
33498 0000B81D E84F040000              <1>       call  update_cluster
33499 0000B822 7308                    <1>       jnc   short loc_createfile_update_fat_cluster_n2
33500 0000B824 09C0                    <1>       or    eax, eax ; EAX = 0 -> cluster value is 0 or eocc
33501 0000B826 0F85A3000000            <1>       jnz   loc_createfile_stc_retn
33502                                  <1>
33503                                  <1> loc_createfile_update_fat_cluster_n2:
33504 0000B82C A1[405F0100]            <1>         mov  eax, [createfile_Cluster]
33505 0000B831 B9FFFFFF0F              <1>       mov   ecx, 0FFFFFFFh
33506 0000B836 E836040000              <1>       call  update_cluster
33507 0000B83B 734E                    <1>       jnc   short loc_createfile_save_fat_buffer_3
33508 0000B83D 09C0                    <1>       or    eax, eax ; EAX = 0 -> cluster value is 0 or eocc
33509 0000B83F 744A                    <1>       jz    short loc_createfile_save_fat_buffer_3
33510                                  <1>
33511                                  <1> loc_createfile_upd_fat_fcluster_stc_retn:
33512 0000B841 E989000000              <1>       jmp   loc_createfile_stc_retn
33513                                  <1>
33514                                  <1> loc_createfile_get_set_wfc_cont:
33515                                  <1>       ;movzx ecx, word [esi+LD_BPB+BytesPerSec] ; 512
33516 0000B846 0FB70D[505F0100]        <1>       movzx  ecx, word [createfile_BytesPerSec] ; 512
33517 0000B84D 01C8                    <1>       add   eax, ecx
33518 0000B84F 48                      <1>       dec   eax  ; add eax, 511
33519 0000B850 29D2                    <1>       sub   edx, edx
33520 0000B852 F7F1                    <1>       div   ecx
33521 0000B854 0FB61D[495F0100]        <1>       movzx  ebx, byte [createfile_SecPerClust]
33522 0000B85B 01D8                    <1>       add   eax, ebx
33523 0000B85D 48                      <1>       dec   eax  ; add eax, SecPerClust - 1
33524 0000B85E 6631D2                  <1>       xor   dx, dx
33525 0000B861 F7F3                    <1>       div   ebx
33526                                  <1>
33527                                  <1> loc_createfile_set_cluster_count:
33528 0000B863 A3[4C5F0100]            <1>       mov   [createfile_CCount], eax
33529                                  <1>
33530 0000B868 BF00000700              <1>       mov   edi, Cluster_Buffer
33531 0000B86D 89C8                    <1>       mov   eax, ecx ; Bytes per Sector
33532 0000B86F F7E3                    <1>       mul   ebx ; Sectors per Cluster
33533                                  <1>       ; EAX = Bytes per Cluster
33534 0000B871 89C1                    <1>       mov   ecx, eax
33535 0000B873 C1E902                  <1>       shr   ecx, 2 ; dword count
33536 0000B876 31C0                    <1>       xor   eax, eax
33537 0000B878 F3AB                    <1>       rep   stosd ; clear cluster buffer
33538                                  <1>
33539 0000B87A A1[405F0100]            <1>       mov   eax, [createfile_Cluster] ; 24/03/2016
33540                                  <1>
33541 0000B87F 89D9                    <1>       mov   ecx, ebx
33542                                  <1>
33543                                  <1> loc_createfile_get_set_wf_fclust_cont:
33544 0000B881 83E802                  <1>       sub   eax, 2
33545 0000B884 F7E1                    <1>       mul   ecx
33546                                  <1>       ; EAX = Logical DOS disk address (offset)
33547 0000B886 E95DFFFFFF              <1>         jmp     loc_createfile_write_fcluster_to_disk
33548                                  <1>
33549                                  <1> loc_createfile_save_fat_buffer_3:
33550                                  <1>       ; byte [FAT_BuffValidData] = 2
33551 0000B88B E89E060000              <1>       call  save_fat_buffer
33552 0000B890 723D                    <1>       jc    loc_createfile_stc_retn
33553                                  <1>
33554                                  <1>       ; 21/03/2016
33555 0000B892 803D[065B0100]01        <1>       cmp   byte [FAT_ClusterCounter], 1
33556 0000B899 721B                    <1>       jb    short loc_createfile_save_fat_buffer_4
33557                                  <1>
33558                                  <1>       ; ESI = Logical DOS Drive Description Table address
33559 0000B89B A1[065B0100]            <1>       mov   eax, [FAT_ClusterCounter]
33560 0000B8A0 66BB01FF                <1>       mov   bx, 0FF01h ; add free clusters
33561 0000B8A4 E81A070000              <1>       call  calculate_fat_freespace
33562                                  <1>
33563                                  <1>       ;inc   eax ; 0FFFFFFFFh -> 0 ; recalculation is needed!
33564                                  <1>       ;jnz   short loc_createfile_save_fat_buffer_4
33565                                  <1>
33566                                  <1>       ; ecx > 0 -> Recalculation is needed
33567 0000B8A9 09C9                    <1>       or    ecx, ecx
33568 0000B8AB 7409                    <1>       jz    short loc_createfile_save_fat_buffer_4
33569                                  <1>
33570 0000B8AD 66BB00FF                <1>       mov   bx, 0FF00h ; ; recalculate free space
33571 0000B8B1 E80D070000              <1>       call  calculate_fat_freespace
33572                                  <1>
33573                                  <1> loc_createfile_save_fat_buffer_4:
```

```
33574 0000B8B6 FF0D[4C5F0100]      <1>       dec    dword [createfile_CCount]
33575                              <1>       ;jz    short loc_createfile_upd_dir_modif_date_time
33576 0000B8BC 743F                <1>       jz     short loc_createfile_stc_retn_cc ; 31/03/2016
33577                              <1>
33578                              <1> loc_createfile_get_set_write_next_cluster:
33579 0000B8BE E8DB020000          <1>       call   get_first_free_cluster
33580 0000B8C3 720A                <1>       jc     short loc_createfile_stc_retn
33581                              <1>
33582                              <1> loc_createfile_get_set_write_next_cluster_1:
33583 0000B8C5 83F8FF              <1>       cmp    eax, 0FFFFFFFFh
33584 0000B8C8 7213                <1>       jb     short loc_createfile_get_set_write_next_cluster_2
33585                              <1>
33586                              <1> loc_createfile_wnc_insufficient_disk_space:
33587 0000B8CA B827000000          <1>       mov    eax, 27h ; Insufficient disk space
33588                              <1>
33589                              <1> loc_createfile_stc_retn:
33590 0000B8CF 803D[525F0100]01    <1>       cmp    byte [createfile_wfc], 1
33591 0000B8D6 7324                <1>       jnb    short loc_createfile_err_retn
33592 0000B8D8 C3                  <1>       retn
33593                              <1>
33594                              <1> loc_createfile_wnc_inv_format_retn:
33595                              <1>       ;mov   eax, 28
33596 0000B8D9 B01C                <1>       mov    al, 28 ; Invalid format
33597 0000B8DB EBF2                <1>       jmp    short loc_createfile_stc_retn
33598                              <1>
33599                              <1> loc_createfile_get_set_write_next_cluster_2:
33600 0000B8DD 83F802              <1>       cmp    eax, 2
33601 0000B8E0 72F7                <1>       jb     short loc_createfile_wnc_inv_format_retn
33602                              <1>
33603                              <1> loc_createfile_get_set_write_next_cluster_3:
33604 0000B8E2 8B0D[405F0100]      <1>       mov    ecx, [createfile_Cluster]
33605 0000B8E8 A3[405F0100]        <1>       mov    [createfile_Cluster], eax
33606 0000B8ED 890D[445F0100]      <1>       mov    [createfile_PCluster], ecx
33607 0000B8F3 0FB60D[495F0100]    <1>       movzx  ecx, byte [createfile_SecPerClust]
33608 0000B8FA EB85                <1>       jmp    short loc_createfile_get_set_wf_fclust_cont
33609                              <1>
33610                              <1> loc_createfile_err_retn:
33611 0000B8FC F9                  <1>       stc
33612                              <1>
33613                              <1> ;loc_createfile_upd_dir_modif_date_time:
33614                              <1> loc_createfile_stc_retn_cc: ; 31/03/2016
33615 0000B8FD 9C                  <1>       pushf  ; cpu is here for an error return or completion
33616 0000B8FE 50                  <1>       push   eax ; error code if cf = 1
33617                              <1>
33618                              <1>       ;call  update_parent_dir_lmdt
33619                              <1>
33620                              <1> ;loc_createfile_stc_retn_cc:
33621 0000B8FF A1[065B0100]        <1>       mov    eax, [FAT_ClusterCounter]
33622 0000B904 09C0                <1>       or     eax, eax
33623 0000B906 741A                <1>       jz     short loc_createfile_stc_retn_pop_eax
33624 0000B908 8A3D[E6520100]      <1>       mov    bh, [Current_Drv]
33625 0000B90E B301                <1>       mov    bl, 01h ; BL = 1 -> add clusters
33626                              <1>       ; NOTE: EAX value will be added to Free Cluster Count
33627                              <1>       ; (If EAX value is negative, Free Cluster Count will be decreased)
33628 0000B910 E8AE060000          <1>       call   calculate_fat_freespace
33629                              <1>         ; ESI = Logical DOS Drive Description Table Address
33630                              <1>         ;jc  short loc_createfile_stc_retn_pop_eax_cf
33631 0000B915 21C9                <1>       and    ecx, ecx ; cx = 0 -> valid free sector count
33632 0000B917 7409                <1>       jz     short loc_createfile_stc_retn_pop_eax
33633                              <1>
33634                              <1> loc_createfile_stc_retn_recalc_FAT_freespace:
33635 0000B919 66BB00FF            <1>       mov    bx, 0FF00h ; bh = 0FFh ->
33636                              <1>       ; ESI = Logical DOS Drv DT Addr
33637                              <1>       ; BL = 0 -> Recalculate
33638 0000B91D E8A1060000          <1>       call   calculate_fat_freespace
33639                              <1>
33640                              <1> loc_createfile_stc_retn_pop_eax:
33641 0000B922 58                  <1>       pop    eax
33642 0000B923 9D                  <1>       popf
33643 0000B924 7218                <1>       jc     short loc_createfile_retn
33644                              <1>
33645                              <1> loc_createfile_retn_fcluster:
33646 0000B926 A1[385F0100]        <1>       mov    eax, [createfile_FFCluster]
33647 0000B92B BB[345F0100]        <1>       mov    ebx, createfile_size
33648                              <1>       ;movzx ecx, byte [esi+LD_BPB+SecPerClust]
33649 0000B930 0FB60D[495F0100]    <1>       movzx  ecx, byte [createfile_SecPerClust] ; 23/03/2016
33650 0000B937 0FB715[4A5F0100]    <1>       movzx  edx, word [createfile_DirIndex]
33651                              <1>
33652                              <1> loc_createfile_retn:
33653 0000B93E C3                  <1>       retn
33654                              <1>
33655                              <1> create_fs_file:
33656                              <1>       ; temporary (21/03/2016)
33657 0000B93F C3                  <1>       retn
33658                              <1>
33659                              <1> delete_fs_file:
33660                              <1>       ; temporary (28/02/2016)
33661 0000B940 C3                  <1>       retn
33662                              <1>
33663                              <1> rename_fs_file_or_directory:
33664 0000B941 C3                  <1>       retn
33665                              <1>
33666                              <1> make_fs_directory:
33667                              <1>       ; temporary (21/02/2016)
33668 0000B942 C3                  <1>       retn
33669                              <1>
33670                              <1> add_new_fs_section:
33671                              <1>       ; temporary (11/03/2016)
33672 0000B943 C3                  <1>       retn
33673                              <1>
33674                              <1> delete_fs_directory_entry:
33675                              <1>       ; temporary (11/03/2016)
33676 0000B944 C3                  <1>       retn
```

```
                                <1>
                                <1> csftdf2_read_fs_file_sectors:
                                <1>       ; temporary (19/03/2016)
0000B945 C3                     <1>       retn
                                <1>
                                <1> csftdf2_write_fs_file_sectors:
                                <1>       ; temporary (19/03/2016)
0000B946 C3                     <1>       retn
                                    %include 'trdosk5.s' ; 24/01/2016
                                <1> ; ****************************************************************************
                                <1> ; TRDOS386.ASM (TRDOS 386 Kernel - v2.0.0) - File System Procedures : trdosk5s
                                <1> ; ----------------------------------------------------------------------------
                                <1> ; Last Update: 23/10/2016
                                <1> ; ----------------------------------------------------------------------------
                                <1> ; Beginning: 24/01/2016
                                <1> ; ----------------------------------------------------------------------------
                                <1> ; Assembler: NASM version 2.11 (trdos386.s)
                                <1> ; ----------------------------------------------------------------------------
                                <1> ; Derived from TRDOS Operating System v1.0 (8086) source code by Erdogan Tan
                                <1> ; DRV_FAT.ASM (21/08/2011)
                                <1> ; ****************************************************************************
                                <1> ; DRV_FAT.ASM (c) 2005-2011 Erdogan TAN [ 07/07/2009 ] Last Update: 21/08/2011
                                <1>
                                <1> get_next_cluster:
                                <1>       ; 15/10/2016
                                <1>       ; 23/03/2016
                                <1>       ; 01/02/2016 (TRDOS 386 =  TRDOS v2.0)
                                <1>       ; 05/07/2011
                                <1>       ; 07/07/2009
                                <1>       ; 2005
                                <1>       ; INPUT ->
                                <1>       ;     EAX = Cluster Number (32 bit)
                                <1>       ;     ESI = Logical DOS Drive Parameters Table
                                <1>       ; OUTPUT ->
                                <1>       ;     cf = 0 -> No Error, EAX valid
                                <1>       ;     cf = 1 & EAX = 0 -> End Of Cluster Chain
                                <1>       ;     cf = 1 & EAX > 0 -> Error
                                <1>       ;     ECX = Current/Previous cluster (if CF = 0)
                                <1>       ;     EAX = Next Cluster Number (32 bit)
                                <1>       ;
                                <1>       ; (Modified registers: EAX, ECX, EBX, EDX)
                                <1>
0000B947 A3[FA5A0100]           <1>       mov   [FAT_CurrentCluster], eax
                                <1> check_next_cluster_fat_type:
0000B94C 29D2                   <1>       sub   edx, edx ; 0
0000B94E 807E0302               <1>         cmp     byte [esi+LD_FATType], 2
0000B952 7250                   <1>       jb    short get_FAT12_next_cluster
0000B954 0F87AF000000           <1>         ja      get_FAT32_next_cluster
                                <1> get_FAT16_next_cluster:
0000B95A BB00030000             <1>       mov   ebx, 300h ;768
0000B95F F7F3                   <1>       div   ebx
                                <1>       ; EAX = Count of 3 FAT sectors
                                <1>       ; EDX = Cluster Offset (< 768)
0000B961 66D1E2                 <1>       shl   dx, 1 ; Multiply by 2
0000B964 89D3                   <1>       mov   ebx, edx ; Byte Offset
0000B966 81C3001C0900           <1>       add   ebx, FAT_Buffer
0000B96C 66BA0300               <1>       mov   dx, 3
0000B970 F7E2                   <1>       mul   edx
                                <1>       ; EAX = FAT Sector (<= 256)
                                <1>       ; EDX = 0
0000B972 8A0E                   <1>       mov   cl, [esi+LD_Name]
0000B974 803D[FE5A0100]00       <1>       cmp   byte [FAT_BuffValidData], 0
0000B97B 0F86CC000000           <1>         jna     load_FAT_sectors0
0000B981 3A0D[FF5A0100]         <1>       cmp   cl, [FAT_BuffDrvName]
0000B987 0F85C0000000           <1>         jne     load_FAT_sectors0
0000B98D 3B05[025B0100]         <1>       cmp   eax, [FAT_BuffSector]
0000B993 0F85BA000000           <1>         jne     load_FAT_sectors1
                                <1>       ;movzx eax, word [ebx]
0000B999 668B03                 <1>       mov   ax, [ebx]
                                <1>       ; 01/02/2016
                                <1>       ; DRV_FAT.ASM (21/08/2011) had a FATal bug here !
                                <1>       ; (cmp ah, 0Fh) ! (ax >= FF7h)
                                <1>       ; (how can i do a such mistake!?)
                                <1>       ;cmp  al, 0F7h
                                <1>       ;jb   short loc_pass_gnc_FAT16_eoc_check
                                <1>       ;cmp  ah, 0FFh
                                <1>       ;jb   short loc_pass_gnc_FAT16_eoc_check
0000B99C 6683F8F7               <1>       cmp   ax, 0FFF7h
0000B9A0 725A                   <1>       jb    short loc_pass_gnc_FAT16_eoc_check
                                <1>       ; ax >= FFF7h (cluster 0002h to FFF6h is valid, in use)
0000B9A2 EB56                   <1>       jmp   short loc_pass_gnc_FAT16_eoc_check_xor_eax
                                <1>
                                <1> get_FAT12_next_cluster:
0000B9A4 BB00040000             <1>       mov   ebx, 400h ;1024
0000B9A9 F7F3                   <1>       div   ebx
                                <1>       ; EAX = Count of 3 FAT sectors
                                <1>       ; EDX = Cluster Offset (< 1024)
0000B9AB 6650                   <1>       push  ax
0000B9AD 66B80300               <1>       mov   ax, 3
0000B9B1 66F7E2                 <1>       mul   dx    ; Multiply by 3
0000B9B4 66D1E8                 <1>       shr   ax, 1 ; Divide by 2
0000B9B7 6689C3                 <1>         mov bx, ax      ; Byte Offset
0000B9BA 81C3001C0900           <1>       add   ebx, FAT_Buffer
0000B9C0 6658                   <1>       pop   ax
0000B9C2 66BA0300               <1>       mov   dx, 3
0000B9C6 F7E2                   <1>       mul   edx
                                <1>       ; EAX = FAT Sector (<= 12)
                                <1>       ; EDX = 0
0000B9C8 8A0E                   <1>       mov   cl, [esi+LD_Name]
0000B9CA 803D[FE5A0100]00       <1>       cmp   byte [FAT_BuffValidData], 0
0000B9D1 767A                   <1>       jna   short load_FAT_sectors0
0000B9D3 3A0D[FF5A0100]         <1>       cmp   cl, [FAT_BuffDrvName]
0000B9D9 7572                   <1>       jne   short load_FAT_sectors0
```

```
33780 0000B9DB 3B05[025B0100]      <1>       cmp    eax, [FAT_BuffSector]
33781 0000B9E1 7570                <1>       jne    short load_FAT_sectors1
33782 0000B9E3 A1[FA5A0100]        <1>       mov    eax, [FAT_CurrentCluster]
33783 0000B9E8 66D1E8              <1>       shr    ax, 1
33784                              <1>       ;movzx eax, word [ebx]
33785 0000B9EB 668B03              <1>       mov    ax, [ebx]
33786 0000B9EE 7314                <1>       jnc    short get_FAT12_nc_even
33787 0000B9F0 66C1E804            <1>       shr    ax, 4
33788                              <1> loc_gnc_fat12_eoc_check:
33789                              <1>       ;cmp   al, 0F7h
33790                              <1>       ;jb    short loc_pass_gnc_FAT16_eoc_check
33791                              <1>       ;cmp   ah, 0Fh
33792                              <1>       ;jb    short loc_pass_gnc_FAT16_eoc_check
33793 0000B9F4 663DF70F            <1>       cmp    ax, 0FF7h
33794 0000B9F8 7202                <1>       jb     short loc_pass_gnc_FAT16_eoc_check
33795                              <1>       ; ax >= FF7h (cluster 0002h to FF6h is valid, in use)
33796                              <1>
33797                              <1> loc_pass_gnc_FAT16_eoc_check_xor_eax:
33798 0000B9FA 31C0                <1>       xor    eax, eax ; 0
33799                              <1> loc_pass_gnc_FAT16_eoc_check:
33800                              <1> loc_pass_gnc_FAT32_eoc_check:
33801 0000B9FC 8B0D[FA5A0100]      <1>       mov    ecx, [FAT_CurrentCluster]
33802 0000BA02 F5                  <1>       cmc
33803 0000BA03 C3                  <1>       retn
33804                              <1>
33805                              <1> get_FAT12_nc_even:
33806 0000BA04 80E40F              <1>       and    ah, 0Fh
33807 0000BA07 EBEB                <1>       jmp    short loc_gnc_fat12_eoc_check
33808                              <1>
33809                              <1> get_FAT32_next_cluster:
33810 0000BA09 BB80010000          <1>       mov    ebx, 180h ;384
33811 0000BA0E F7F3                <1>       div    ebx
33812                              <1>       ; EAX = Count of 3 FAT sectors
33813                              <1>       ; EDX = Cluster Offset (< 384)
33814 0000BA10 66C1E202            <1>       shl    dx, 2  ; Multiply by 4
33815 0000BA14 89D3                <1>       mov    ebx, edx ; Byte Offset
33816 0000BA16 81C3001C0900        <1>       add    ebx, FAT_Buffer
33817 0000BA1C 66BA0300            <1>       mov    dx, 3
33818 0000BA20 F7E2                <1>       mul    edx
33819                              <1>         ; EAX = FAT Sector (<= 2097152) ; (FFFFFF7h * 4) / 512
33820                              <1>       ;      for 32KB cluster size:
33821                              <1>       ;      EAX <= 1024 = (4GB / 32KB) * 4) / 512
33822                              <1>       ; EDX = 0
33823 0000BA22 8A0E                <1>       mov    cl, [esi+LD_Name]
33824 0000BA24 803D[FE5A0100]00    <1>       cmp    byte [FAT_BuffValidData], 0
33825 0000BA2B 7620                <1>       jna    short load_FAT_sectors0
33826 0000BA2D 3A0D[FF5A0100]      <1>       cmp    cl, [FAT_BuffDrvName]
33827 0000BA33 7518                <1>       jne    short load_FAT_sectors0
33828 0000BA35 3B05[025B0100]      <1>       cmp    eax, [FAT_BuffSector] ; 0, 3, 6, 9 ...
33829 0000BA3B 7516                <1>       jne    short load_FAT_sectors1
33830 0000BA3D 8B03                <1>       mov    eax, [ebx]
33831 0000BA3F 25FFFFFF0F          <1>       and    eax, 0FFFFFFFh ; 28 bit Cluster
33832 0000BA44 3DF7FFFF0F          <1>       cmp    eax, 0FFFFFF7h
33833 0000BA49 72B1                <1>       jb     short loc_pass_gnc_FAT32_eoc_check
33834                              <1>       ; eax >= FFFFFF7h (cluster 0002h to FFFFFF6h is valid)
33835 0000BA4B EBAD                <1>       jmp    short loc_pass_gnc_FAT16_eoc_check_xor_eax
33836                              <1>
33837                              <1> load_FAT_sectors0:
33838 0000BA4D 880D[FF5A0100]      <1>       mov    [FAT_BuffDrvName], cl
33839                              <1> load_FAT_sectors1:
33840 0000BA53 A3[025B0100]        <1>       mov    [FAT_BuffSector], eax
33841 0000BA58 89C3                <1>       mov    ebx, eax
33842 0000BA5A 034660              <1>         add     eax, [esi+LD_FATBegin]
33843 0000BA5D 807E0302            <1>       cmp    byte [esi+LD_FATType], 2
33844 0000BA61 7706                <1>         ja      short load_FAT_sectors3
33845 0000BA63 0FB74E1C            <1>       movzx  ecx, word [esi+LD_BPB+BPB_FATSz16]
33846 0000BA67 EB03                <1>       jmp    short load_FAT_sectors4
33847                              <1> load_FAT_sectors3:
33848 0000BA69 8B4E2A              <1>       mov    ecx, [esi+LD_BPB+BPB_FATSz32]
33849                              <1> load_FAT_sectors4:
33850 0000BA6C 29D9                <1>       sub    ecx, ebx ; [FAT_BuffSector]
33851 0000BA6E 83F903              <1>         cmp     ecx, 3
33852 0000BA71 7605                <1>         jna     short load_FAT_sectors5
33853 0000BA73 B903000000          <1>       mov    ecx, 3
33854                              <1> load_FAT_sectors5:
33855 0000BA78 BB001C0900          <1>       mov    ebx, FAT_Buffer
33856 0000BA7D E856370000          <1>       call   disk_read
33857 0000BA82 730D                <1>       jnc    short load_FAT_sectors_ok
33858                              <1>       ; 15/10/2016 (15h -> 17)
33859                              <1>       ; 23/03/2016 (15h)
33860 0000BA84 B811000000          <1>       mov    eax, 17 ; Drive not ready or read error
33861 0000BA89 C605[FE5A0100]00    <1>       mov    byte [FAT_BuffValidData], 0
33862 0000BA90 C3                  <1>       retn
33863                              <1> load_FAT_sectors_ok:
33864 0000BA91 C605[FE5A0100]01    <1>       mov    byte [FAT_BuffValidData], 1
33865 0000BA98 A1[FA5A0100]        <1>       mov    eax, [FAT_CurrentCluster]
33866 0000BA9D E9AAFEFFFF          <1>         jmp     check_next_cluster_fat_type
33867                              <1>
33868                              <1> load_FAT_root_directory:
33869                              <1>       ; 23/10/2016
33870                              <1>       ; 15/10/2016
33871                              <1>       ; 07/02/2016
33872                              <1>       ; 02/02/2016
33873                              <1>       ; 01/02/2016 (TRDOS 386 =  TRDOS v2.0)
33874                              <1>       ; 21/05/2011
33875                              <1>       ; 22/08/2009
33876                              <1>       ;
33877                              <1>       ; INPUT ->
33878                              <1>       ;      ESI = Logical DOS Drive Description Table
33879                              <1>       ; OUTPUT ->
33880                              <1>       ;      cf = 1 -> Root directory could not be loaded
33881                              <1>       ;           EAX > 0 -> Error number
33882                              <1>       ;      cf = 0 -> EAX = 0
```

```
33883                                  <1>       ;     ECX = Directory buffer size in sectors (CL)
33884                                  <1>       ;     EBX = Directory buffer address
33885                                  <1>       ;     NOTE: DirBuffer_Size is in bytes ! (word)
33886                                  <1>       ;
33887                                  <1>       ; (Modified registers: EAX, ECX, EBX, EDX)
33888                                  <1>
33889                                  <1>       ; NOTE: Only for FAT12 and FAT16 file systems !
33890                                  <1>       ; (FAT32 fs root dir must be loaded as sub directory)
33891                                  <1>
33892 0000BAA2 8A1E                   <1>       mov   bl, [esi+LD_Name]
33893 0000BAA4 8A7E03                 <1>       mov   bh, [esi+LD_FATType]
33894                                  <1>
33895                                  <1>       ;mov  [DirBuff_DRV], bl
33896                                  <1>       ;mov  [DirBuff_FATType], bh
33897 0000BAA7 66891D[0E5B0100]       <1>       mov   [DirBuff_DRV], bx
33898                                  <1>
33899                                  <1>       ;cmp  bh, 2
33900                                  <1>       ;ja   short load_FAT32_root_dir0 ; FAT32 root dir
33901                                  <1>
33902                                  <1> load_FAT_root_dir0: ; 23/10/2016
33903 0000BAAE 0FB75617               <1>       movzx  edx, word [esi+LD_BPB+RootDirEnts]
33904                                  <1>
33905                                  <1>       ;or   dx, dx ; 0 for FAT32 file systems
33906                                  <1>       ;jz   short load_FAT32_root_dir0 ; FAT32 root dir
33907                                  <1>
33908 0000BAB2 6681FA0002             <1>       cmp   dx, 512 ; Number of Root Dir Entries
33909 0000BAB7 7414                   <1>       je    short lrd_mov_ecx_32
33910 0000BAB9 89D0                   <1>       mov   eax, edx
33911                                  <1>       ; 23/10/2016
33912 0000BABB 89C1                   <1>       mov   ecx, eax
33913 0000BABD 6683C10F               <1>       add   cx, 15 ; round up
33914 0000BAC1 66C1E904               <1>       shr   cx, 4  ; 16 entries per sector (512/32)
33915                                  <1>       ; ecx = Root directory size in sectors
33916 0000BAC5 66C1E005               <1>       shl   ax, 5 ; Root directory size in bytes
33917 0000BAC9 664A                   <1>       dec   dx    ; Last entry number of root dir
33918                                  <1>       ; cx = Dir Buffer sector count
33919 0000BACB EB0B                   <1>       jmp   short lrd_check_dir_buffer
33920                                  <1>
33921                                  <1> lrd_mov_ecx_32:
33922 0000BACD B920000000             <1>       mov   ecx, 32
33923 0000BAD2 664A                   <1>       dec   dx ; 511
33924 0000BAD4 66B80040               <1>       mov   ax, 32*512
33925                                  <1>
33926                                  <1> lrd_check_dir_buffer:
33927 0000BAD8 29DB                   <1>       sub   ebx, ebx ; 0
33928 0000BADA 881D[105B0100]         <1>       mov   [DirBuff_ValidData], bl ; 0
33929 0000BAE0 668915[135B0100]       <1>       mov   [DirBuff_LastEntry], dx
33930 0000BAE7 891D[155B0100]         <1>       mov   [DirBuff_Cluster], ebx ; 0
33931 0000BAED 66A3[195B0100]         <1>       mov   [DirBuffer_Size], ax
33932                                  <1>
33933 0000BAF3 8B4664                 <1>       mov   eax, [esi+LD_ROOTBegin]
33934                                  <1> read_directory:
33935 0000BAF6 BB00000800             <1>       mov   ebx, Directory_Buffer
33936 0000BAFB 51                     <1>       push  ecx ; Directory buffer sector count
33937 0000BAFC 53                     <1>       push  ebx
33938 0000BAFD E8D6360000             <1>       call  disk_read
33939 0000BB02 5B                     <1>       pop   ebx
33940 0000BB03 720B                   <1>       jc    short load_DirBuff_error
33941                                  <1>
33942                                  <1> validate_DirBuff_and_return:
33943 0000BB05 59                     <1>       pop   ecx ; Number of loaded sectors
33944 0000BB06 C605[105B0100]01       <1>       mov   byte [DirBuff_ValidData], 1
33945 0000BB0D 31C0                   <1>       xor   eax, eax ; 0 = no error
33946 0000BB0F C3                     <1>       retn
33947                                  <1>
33948                                  <1> load_DirBuff_error:
33949 0000BB10 89C8                   <1>       mov   eax, ecx ; remaining sectors
33950 0000BB12 59                     <1>       pop   ecx ; sector count
33951 0000BB13 29C1                   <1>       sub   ecx, eax ; Number of loaded sectors
33952                                  <1>       ; 15/10/2016 (15h -> 17)
33953 0000BB15 B811000000             <1>       mov   eax, 17 ; DRV NOT READY OR READ ERROR !
33954 0000BB1A F9                     <1>       stc
33955 0000BB1B C3                     <1>         retn
33956                                  <1>
33957                                  <1> load_FAT32_root_directory:
33958                                  <1>       ; 02/02/2016 (TRDOS 386 =  TRDOS v2.0)
33959                                  <1>       ;
33960                                  <1>       ; INPUT ->
33961                                  <1>       ;     ESI = Logical DOS Drive Description Table
33962                                  <1>       ; OUTPUT ->
33963                                  <1>       ;     cf = 1 -> Root directory could not be loaded
33964                                  <1>       ;         EAX > 0 -> Error number
33965                                  <1>       ;     cf = 0 -> EAX = 0
33966                                  <1>       ;     ECX = Directory buffer size in sectors (CL)
33967                                  <1>       ;     EBX = Directory buffer address
33968                                  <1>       ;     NOTE: DirBuffer_Size is in bytes ! (word)
33969                                  <1>       ;
33970                                  <1>       ; (Modified registers: EAX, ECX, EBX, EDX)
33971                                  <1>
33972                                  <1>
33973 0000BB1C 8A1E                   <1>       mov   bl, [esi+LD_Name]
33974 0000BB1E 8A7E03                 <1>       mov   bh, [esi+LD_FATType]
33975                                  <1>
33976                                  <1>       ;mov  [DirBuff_DRV], bl
33977                                  <1>       ;mov  [DirBuff_FATType], bh
33978 0000BB21 66891D[0E5B0100]       <1>       mov   [DirBuff_DRV], bx
33979                                  <1>
33980                                  <1> load_FAT32_root_dir0:
33981 0000BB28 8B4632                 <1>       mov   eax, [esi+LD_BPB+FAT32_RootFClust]
33982 0000BB2B EB0C                   <1>       jmp   short load_FAT_sub_dir0
33983                                  <1>
33984                                  <1> load_FAT_sub_directory:
33985                                  <1>       ; 01/02/2016 (TRDOS 386 =  TRDOS v2.0)
```

```
                                <1>       ; 05/07/2011
                                <1>       ; 23/08/2009
                                <1>       ;
                                <1>       ; INPUT ->
                                <1>       ;     ESI = Logical DOS Drive Description Table
                                <1>       ;     EAX = Cluster Number
                                <1>       ; OUTPUT ->
                                <1>       ;     cf = 1 -> Sub directory could not be loaded
                                <1>       ;         EAX > 0 -> Error number
                                <1>       ;     cf = 0 -> EAX = 0
                                <1>       ;     ECX = Directory buffer size in sectors (CL)
                                <1>       ;     EBX = Directory buffer address
                                <1>       ;
                                <1>       ;     NOTE: DirBuffer_Size is in bytes ! (word)
                                <1>       ;
                                <1>       ; (Modified registers: EAX, ECX, EBX, EDX)
                                <1>
0000BB2D 8A1E                   <1>       mov   bl, [esi+LD_Name]
0000BB2F 8A7E03                 <1>       mov   bh, [esi+LD_FATType]
                                <1>
                                <1>       ;mov   [DirBuff_DRV], bl
                                <1>       ;mov   [DirBuff_FATType], bh
0000BB32 66891D[0E5B0100]       <1>       mov   [DirBuff_DRV], bx
                                <1>
                                <1> load_FAT_sub_dir0:
0000BB39 0FB64E13               <1>       movzx  ecx, byte [esi+LD_BPB+SecPerClust]
                                <1>
0000BB3D 882D[105B0100]         <1>       mov   [DirBuff_ValidData], ch ; 0
0000BB43 A3[155B0100]           <1>       mov   [DirBuff_Cluster], eax
                                <1>
0000BB48 0FB74611               <1>       movzx  eax, word [esi+LD_BPB+BytesPerSec]
0000BB4C F7E1                   <1>       mul   ecx
0000BB4E C1E805                 <1>       shr   eax, 5 ; directory entry count (dir size / 32)
0000BB51 6648                   <1>       dec   ax ; last entry
0000BB53 66A3[135B0100]         <1>       mov   [DirBuff_LastEntry], ax
                                <1>
0000BB59 A1[155B0100]           <1>       mov   eax, [DirBuff_Cluster]
0000BB5E 83E802                 <1>       sub   eax, 2
0000BB61 F7E1                   <1>       mul   ecx
0000BB63 034668                 <1>       add   eax, [esi+LD_DATABegin]
                                <1>       ; ecx = sector per cluster (dir buffer size = 32 sectors)
0000BB66 EB8E                   <1>       jmp   short read_directory
                                <1>
                                <1> ; DRV_FS.ASM
                                <1>
                                <1> load_current_FS_directory:
0000BB68 C3                     <1>       retn
                                <1> load_FS_root_directory:
0000BB69 C3                     <1>       retn
                                <1> load_FS_sub_directory:
0000BB6A C3                     <1>       retn
                                <1>
                                <1> read_cluster:
                                <1>       ; 15/10/2016
                                <1>       ; 18/03/2016
                                <1>       ; 16/03/2016
                                <1>       ; 17/02/2016
                                <1>       ; 15/02/2016 (TRDOS 386 =  TRDOS v2.0)
                                <1>       ;
                                <1>       ; INPUT ->
                                <1>       ;     EAX = Cluster Number (Sector index for SINGLIX FS)
                                <1>       ;     ESI = Logical DOS Drive Description Table address
                                <1>       ;     EBX = Cluster (File R/W) Buffer address (max. 64KB)
                                <1>       ;     Only for SINGLIX FS:
                                <1>       ;     EDX = File Number (The 1st FDT address)
                                <1>       ; OUTPUT ->
                                <1>       ;     cf = 1 -> Cluster can not be loaded at the buffer
                                <1>       ;         EAX > 0 -> Error number
                                <1>       ;     cf = 0 -> Cluster has been loaded at the buffer
                                <1>       ;
                                <1>       ; (Modified registers: EAX, ECX, EBX, EDX)
                                <1>
0000BB6B 0FB64E13               <1>       movzx  ecx, byte [esi+LD_BPB+BPB_SecPerClust]
                                <1>       ; CL = 1 = [esi+LD_FS_Reserved2] ; SectPerClust for Singlix FS
                                <1>
                                <1> read_file_sectors: ; 16/03/2016
0000BB6F 807E0300               <1>       cmp   byte [esi+LD_FATType], 0
0000BB73 761C                   <1>       jna   short read_fs_cluster
                                <1>
                                <1> read_fat_file_sectors: ; 18/03/2016
0000BB75 83E802                 <1>       sub   eax, 2 ; Beginning cluster number is always 2
0000BB78 0FB65613               <1>       movzx  edx, byte [esi+LD_BPB+BPB_SecPerClust] ; 18/03/2016
0000BB7C F7E2                   <1>       mul   edx
0000BB7E 034668                 <1>       add   eax, [esi+LD_DATABegin] ; absolute address of the cluster
                                <1>
                                <1>       ; EAX = Disk sector address
                                <1>       ; ECX = Sector count
                                <1>       ; EBX = Buffer address
                                <1>       ; (EDX = 0)
                                <1>       ; ESI = Logical DOS drive description table address
                                <1>
0000BB81 E852360000             <1>       call  disk_read
0000BB86 7306                   <1>       jnc   short rclust_retn
                                <1>
                                <1>       ; 15/10/2016 (15h -> 17)
0000BB88 B811000000             <1>       mov   eax, 17 ; Drive not ready or read error !
0000BB8D C3                     <1>       retn
                                <1>
                                <1> rclust_retn:
0000BB8E 29C0                   <1>       sub   eax, eax ; 0
0000BB90 C3                     <1>       retn
                                <1>
                                <1> read_fs_cluster:
```

```
34089                                  <1>       ; 15/02/2016 (TRDOS 386 =  TRDOS v2.0)
34090                                  <1>       ; Singlix FS
34091                                  <1>
34092                                  <1>       ; EAX = Cluster number is sector index number of the file (eax)
34093                                  <1>
34094                                  <1>       ; EDX = File number is the first File Descriptor Table address
34095                                  <1>       ;     of the file. (Absolute address of the FDT).
34096                                  <1>
34097                                  <1>       ; eax = sector index (0 for the first sector)
34098                                  <1>       ; edx = FDT0 address
34099                                  <1>             ; 64 KB buffer = 128 sectors (limit)
34100 0000BB91 B980000000              <1>       mov   ecx, 128 ; maximum count of sectors (before eof)
34101 0000BB96 E801000000              <1>       call  read_fs_sectors
34102 0000BB9B C3                      <1>       retn
34103                                  <1>
34104                                  <1> read_fs_sectors:
34105                                  <1>       ; 15/02/2016 (TRDOS 386 =  TRDOS v2.0)
34106 0000BB9C F9                      <1>       stc
34107 0000BB9D C3                      <1>       retn
34108                                  <1>
34109                                  <1> get_first_free_cluster:
34110                                  <1>       ; 02/03/2016
34111                                  <1>       ; 21/02/2016 (TRDOS 386 =  TRDOS v2.0)
34112                                  <1>       ; 26/10/2010 (DRV_FAT.ASM, 'proc_get_first_free_cluster')
34113                                  <1>       ; 10/07/2010
34114                                  <1>       ; INPUT ->
34115                                  <1>       ;     ESI = Logical DOS Drive Description Table address
34116                                  <1>       ; OUTPUT ->
34117                                  <1>       ;     cf = 1 -> Error code in AL (EAX)
34118                                  <1>       ;     cf = 0 ->
34119                                  <1>       ;       EAX = Cluster number
34120                                  <1>       ;       If EAX = FFFFFFFFh -> no free space
34121                                  <1>       ;     If the drive has FAT32 fs:
34122                                  <1>       ;       EBX = FAT32 FSI sector buffer address (if > 0)
34123                                  <1>
34124 0000BB9E 8B4678                  <1>       mov   eax, [esi+LD_Clusters]
34125 0000BBA1 40                      <1>       inc   eax ; add eax, 1
34126 0000BBA2 A3[985D0100]            <1>       mov   [gffc_last_free_cluster], eax
34127                                  <1>
34128 0000BBA7 31DB                    <1>       xor   ebx, ebx ; 0 ; 02/03/2016
34129                                  <1>
34130 0000BBA9 807E0302                <1>       cmp   byte [esi+LD_FATType], 2
34131 0000BBAD 760E                    <1>       jna   short loc_gffc_get_first_fat_free_cluster0
34132                                  <1>
34133                                  <1> loc_gffc_get_first_fat32_free_cluster:
34134                                  <1>       ; 02/03/2016
34135 0000BBAF E844060000              <1>       call  get_fat32_fsinfo_sector_parms
34136 0000BBB4 7207                    <1>       jc    short loc_gffc_get_first_fat_free_cluster0
34137                                  <1>
34138                                  <1> loc_gffc_check_fsinfo_parms:
34139                                  <1>       ;;mov  ebx, DOSBootSectorBuff
34140                                  <1>       ;cmp   dword [ebx], 41615252h
34141                                  <1>       ;jne   short loc_gffc_fat32_fsinfo_err
34142                                  <1>       ;cmp   dword [ebx+484], 61417272h
34143                                  <1>       ;jne   short loc_gffc_fat32_fsinfo_err
34144                                  <1>       ;mov   eax, [ebx+492] ; FSI_Next_Free
34145                                  <1>       ;EAX = First free cluster
34146                                  <1>       ;(from FAT32 FSInfo sector)
34147 0000BBB6 89D0                    <1>       mov   eax, edx ; FSI_Next_Free (First Free Cluster)
34148 0000BBB8 83F8FF                  <1>       cmp   eax, 0FFFFFFFFh ; invalid (unknown) !
34149 0000BBBB 7205                    <1>       jb    short loc_gffc_get_first_fat_free_cluster1
34150                                  <1>
34151                                  <1>       ; Start from the 1st cluster of the FAT(32) file system
34152                                  <1> loc_gffc_get_first_fat_free_cluster0:
34153 0000BBBD B802000000              <1>       mov   eax, 2
34154                                  <1>       ;xor   edx, edx
34155                                  <1>
34156                                  <1> loc_gffc_get_first_fat_free_cluster1:
34157 0000BBC2 53                      <1>       push  ebx ; 02/03/2016
34158                                  <1>
34159                                  <1> loc_gffc_get_first_fat_free_cluster2:
34160 0000BBC3 A3[945D0100]            <1>       mov   [gffc_first_free_cluster], eax
34161 0000BBC8 A3[905D0100]            <1>       mov   [gffc_next_free_cluster], eax
34162                                  <1>
34163                                  <1>       ; EBX = FAT32 FSINFO sector buffer address
34164                                  <1>       ; (EBX = 0, if the drive has not got FAT32 fs or
34165                                  <1>       ; FAT32 FSINFO sector buffer is invalid.)
34166                                  <1>
34167                                  <1> loc_gffc_get_first_fat_free_cluster3:
34168 0000BBCD E875FDFFFF              <1>       call  get_next_cluster
34169 0000BBD2 7307                    <1>       jnc   short loc_gffc_get_first_fat_free_cluster4
34170 0000BBD4 09C0                    <1>       or    eax, eax
34171 0000BBD6 740B                    <1>       jz    short loc_gffc_first_free_fat_cluster_next
34172 0000BBD8 5B                      <1>       pop   ebx ; 02/03/2016
34173 0000BBD9 F5                      <1>       cmc   ; stc
34174 0000BBDA C3                      <1>       retn
34175                                  <1>
34176                                  <1> loc_gffc_get_first_fat_free_cluster4:
34177 0000BBDB 21C0                    <1>       and   eax, eax ; next cluster value
34178 0000BBDD 7504                    <1>       jnz   short loc_gffc_first_free_fat_cluster_next
34179 0000BBDF 89C8                    <1>       mov   eax, ecx ; current (previous cluster) value
34180 0000BBE1 EB22                    <1>       jmp   short loc_gffc_check_for_set
34181                                  <1>
34182                                  <1> loc_gffc_first_free_fat_cluster_next:
34183 0000BBE3 A1[905D0100]            <1>       mov   eax, [gffc_next_free_cluster]
34184 0000BBE8 3B05[985D0100]          <1>       cmp   eax, [gffc_last_free_cluster]
34185 0000BBEE 7308                    <1>       jnb   short retn_stc_from_get_first_free_cluster
34186                                  <1> pass_gffc_last_cluster_eax_check:
34187 0000BBF0 40                      <1>       inc   eax ; add eax, 1
34188 0000BBF1 A3[905D0100]            <1>       mov   [gffc_next_free_cluster], eax
34189 0000BBF6 EBD5                    <1>       jmp   short loc_gffc_get_first_fat_free_cluster3
34190                                  <1>
34191                                  <1> retn_stc_from_get_first_free_cluster:
```

```
34192 0000BBF8 A1[945D0100]        <1>       mov    eax, [gffc_first_free_cluster]
34193 0000BBFD 83F802              <1>       cmp    eax, 2
34194 0000BC00 7709                <1>       ja     short loc_gffc_check_previous_clusters
34195 0000BC02 29C0                <1>       sub    eax, eax
34196 0000BC04 48                  <1>       dec    eax ; FFFFFFFFh
34197                              <1>
34198                              <1> loc_gffc_check_for_set:
34199                              <1>       ; 02/03/2016
34200 0000BC05 5B                  <1>       pop    ebx
34201                              <1>
34202                              <1>       ; EBX = FAT32 FSINFO sector buffer address
34203                              <1>       ; (EBX = 0, if the drive has not got FAT32 fs or
34204                              <1>       ; FAT32 FSINFO sector buffer is invalid.)
34205                              <1>
34206 0000BC06 09DB                <1>       or     ebx, ebx
34207 0000BC08 750E                <1>       jnz    short loc_gffc_set_ffree_fat32_cluster
34208                              <1>
34209                              <1>       ;cmp   byte [esi+LD_FATType], 3
34210                              <1>       ;jnb   short loc_gffc_set_ffree_fat32_cluster
34211                              <1>
34212                              <1>       ;xor   ebx, ebx ; 0
34213                              <1>
34214                              <1> loc_gffc_retn:
34215 0000BC0A C3                  <1>       retn
34216                              <1>
34217                              <1> loc_gffc_check_previous_clusters:
34218 0000BC0B 48                  <1>       dec    eax ; sub eax, 1
34219 0000BC0C A3[985D0100]        <1>       mov    [gffc_last_free_cluster], eax
34220 0000BC11 B802000000          <1>       mov    eax, 2
34221                              <1>       ;xor   edx, edx
34222 0000BC16 EBAB                <1>       jmp    short loc_gffc_get_first_fat_free_cluster2
34223                              <1>
34224                              <1> loc_gffc_set_ffree_fat32_cluster:
34225                              <1>       ;call  set_first_free_cluster
34226                              <1>       ;retn
34227                              <1>       ;jmp   short set_first_free_cluster
34228                              <1>
34229                              <1> set_first_free_cluster:
34230                              <1>       ; 15/10/2016
34231                              <1>       ; 23/03/2016
34232                              <1>       ; 02/03/2016
34233                              <1>       ; 29/02/2016
34234                              <1>       ; 26/02/2016
34235                              <1>       ; 21/02/2016 (TRDOS 386 =  TRDOS v2.0)
34236                              <1>       ; 21/08/2011 (DRV_FAT.ASM, 'proc_set_first_free_cluster')
34237                              <1>       ; 11/07/2010
34238                              <1>       ; INPUT ->
34239                              <1>       ;      ESI = Logical DOS Drive Description Table address
34240                              <1>       ;      EAX = First free cluster
34241                              <1>       ;      EBX = FSINFO sector buffer address
34242                              <1>       ;      ;;If EBX > 0, it is FSINFO sector buffer address
34243                              <1>       ;      ;;EBX = 0, if FSINFO sector is not loaded
34244                              <1>       ; OUTPUT->
34245                              <1>       ;      ESI = Logical DOS Drive Description Table address
34246                              <1>       ;      If EBX > 0, it is FSINFO sector buffer address
34247                              <1>       ;      EBX = 0, if FSINFO sector could not be loaded
34248                              <1>       ;      CF = 1 -> Error code in AL (EAX)
34249                              <1>       ;      CF = 0 -> first free cluster is successfully updated
34250                              <1>
34251                              <1>       ;cmp   byte [esi+LD_FATType], 3
34252                              <1>       ;jb    short loc_sffc_invalid_drive
34253                              <1>
34254                              <1>       ; Save First Free Cluster value for 'update_cluster'
34255 0000BC18 89463E              <1>       mov    [esi+LD_BPB+BPB_Reserved+4], eax ; First free Cluster
34256                              <1>
34257                              <1>       ;or    ebx, ebx
34258                              <1>       ;jnz   short loc_sffc_read_fsinfo_sector
34259                              <1>
34260 0000BC1B 813B52526141        <1>       cmp     dword [ebx], 41615252h
34261 0000BC21 7540                <1>       jne    short loc_sffc_read_fsinfo_sector
34262 0000BC23 81BBE4010000727241- <1>       cmp    dword [ebx+484], 61417272h
34263 0000BC2C 61                  <1>
34264 0000BC2D 7534                <1>       jne    short loc_sffc_read_fsinfo_sector
34265                              <1>
34266 0000BC2F 3B83EC010000        <1>       cmp    eax, [ebx+492]  ; FSI_Next_Free
34267 0000BC35 741F                <1>       je     short loc_sffc_retn
34268                              <1>
34269                              <1> loc_sffc_write_fsinfo_sector:
34270                              <1>       ; EBX = FSINFO sector buffer
34271                              <1>       ; [CFS_FAT32FSINFOSEC] is set in 'get_fat32_fsinfo_sector_parms'
34272 0000BC37 8983EC010000        <1>       mov    [ebx+492], eax
34273 0000BC3D A1[A85D0100]        <1>       mov    eax, [CFS_FAT32FSINFOSEC]
34274 0000BC42 B901000000          <1>       mov    ecx, 1
34275 0000BC47 53                  <1>       push   ebx
34276 0000BC48 E87C350000          <1>       call   disk_write
34277 0000BC4D 7208                <1>       jc      short loc_sffc_read_fsinfo_sector_err1
34278 0000BC4F 5B                  <1>       pop    ebx
34279                              <1>
34280 0000BC50 8B83EC010000        <1>       mov    eax, [ebx+492] ; First (Next) Free Cluster
34281                              <1>
34282                              <1> loc_sffc_retn:
34283 0000BC56 C3                  <1>       retn
34284                              <1>
34285                              <1> ;loc_sffc_invalid_drive:
34286                              <1> ;     mov    eax, 0Fh ; MSDOS Error : Invalid drive
34287                              <1> ;     push   edx
34288                              <1>
34289                              <1> loc_sffc_read_fsinfo_sector_err1:
34290 0000BC57 BB00000000          <1>       mov    ebx, 0
34291                              <1>       ; 15/10/2016 (1Dh -> 18)
34292                              <1>       ; 23/03/2016 (1Dh)
34293 0000BC5C B812000000          <1>       mov    eax, 18 ; Drive not ready or write error
34294                              <1>
```

```
34295                                  <1> loc_sffc_read_fsinfo_sector_err2:
34296 0000BC61 5A                      <1>       pop    edx
34297 0000BC62 C3                      <1>       retn
34298                                  <1>
34299                                  <1> loc_sffc_read_fsinfo_sector:
34300 0000BC63 50                      <1>       push   eax
34301                                  <1>
34302 0000BC64 E88F050000              <1>       call   get_fat32_fsinfo_sector_parms
34303 0000BC69 72F6                    <1>       jc     short loc_sffc_read_fsinfo_sector_err2
34304                                  <1>
34305 0000BC6B 58                      <1>       pop    eax
34306                                  <1>       ; EDX = First (Next) Free Cluster value from FSINFO sector
34307                                  <1>       ; EAX = First Free Cluster value from 'get_next_cluster'
34308                                  <1>       ; (edx = old value)
34309 0000BC6C 39D0                    <1>       cmp    eax, edx ; First free Cluster (eax = new value)
34310 0000BC6E 75C7                    <1>       jne    short loc_sffc_write_fsinfo_sector
34311                                  <1>
34312 0000BC70 C3                      <1>       retn
34313                                  <1>
34314                                  <1> update_cluster:
34315                                  <1>       ; 23/10/2016
34316                                  <1>       ; 23/03/2016
34317                                  <1>       ; 02/03/2016
34318                                  <1>       ; 01/03/2016
34319                                  <1>       ; 29/02/2016
34320                                  <1>       ; 27/02/2016
34321                                  <1>       ; 26/02/2016
34322                                  <1>       ; 22/02/2016 (TRDOS 386 =  TRDOS v2.0)
34323                                  <1>       ; 11/08/2011
34324                                  <1>       ; 09/02/2005
34325                                  <1>       ; INPUT ->
34326                                  <1>       ;      EAX = Cluster Number
34327                                  <1>       ;      ECX = New Cluster Value
34328                                  <1>       ;      ESI = Logical Dos Drive Parameters Table
34329                                  <1>       ;
34330                                  <1>       ;      /// dword [FAT_ClusterCounter] ///
34331                                  <1>       ;
34332                                  <1>       ; OUTPUT ->
34333                                  <1>       ;      cf = 0 -> No Error, EAX is valid
34334                                  <1>       ;      cf = 1 & EAX = 0 -> End Of Cluster Chain
34335                                  <1>       ;      cf = 1 & EAX > 0 -> Error
34336                                  <1>       ;            (ECX -> any value)
34337                                  <1>       ;      EAX = Next Cluster
34338                                  <1>       ;      ECX = New Cluster Value
34339                                  <1>       ;
34340                                  <1>       ;      /// [FAT_ClusterCounter] is updated,
34341                                  <1>       ;      /// decreased when a free cluster is assigned,
34342                                  <1>       ;      /// increased if an assigned cluster is freed.
34343                                  <1>       ;
34344                                  <1>       ;
34345                                  <1>       ; (Modified registers: EAX, EBX, -ECX-, EDX)
34346                                  <1>
34347 0000BC71 A3[FA5A0100]            <1>       mov    [FAT_CurrentCluster], eax
34348 0000BC76 890D[9C5D0100]          <1>       mov    [ClusterValue], ecx
34349                                  <1>
34350                                  <1> loc_update_cluster_check_fat_buffer:
34351 0000BC7C 8A1E                    <1>       mov    bl, [esi+LD_Name]
34352 0000BC7E 381D[FF5A0100]          <1>       cmp    [FAT_BuffDrvName], bl
34353 0000BC84 741A                    <1>       je     short loc_update_cluster_check_fat_type
34354 0000BC86 803D[FE5A0100]02        <1>       cmp    byte [FAT_BuffValidData], 2
34355 0000BC8D 0F84C2000000            <1>         je     loc_uc_save_fat_buffer
34356                                  <1>
34357                                  <1> loc_uc_reset_fat_buffer_validation:
34358 0000BC93 C605[FE5A0100]00        <1>       mov    byte [FAT_BuffValidData], 0
34359                                  <1>
34360                                  <1> loc_uc_check_fat_type_reset_drvname:
34361 0000BC9A 881D[FF5A0100]          <1>       mov    [FAT_BuffDrvName], bl
34362                                  <1>
34363                                  <1> loc_update_cluster_check_fat_type:
34364 0000BCA0 29D2                    <1>       sub    edx, edx ; 26/02/2016
34365 0000BCA2 8A5E03                  <1>       mov    bl, [esi+LD_FATType]
34366 0000BCA5 83F802                  <1>       cmp    eax, 2
34367 0000BCA8 0F82BE000000            <1>         jb     update_cluster_inv_data
34368 0000BCAE 80FB02                  <1>       cmp    bl, 2
34369 0000BCB1 0F877A010000            <1>         ja     update_fat32_cluster
34370                                  <1>       ;cmp   bl, 1
34371                                  <1>       ;jb    short update_cluster_inv_data
34372 0000BCB7 8B4E78                  <1>       mov    ecx, [esi+LD_Clusters]
34373 0000BCBA 41                      <1>       inc    ecx
34374 0000BCBB 890D[0A5B0100]          <1>       mov    [LastCluster], ecx
34375 0000BCC1 39C8                    <1>       cmp    eax, ecx ; dword [LastCluster]
34376 0000BCC3 0F87A6000000            <1>         ja     return_uc_fat_stc
34377                                  <1>       ; TRDOS v1 has a FATal bug here !
34378                                  <1>              ; or bl, bl ; cmp bl, 0
34379                                  <1>              ; jz short update_fat12_cluster
34380                                  <1>       ; !! It would destroy FAT12 floppy disk fs here !!
34381                                  <1>       ; ('A:' disks of TRDOS v1 operating system project
34382                                  <1>       ; had 'singlix fs', so, I could not differ this mistake
34383                                  <1>       ; on a drive 'A:')
34384 0000BCC9 80FB01                  <1>       cmp    bl, 1 ; correct comparison is this !
34385 0000BCCC 0F86A2000000            <1>         jna    update_fat12_cluster
34386                                  <1>
34387                                  <1> update_fat16_cluster:
34388                                  <1> pass_uc_fat16_errc:
34389                                  <1>       ;sub   edx, edx
34390 0000BCD2 BB00030000              <1>       mov    ebx, 300h ;768
34391 0000BCD7 F7F3                    <1>       div    ebx
34392                                  <1>       ; EAX = Count of 3 FAT sectors
34393                                  <1>       ; DX = Cluster offset in FAT buffer
34394 0000BCD9 6689D3                  <1>       mov    bx, dx
34395 0000BCDC 66D1E3                  <1>       shl    bx, 1 ; Multiply by 2
34396 0000BCDF 66BA0300                <1>       mov    dx, 3
34397 0000BCE3 F7E2                    <1>       mul    edx
```

```
                                <1>       ; EAX = FAT Sector
                                <1>       ; EDX = 0
                                <1>       ; EBX = Byte offset in FAT buffer
0000BCE5 8A0D[FE5A0100]         <1>       mov   cl, [FAT_BuffValidData]
0000BCEB 80F902                 <1>       cmp   cl, 2
0000BCEE 750A                   <1>       jne   short loc_uc_check_fat16_buff_sector_load
                                <1>
                                <1> loc_uc_check_fat16_buff_sector_save:
0000BCF0 3B05[025B0100]         <1>       cmp   eax, [FAT_BuffSector]
0000BCF6 755D                   <1>       jne   short loc_uc_save_fat_buffer
0000BCF8 EB15                   <1>       jmp   short loc_update_fat16_cell
                                <1>
                                <1> loc_uc_check_fat16_buff_sector_load:
0000BCFA 80F901                 <1>       cmp   cl, 1 ; byte [FAT_BuffValidData]
0000BCFD 0F85FB010000           <1>         jne     loc_uc_load_fat_sectors
0000BD03 3B05[025B0100]         <1>       cmp   eax, [FAT_BuffSector]
0000BD09 0F85EF010000           <1>         jne     loc_uc_load_fat_sectors
                                <1>
                                <1> loc_update_fat16_cell:
                                <1> loc_update_fat16_buffer:
0000BD0F 81C3001C0900           <1>       add   ebx, FAT_Buffer ; 26/02/2016
                                <1>       ;movzx eax, word [ebx]
0000BD15 668B03                 <1>       mov   ax, [ebx]
                                <1>       ; 01/03/2016
0000BD18 89C2                   <1>       mov   edx, eax ; old value of the cluster
0000BD1A A3[FA5A0100]           <1>       mov   [FAT_CurrentCluster], eax
0000BD1F 8B0D[9C5D0100]         <1>       mov   ecx, [ClusterValue] ; 32 bits
0000BD25 66890B                 <1>       mov   [ebx], cx ; 16 bits !
                                <1>
0000BD28 C605[FE5A0100]02       <1>       mov   byte [FAT_BuffValidData], 2
                                <1>
0000BD2F 6683F802               <1>       cmp   ax, 2
0000BD33 723A                   <1>       jb    short return_uc_fat_stc
0000BD35 3B05[0A5B0100]         <1>       cmp   eax, [LastCluster]
0000BD3B 7732                   <1>       ja    short return_uc_fat_stc
                                <1>
                                <1> loc_fat_buffer_updated:
                                <1>       ; 01/03/2016
0000BD3D F8                     <1>       clc
                                <1> loc_fat_buffer_stc_1:
0000BD3E 9C                     <1>       pushf
0000BD3F 21C9                   <1>       and   ecx, ecx
0000BD41 7506                   <1>       jnz   short loc_fat_buffer_updated_1
                                <1>
                                <1>       ; 01/03/2016
                                <1>       ; new value of the cluster = 0 (free)
                                <1>       ; increase free(d) cluster count
0000BD43 FF05[065B0100]         <1>       inc   dword [FAT_ClusterCounter]
                                <1>
                                <1> loc_fat_buffer_updated_1: ; new value of the cluster > 0
0000BD49 09D2                   <1>       or    edx, edx ; 02/03/2016
0000BD4B 7506                   <1>       jnz   short loc_fat_buffer_updated_2
                                <1>       ; old value of the cluster = 0 (it was free cluster)
                                <1>       ; decrease free(d) cluster count
0000BD4D FF0D[065B0100]         <1>       dec   dword [FAT_ClusterCounter] ; it may be negative number
                                <1>
                                <1> loc_fat_buffer_updated_2:
0000BD53 9D                     <1>       popf
0000BD54 C3                     <1>       retn
                                <1>
                                <1> loc_uc_save_fat_buffer:
                                <1>       ; byte [FAT_BuffValidData] = 2
0000BD55 E8D4010000             <1>       call  save_fat_buffer
0000BD5A 0F8297010000           <1>         jc      loc_fat_sectors_rw_error2
                                <1>       ;mov  byte [FAT_BuffValidData], 1
0000BD60 A1[FA5A0100]           <1>       mov   eax, [FAT_CurrentCluster]
                                <1>       ;mov  ecx, [ClusterValue]
                                <1>       ;jmp  short loc_update_cluster_check_fat_buffer
0000BD65 8A1E                   <1>       mov   bl, [esi+LD_Name] ; 01/03/2016
0000BD67 E927FFFFFF             <1>         jmp     loc_uc_reset_fat_buffer_validation
                                <1>
                                <1> update_cluster_inv_data:
                                <1>       ;mov  eax, 0Dh
0000BD6C B00D                   <1>       mov   al, 0Dh  ; Invalid Data
0000BD6E C3                     <1>       retn
                                <1>
                                <1> return_uc_fat_stc:
                                <1>       ; 01/03/2016
0000BD6F 31C0                   <1>       xor   eax, eax
0000BD71 F9                     <1>       stc
0000BD72 EBCA                   <1>       jmp   short loc_fat_buffer_stc_1
                                <1>
                                <1> update_fat12_cluster:
                                <1> pass_uc_fat12_errc:
                                <1>       ;sub  edx, edx
0000BD74 BB00040000             <1>       mov   ebx, 400h ;1024
0000BD79 F7F3                   <1>       div   ebx
                                <1>       ; EAX = Count of 3 FAT sectors
                                <1>       ; DX = Cluster offset in FAT buffer
0000BD7B 66B90300               <1>       mov   cx, 3
0000BD7F 6689C3                 <1>       mov   bx, ax
0000BD82 6689C8                 <1>       mov   ax, cx ; 3
0000BD85 66F7E2                 <1>       mul   dx     ; Multiply by 3
0000BD88 66D1E8                 <1>       shr   ax, 1  ; Divide by 2
0000BD8B 6693                   <1>       xchg  bx, ax
                                <1>       ; EAX = Count of 3 FAT sectors
                                <1>       ; EBX = Byte Offset in FAT buffer
0000BD8D 66F7E1                 <1>       mul   cx  ; 3 * AX
                                <1>       ; EAX = FAT Beginning Sector
                                <1>       ; EDX = 0
0000BD90 8A0D[FE5A0100]         <1>       mov   cl, [FAT_BuffValidData]
                                <1>       ; TRDOS v1 has a FATal bug here !
                                <1>       ; (it does not have 'cmp cl, 2' instruction here !
```

```
34501                                  <1>       ;  while 'jne' is existing !)
34502 0000BD96 80F902                  <1>       cmp    cl, 2 ; 2 = dirty buffer (must be written to disk)
34503 0000BD99 750A                    <1>       jne    short loc_uc_check_fat12_buff_sector_load
34504                                  <1>
34505                                  <1> loc_uc_check_fat12_buff_sector_save:
34506 0000BD9B 3B05[025B0100]          <1>       cmp    eax, [FAT_BuffSector]
34507 0000BDA1 75B2                    <1>         jne     short loc_uc_save_fat_buffer
34508 0000BDA3 EB15                    <1>       jmp    short loc_update_fat12_cell
34509                                  <1>
34510                                  <1> loc_uc_check_fat12_buff_sector_load:
34511 0000BDA5 80F901                  <1>       cmp    cl, 1 ; byte ptr [FAT_BuffValidData]
34512 0000BDA8 0F8550010000            <1>         jne     loc_uc_load_fat_sectors
34513 0000BDAE 3B05[025B0100]          <1>       cmp    eax, [FAT_BuffSector]
34514 0000BDB4 0F8544010000            <1>         jne     loc_uc_load_fat_sectors
34515                                  <1>
34516                                  <1> loc_update_fat12_cell:
34517 0000BDBA 81C3001C0900            <1>       add    ebx, FAT_Buffer ; 26/02/2016
34518 0000BDC0 668B0D[FA5A0100]        <1>       mov    cx, [FAT_CurrentCluster]
34519 0000BDC7 66D1E9                  <1>       shr    cx, 1
34520 0000BDCA 668B03                  <1>       mov    ax, [ebx]
34521 0000BDCD 6689C2                  <1>       mov    dx, ax
34522 0000BDD0 7344                    <1>       jnc    short uc_fat12_nc_even
34523                                  <1>
34524 0000BDD2 6683E00F                <1>       and    ax, 0Fh
34525 0000BDD6 8B0D[9C5D0100]          <1>       mov    ecx, [ClusterValue] ; 32 bits
34526 0000BDDC 66C1E104                <1>       shl    cx, 4
34527 0000BDE0 6609C1                  <1>       or     cx, ax
34528 0000BDE3 6689D0                  <1>       mov    ax, dx
34529 0000BDE6 66890B                  <1>       mov    [ebx], cx  ; 16 bits !
34530 0000BDE9 66C1E804                <1>       shr    ax, 4 ; al(bit4..7)+ah(bit0..7)
34531                                  <1>
34532                                  <1> update_fat12_buffer:
34533 0000BDED A3[FA5A0100]            <1>       mov    [FAT_CurrentCluster], eax
34534 0000BDF2 89C2                    <1>       mov    edx, eax ; 01/03/2016
34535 0000BDF4 C605[FE5A0100]02        <1>       mov    byte [FAT_BuffValidData], 2
34536 0000BDFB 6683F802                <1>       cmp    ax, 2
34537 0000BDFF 0F826AFFFFFF            <1>         jb      return_uc_fat_stc
34538 0000BE05 3B05[0A5B0100]          <1>       cmp    eax, [LastCluster]
34539 0000BE0B 0F875EFFFFFF            <1>         ja      return_uc_fat_stc
34540 0000BE11 E927FFFFFF              <1>         jmp     loc_fat_buffer_updated
34541                                  <1>
34542                                  <1> uc_fat12_nc_even:
34543 0000BE16 662500F0                <1>       and    ax, 0F000h
34544 0000BE1A 8B0D[9C5D0100]          <1>       mov    ecx, [ClusterValue] ; 32 bits
34545 0000BE20 80E50F                  <1>       and    ch, 0Fh
34546 0000BE23 6609C1                  <1>       or     cx, ax
34547 0000BE26 6689D0                  <1>       mov    ax, dx
34548 0000BE29 66890B                  <1>       mov    [ebx], cx ; 16 bits !
34549 0000BE2C 80E40F                  <1>       and    ah, 0Fh ; al(bit0..7)+ah(bit0..3)
34550 0000BE2F EBBC                    <1>       jmp    short update_fat12_buffer
34551                                  <1>
34552                                  <1> update_fat32_cluster:
34553 0000BE31 8B4E78                  <1>       mov    ecx, [esi+LD_Clusters]
34554 0000BE34 41                      <1>       inc    ecx
34555 0000BE35 890D[0A5B0100]          <1>       mov    [LastCluster], ecx
34556                                  <1>
34557 0000BE3B 39C8                    <1>       cmp    eax, ecx
34558 0000BE3D 0F872CFFFFFF            <1>         ja      return_uc_fat_stc
34559                                  <1>
34560                                  <1> pass_uc_fat32_errc:
34561                                  <1>       ;sub   edx, edx
34562 0000BE43 BB80010000              <1>       mov    ebx, 180h ;384
34563 0000BE48 F7F3                    <1>       div    ebx
34564                                  <1>       ; EAX = Count of 3 FAT sectors
34565                                  <1>       ; DX = Cluster offset in FAT buffer
34566 0000BE4A 89D3                    <1>       mov    ebx, edx
34567 0000BE4C C1E302                  <1>       shl    ebx, 2 ; Multiply by 4
34568 0000BE4F BA03000000              <1>       mov    edx, 3
34569 0000BE54 F7E2                    <1>       mul    edx
34570                                  <1>       ; EBX = Cluster Offset in FAT buffer
34571                                  <1>       ; EAX = FAT Sector
34572                                  <1>       ; EDX = 0
34573 0000BE56 8A0D[FE5A0100]          <1>       mov    cl, [FAT_BuffValidData]
34574 0000BE5C 80F902                  <1>       cmp    cl, 2
34575 0000BE5F 750E                    <1>       jne    short loc_uc_check_fat32_buff_sector_load
34576                                  <1>
34577                                  <1> loc_uc_check_fat32_buff_sector_save:
34578 0000BE61 3B05[025B0100]          <1>       cmp    eax, [FAT_BuffSector]
34579 0000BE67 0F85E8FEFFFF            <1>         jne     loc_uc_save_fat_buffer
34580 0000BE6D EB11                    <1>       jmp    short loc_update_fat32_cell
34581                                  <1>
34582                                  <1> loc_uc_check_fat32_buff_sector_load:
34583 0000BE6F 80F901                  <1>       cmp    cl, 1 ; byte [FAT_BuffValidData]
34584 0000BE72 0F8586000000            <1>         jne     loc_uc_load_fat_sectors
34585 0000BE78 3B05[025B0100]          <1>       cmp    eax, [FAT_BuffSector]
34586 0000BE7E 757E                    <1>         jne     loc_uc_load_fat_sectors
34587                                  <1>
34588                                  <1> loc_update_fat32_cell:
34589                                  <1> loc_update_fat32_buffer:
34590 0000BE80 81C3001C0900            <1>       add    ebx, FAT_Buffer ; 26/02/2016
34591 0000BE86 8B03                    <1>       mov    eax, [ebx]
34592 0000BE88 25FFFFFF0F              <1>       and    eax, 0FFFFFFFh ; 28 bit cluster value
34593                                  <1>
34594 0000BE8D 8B15[FA5A0100]          <1>       mov    edx, [FAT_CurrentCluster] ; 01/03/2016
34595                                  <1>
34596 0000BE93 A3[FA5A0100]            <1>       mov    [FAT_CurrentCluster], eax
34597 0000BE98 8B0D[9C5D0100]          <1>       mov    ecx, [ClusterValue]
34598 0000BE9E 890B                    <1>       mov    [ebx], ecx ; 29/02/2016
34599                                  <1>
34600 0000BEA0 C605[FE5A0100]02        <1>       mov    byte [FAT_BuffValidData], 2
34601                                  <1>
34602                                  <1>       ; 01/03/2016
34603 0000BEA7 21C0                    <1>       and    eax, eax ; was it free cluster ?
```

```
34604 0000BEA9 7514                <1>       jnz    short loc_upd_fat32_c0
34605                              <1>
34606                              <1>       ;or    ecx, ecx ; it will be left free ?!
34607                              <1>       ;jz    short loc_upd_fat32_c3
34608                              <1>
34609 0000BEAB 3B563E              <1>       cmp    edx, [esi+LD_BPB+BPB_Reserved+4] ; First free cluster
34610 0000BEAE 7520                <1>       jne    short loc_upd_fat32_c3
34611                              <1>
34612 0000BEB0 3B15[0A5B0100]      <1>       cmp    edx, [LastCluster]
34613 0000BEB6 7207                <1>       jb     short loc_upd_fat32_c0
34614                              <1>
34615 0000BEB8 BA02000000          <1>       mov    edx, 2 ; rewind !
34616 0000BEBD EB0E                <1>       jmp    short loc_upd_fat32_c2
34617                              <1>
34618                              <1> loc_upd_fat32_c0:
34619 0000BEBF FF463E              <1>       inc    dword [esi+LD_BPB+BPB_Reserved+4] ; set it to next cluster
34620 0000BEC2 EB0C                <1>       jmp    short loc_upd_fat32_c3
34621                              <1>
34622                              <1> loc_upd_fat32_c1:
34623 0000BEC4 09C9                <1>       or     ecx, ecx ; will it be free cluster ?
34624 0000BEC6 7508                <1>       jnz    short loc_upd_fat32_c3
34625                              <1>
34626 0000BEC8 3B563E              <1>       cmp    edx, [esi+LD_BPB+BPB_Reserved+4] ; First free cluster
34627 0000BECB 7303                <1>       jnb    short loc_upd_fat32_c3
34628                              <1>
34629                              <1> loc_upd_fat32_c2:
34630 0000BECD 89563E              <1>       mov    [esi+LD_BPB+BPB_Reserved+4], edx
34631                              <1>
34632                              <1> loc_upd_fat32_c3:
34633 0000BED0 89C2                <1>       mov    edx, eax
34634                              <1>
34635                              <1> loc_upd_fat32_c4:
34636 0000BED2 83F802              <1>       cmp    eax, 2
34637 0000BED5 0F8294FEFFFF        <1>         jb      return_uc_fat_stc
34638                              <1>
34639                              <1> pass_uc_fat32_c_zero_check_2:
34640 0000BEDB 3B05[0A5B0100]      <1>       cmp    eax, [LastCluster]
34641 0000BEE1 0F8788FEFFFF        <1>         ja      return_uc_fat_stc
34642                              <1>
34643 0000BEE7 E951FEFFFF          <1>       jmp     loc_fat_buffer_updated
34644                              <1>
34645                              <1> loc_fat_sectors_rw_error1:
34646                              <1>       ;mov   byte [FAT_BuffValidData], 0
34647                              <1>       ; 23/10/2016 (15h -> 17)
34648                              <1>       ; 23/03/2016
34649 0000BEEC B811000000          <1>       mov    eax, 17 ; Drive not ready or read error
34650 0000BEF1 8825[FE5A0100]      <1>       mov    [FAT_BuffValidData], ah ; 0
34651                              <1>
34652                              <1> loc_fat_sectors_rw_error2:
34653                              <1>       ;mov   eax, error code
34654                              <1>       ;mov   edx, 0
34655 0000BEF7 8B0D[9C5D0100]      <1>       mov    ecx, [ClusterValue]
34656 0000BEFD C3                  <1>       retn
34657                              <1>
34658                              <1> loc_uc_load_fat_sectors:
34659 0000BEFE A3[025B0100]        <1>       mov    [FAT_BuffSector], eax
34660                              <1>
34661                              <1> load_uc_fat_sectors_zero:
34662 0000BF03 034660              <1>       add    eax, [esi+LD_FATBegin]
34663 0000BF06 BB001C0900          <1>       mov    ebx, FAT_Buffer
34664 0000BF0B B903000000          <1>       mov    ecx, 3
34665 0000BF10 E8C3320000          <1>       call   disk_read
34666 0000BF15 72D5                <1>       jc     short loc_fat_sectors_rw_error1
34667                              <1>
34668 0000BF17 C605[FE5A0100]01    <1>         mov     byte [FAT_BuffValidData], 1
34669 0000BF1E A1[FA5A0100]        <1>       mov    eax, [FAT_CurrentCluster]
34670 0000BF23 8B0D[9C5D0100]      <1>       mov    ecx, [ClusterValue]
34671 0000BF29 E972FDFFFF          <1>         jmp     loc_update_cluster_check_fat_type
34672                              <1>
34673                              <1> save_fat_buffer:
34674                              <1>       ; 15/10/2016
34675                              <1>       ; 01/03/2016
34676                              <1>       ; 22/02/2016 (TRDOS 386 =  TRDOS v2.0)
34677                              <1>       ; 11/08/2011
34678                              <1>       ; 09/02/2005
34679                              <1>       ; INPUT ->
34680                              <1>       ;      None
34681                              <1>       ; OUTPUT ->
34682                              <1>       ;      cf = 0 -> OK.
34683                              <1>       ;      cf = 1 -> error code in AL (EAX)
34684                              <1>       ;
34685                              <1>       ;      EBX = FAT_Buffer address
34686                              <1>       ;
34687                              <1>       ; (EAX, EDX, ECX will be modified)
34688                              <1>
34689                              <1>       ;cmp   byte [FAT_BuffValidData], 2
34690                              <1>       ;je    short loc_save_fat_buff
34691                              <1>
34692                              <1> ;loc_save_fat_buffer_retn:
34693                              <1> ;     xor    eax, eax
34694                              <1> ;     retn
34695                              <1>
34696                              <1> loc_save_fat_buff:
34697 0000BF2E 31D2                <1>       xor    edx, edx
34698 0000BF30 8A35[FF5A0100]      <1>       mov    dh, [FAT_BuffDrvName]
34699 0000BF36 80FE41              <1>       cmp    dh, 'A'
34700 0000BF39 722E                <1>       jb     short loc_save_fat_buffer_inv_data_retn
34701 0000BF3B 80EE41              <1>       sub    dh, 'A'
34702 0000BF3E 56                  <1>       push   esi ; *
34703 0000BF3F BE00010900          <1>         mov     esi, Logical_DOSDisks
34704 0000BF44 01D6                <1>       add    esi, edx
34705                              <1>
34706 0000BF46 8A5603              <1>       mov    dl, [esi+LD_FATType]
```

```
34707 0000BF49 20D2                <1>       and    dl, dl
34708 0000BF4B 741B                <1>       jz     short loc_save_fat_buffer_inv_data_pop_retn
34709                              <1>
34710 0000BF4D A1[025B0100]        <1>       mov    eax, [FAT_BuffSector]
34711 0000BF52 80FA02              <1>       cmp    dl, 2
34712 0000BF55 770A                <1>       ja     short loc_save_fat32_buff
34713                              <1>
34714                              <1> loc_save_fat_12_16_buff:
34715                              <1>       ; 01/03/2016
34716                              <1>       ; TRDOS v1 has a FATal bug here!
34717                              <1>       ; Correct code: mov dx, word ptr [FAT_BuffSector]+2
34718                              <1>       ; (DX:AX in TRDOS v1 -> EAX in TRDOS v2)
34719                              <1>       ;
34720 0000BF57 0FB74E1C            <1>       movzx  ecx, word [esi+LD_BPB+FATSecs]
34721 0000BF5B 29C1                <1>       sub    ecx, eax
34722                              <1>       ; TRDOS v1 has a bug here... ('pop esi' was forgotten!)
34723                              <1>       ;jna   short loc_save_fat_buffer_inv_data_retn ; wrong addr!
34724 0000BF5D 7609                <1>       jna    short loc_save_fat_buffer_inv_data_pop_retn ; correct addr.
34725 0000BF5F EB15                <1>       jmp    short loc_save_fat_buffer_check_rs3
34726                              <1>
34727                              <1> loc_save_fat32_buff:
34728 0000BF61 8B4E2A              <1>       mov    ecx, [esi+LD_BPB+FAT32_FAT_Size]
34729 0000BF64 29C1                <1>       sub    ecx, eax
34730 0000BF66 770E                <1>       ja     short loc_save_fat_buffer_check_rs3
34731                              <1>
34732                              <1> loc_save_fat_buffer_inv_data_pop_retn:
34733 0000BF68 5E                  <1>       pop    esi ; *
34734                              <1> loc_save_fat_buffer_inv_data_retn:
34735 0000BF69 B80D000000          <1>       mov    eax, 0Dh ; Invalid DATA
34736 0000BF6E C3                  <1>       retn
34737                              <1>
34738                              <1> loc_save_fat_buff_remain_sectors_3:
34739 0000BF6F B903000000          <1>       mov    ecx, 3
34740 0000BF74 EB05                <1>       jmp    short loc_save_fat_buff_continue
34741                              <1>
34742                              <1> loc_save_fat_buffer_check_rs3:
34743 0000BF76 83F903              <1>       cmp    ecx, 3
34744 0000BF79 77F4                <1>       ja     short loc_save_fat_buff_remain_sectors_3
34745                              <1>
34746                              <1> loc_save_fat_buff_continue:
34747 0000BF7B BB001C0900          <1>       mov    ebx, FAT_Buffer
34748 0000BF80 034660              <1>       add    eax, [esi+LD_FATBegin]
34749 0000BF83 51                  <1>       push   ecx
34750 0000BF84 E840320000          <1>       call   disk_write
34751 0000BF89 59                  <1>       pop    ecx
34752 0000BF8A 722B                <1>       jc     short loc_save_FAT_buff_write_err
34753                              <1>
34754 0000BF8C 807E0302            <1>       cmp    byte [esi+LD_FATType], 2
34755 0000BF90 7605                <1>       jna    short loc_calc_2nd_fat12_16_addr
34756                              <1>
34757                              <1> loc_calc_2nd_fat32_addr:
34758 0000BF92 8B462A              <1>       mov    eax, [esi+LD_BPB+FAT32_FAT_Size]
34759 0000BF95 EB04                <1>       jmp    short loc_calc_2nd_fat_addr
34760                              <1>
34761                              <1> loc_calc_2nd_fat12_16_addr:
34762 0000BF97 0FB7461C            <1>       movzx  eax, word [esi+LD_BPB+FATSecs]
34763                              <1>
34764                              <1> loc_calc_2nd_fat_addr:
34765 0000BF9B 034660              <1>       add    eax, [esi+LD_FATBegin]
34766 0000BF9E 0305[025B0100]      <1>       add    eax, [FAT_BuffSector]
34767 0000BFA4 BB001C0900          <1>       mov    ebx, FAT_Buffer
34768                              <1>       ; ecx = 1 to 3
34769 0000BFA9 E81B320000          <1>       call   disk_write
34770 0000BFAE 7207                <1>       jc     short loc_save_FAT_buff_write_err
34771                              <1>       ; Valid  buffer (1 = valid but do not save)
34772 0000BFB0 C605[FE5A0100]01    <1>       mov    byte [FAT_BuffValidData], 1
34773                              <1>
34774                              <1> loc_save_FAT_buff_write_err:
34775 0000BFB7 5E                  <1>       pop    esi ; *
34776 0000BFB8 BB001C0900          <1>       mov    ebx, FAT_Buffer
34777                              <1>       ; 15/10/2016 (1Dh -> 18)
34778                              <1>       ; 23/03/2016 (1Dh)
34779 0000BFBD B812000000          <1>       mov    eax, 18 ; Drive not ready or write error
34780 0000BFC2 C3                  <1>       retn
34781                              <1>
34782                              <1> calculate_fat_freespace:
34783                              <1>       ; 23/03/2016
34784                              <1>       ; 02/03/2016
34785                              <1>       ; 01/03/2016
34786                              <1>       ; 29/02/2016
34787                              <1>       ; 22/02/2016 (TRDOS 386 =  TRDOS v2.0)
34788                              <1>       ; 30/04/2011
34789                              <1>       ; 03/04/2010
34790                              <1>       ; 2005
34791                              <1>       ; INPUT ->
34792                              <1>       ;     EAX = Cluster count to be added or subtracted
34793                              <1>       ;     If BH = FFh, ESI = TR-DOS Logical Drive Description Table
34794                              <1>       ;     If BH < FFh, BH = TR-DOS Logical Drive Number
34795                              <1>       ;     BL:
34796                              <1>       ;     0 = Calculate, 1 = Add, 2 = Subtract, 3 = Get (Not Set/Calc)
34797                              <1>       ; OUTPUT ->
34798                              <1>       ;     EAX = Free Space in sectors
34799                              <1>       ;     ESI = Logical Dos Drive Description Table address
34800                              <1>       ;     BH = Logical Dos Drive Number (same with input value of BH)
34801                              <1>       ;     BL = Type of operation (same with input value of BL)
34802                              <1>       ;     ECX = 0 -> valid
34803                              <1>       ;     ECX > 0 -> error or invalid
34804                              <1>       ;     If EAX = FFFFFFFFh, it is 're-calculation needed'
34805                              <1>       ;                             sign due to r/w error
34806                              <1>
34807 0000BFC3 66891D[A25D0100]    <1>       mov    [CFS_OPType], bx
34808 0000BFCA A3[A45D0100]        <1>       mov    [CFS_CC], eax
34809                              <1>
```

```
34810 0000BFCF 80FFFF              <1>       cmp    bh, 0FFh
34811 0000BFD2 740B                <1>       je     short pass_calculate_freespace_get_drive_dt_offset
34812                              <1>
34813                              <1> loc_calculate_freespace_get_drive_dt_offset:
34814 0000BFD4 31C0                <1>       xor    eax, eax
34815 0000BFD6 88FC                <1>         mov     ah, bh
34816 0000BFD8 BE00010900          <1>       mov    esi, Logical_DOSDisks
34817 0000BFDD 01C6                <1>         add     esi, eax
34818                              <1>
34819                              <1> pass_calculate_freespace_get_drive_dt_offset:
34820 0000BFDF 08DB                <1>       or     bl, bl
34821 0000BFE1 7435                <1>       jz     short loc_reset_fcc
34822                              <1>
34823                              <1> loc_get_free_sectors:
34824 0000BFE3 8B4674              <1>       mov    eax, [esi+LD_FreeSectors]
34825                              <1>
34826                              <1>       ;xor   ecx, ecx
34827                              <1>       ;dec   ecx ; 0FFFFFFFFh
34828                              <1>       ;cmp   eax, ecx ; 29/02/2016
34829                              <1>       ;je    short loc_get_free_sectors_retn ; recalculation is needed!
34830                              <1>
34831                              <1>       ; 23/03/2016
34832 0000BFE6 8B4E70              <1>       mov    ecx, [esi+LD_TotalSectors]
34833 0000BFE9 39C1                <1>       cmp    ecx, eax ; Total sectors must be greater than Free sectors !
34834 0000BFEB 7707                <1>       ja     short loc_get_free_sectors_check_optype
34835                              <1>
34836 0000BFED 31C0                <1>       xor    eax, eax
34837 0000BFEF 48                  <1>       dec    eax ; 0FFFFFFFFh  ; recalculation is needed!
34838 0000BFF0 894674              <1>       mov    [esi+LD_FreeSectors], eax ; reset (for recalculation)
34839                              <1>
34840                              <1> loc_get_free_sectors_retn:
34841 0000BFF3 C3                  <1>       retn
34842                              <1>
34843                              <1> loc_get_free_sectors_check_optype:
34844 0000BFF4 80FB03              <1>       cmp    bl, 3
34845 0000BFF7 7203                <1>       jb     short loc_set_fcc
34846                              <1>
34847 0000BFF9 29C9                <1>       sub    ecx, ecx ; 0
34848                              <1>
34849 0000BFFB C3                  <1>       retn
34850                              <1>
34851                              <1> loc_set_fcc:
34852 0000BFFC 807E0302            <1>       cmp    byte [esi+LD_FATType], 2
34853 0000C000 0F87DF000000        <1>         ja      loc_update_FAT32_fs_info_fcc
34854                              <1>
34855                              <1>       ;mov   eax, [esi+LD_FreeSectors]
34856 0000C006 0FB64E13            <1>       movzx  ecx, byte [esi+LD_BPB+SecPerClust]
34857 0000C00A 29D2                <1>       sub    edx, edx
34858 0000C00C F7F1                <1>       div    ecx
34859                              <1>       ;or    dx, dx
34860                              <1>       ;      ; DX -> Remain sectors < SecPerClust
34861                              <1>       ;      ; DX > 0 -> invalid free sector count
34862                              <1>       ;jnz   short loc_reset_fcc
34863                              <1>
34864                              <1> ;pass_set_fcc_div32:
34865 0000C00E A3[1B5B0100]        <1>       mov    [FreeClusterCount], eax
34866 0000C013 E988000000          <1>         jmp     loc_set_free_sectors_FAT12_FAT16
34867                              <1>
34868                              <1> loc_reset_fcc:
34869 0000C018 31C0                <1>       xor    eax, eax
34870 0000C01A A3[1B5B0100]        <1>       mov    [FreeClusterCount], eax ; 0
34871 0000C01F 8B5678              <1>       mov    edx, [esi+LD_Clusters]
34872 0000C022 42                  <1>       inc    edx
34873 0000C023 8915[0A5B0100]      <1>       mov    [LastCluster], edx
34874                              <1>
34875 0000C029 807E0302            <1>       cmp    byte [esi+LD_FATType], 2
34876 0000C02D 7647                <1>       jna    short loc_count_free_fat_clusters_0
34877                              <1>
34878 0000C02F 48                  <1>       dec    eax ; FFFFFFFFh
34879 0000C030 A3[AC5D0100]        <1>       mov    [CFS_FAT32FC], eax
34880                              <1>
34881                              <1>       ; 29/02/2016
34882 0000C035 89463A              <1>       mov    [esi+LD_BPB+BPB_Reserved], eax ; reset
34883 0000C038 89463E              <1>       mov    [esi+LD_BPB+BPB_Reserved+4], eax ; reset
34884                              <1>
34885 0000C03B B802000000          <1>       mov    eax, 2
34886                              <1>
34887                              <1> loc_count_fc_next_cluster_0:
34888 0000C040 50                  <1>       push   eax
34889 0000C041 E801F9FFFF          <1>       call   get_next_cluster
34890 0000C046 7310                <1>       jnc    short loc_check_fat32_ff_cluster
34891 0000C048 09C0                <1>       or     eax, eax
34892 0000C04A 741E                <1>       jz     short pass_inc_cfs_fcc_0
34893                              <1>
34894                              <1> loc_put_fcc_unknown_sign:
34895 0000C04C 58                  <1>       pop    eax
34896                              <1>       ; "Free count is Unknown" sign
34897                              <1>       ;mov   dword [FreeClusterCount], 0FFFFFFFFh
34898                              <1>
34899                              <1>       ; 29/02/2016
34900                              <1>       ; Save Free Cluster Count value in FAT32 'BPB_Reserved' area
34901                              <1>       ;mov   [esi+LD_BPB+BPB_Reserved], 0FFFFFFFFh ; unknown!
34902 0000C04D 8B15[AC5D0100]      <1>       mov    edx, [CFS_FAT32FC] ; First Free Cluster
34903                              <1>       ; Save First Free Cluster value in FAT32 'BPB_Reserved+4' area
34904 0000C053 89563E              <1>       mov    [esi+LD_BPB+BPB_Reserved+4], edx
34905                              <1>
34906 0000C056 EB7D                <1>         jmp     loc_put_fcc_invalid_sign
34907                              <1>
34908                              <1> loc_check_fat32_ff_cluster:
34909 0000C058 09C0                <1>       or     eax, eax
34910 0000C05A 750E                <1>       jnz    short pass_inc_cfs_fcc_0
34911 0000C05C 58                  <1>       pop    eax
34912 0000C05D A3[AC5D0100]        <1>       mov    [CFS_FAT32FC], eax
```

```
                                <1>       ;mov   dword [FreeClusterCount], 1
0000C062 FF05[1B5B0100]         <1>       inc    dword [FreeClusterCount]
0000C068 EB27                   <1>       jmp    short pass_inc_cfs_fcc_1
                                <1>
                                <1> pass_inc_cfs_fcc_0:
0000C06A 58                     <1>       pop    eax
                                <1>
                                <1> pass_inc_cfs_fcc_0c:
0000C06B 40                     <1>       inc    eax ; add eax, 1
0000C06C 3B05[0A5B0100]         <1>       cmp    eax, [LastCluster]
0000C072 76CC                   <1>       jna    short loc_count_fc_next_cluster_0
0000C074 EB6F                   <1>       jmp    short loc_update_FAT32_fs_info_fcc
                                <1>
                                <1> loc_count_free_fat_clusters_0:
                                <1>       ;mov   eax, 2
0000C076 B002                   <1>       mov    al, 2
                                <1>
                                <1> loc_count_fc_next_cluster:
0000C078 50                     <1>       push   eax
0000C079 E8C9F8FFFF             <1>       call   get_next_cluster
0000C07E 720C                   <1>       jc     short loc_count_fcc_stc
                                <1>
                                <1> loc_count_free_clusters_1:
0000C080 21C0                   <1>       and    eax, eax
0000C082 750C                   <1>       jnz    short pass_inc_cfs_fcc
                                <1>
0000C084 FF05[1B5B0100]         <1>       inc    dword [FreeClusterCount]
0000C08A EB04                   <1>       jmp    short pass_inc_cfs_fcc
                                <1>
                                <1> loc_count_fcc_stc:
0000C08C 09C0                   <1>       or     eax, eax
0000C08E 75BC                   <1>       jnz    short loc_put_fcc_unknown_sign ; 29/02/2016
                                <1>
                                <1> pass_inc_cfs_fcc:
0000C090 58                     <1>       pop    eax
                                <1>
                                <1> pass_inc_cfs_fcc_1:
0000C091 40                     <1>       inc    eax ; add eax, 1
0000C092 3B05[0A5B0100]         <1>       cmp    eax, [LastCluster]
0000C098 76DE                   <1>       jna    short loc_count_fc_next_cluster
                                <1>
                                <1> loc_set_free_sectors:
0000C09A 807E0302               <1>       cmp    byte [esi+LD_FATType], 2
0000C09E 7745                   <1>       ja     short loc_update_FAT32_fs_info_fcc
                                <1>
                                <1> loc_set_free_sectors_FAT12_FAT16:
0000C0A0 803D[A25D0100]00       <1>       cmp    byte [CFS_OPType], 0
0000C0A7 761C                   <1>       jna    short pass_FAT_add_sub_fcc
0000C0A9 A1[A45D0100]           <1>       mov    eax, [CFS_CC]
0000C0AE 803D[A25D0100]01       <1>       cmp    byte [CFS_OPType], 1
0000C0B5 7708                   <1>       ja     short pass_FAT_add_fcc
0000C0B7 0105[1B5B0100]         <1>       add    [FreeClusterCount], eax
0000C0BD EB06                   <1>       jmp    short pass_FAT_add_sub_fcc
                                <1>
                                <1> pass_FAT_add_fcc:
0000C0BF 2905[1B5B0100]         <1>       sub    [FreeClusterCount], eax
                                <1>
                                <1> pass_FAT_add_sub_fcc:
0000C0C5 0FB64613               <1>       movzx  eax, byte [esi+LD_BPB+SecPerClust]
0000C0C9 8B15[1B5B0100]         <1>       mov    edx, [FreeClusterCount]
0000C0CF F7E2                   <1>       mul    edx
                                <1>
0000C0D1 31C9                   <1>       xor    ecx, ecx
0000C0D3 EB05                   <1>       jmp    short loc_cfs_retn_params
                                <1>
                                <1> loc_put_fcc_invalid_sign:
0000C0D5 29C0                   <1>                sub    eax, eax ; 0
0000C0D7 48                     <1>       dec    eax ; FFFFFFFFh
                                <1> loc_fat32_ffc_recalc_needed:
0000C0D8 89C1                   <1>       mov    ecx, eax
                                <1>
                                <1> loc_cfs_retn_params:
0000C0DA 894674                 <1>       mov    [esi+LD_FreeSectors], eax
0000C0DD 0FB71D[A25D0100]       <1>       movzx  ebx, word [CFS_OPType]
0000C0E4 C3                     <1>       retn
                                <1>
                                <1> loc_update_FAT32_fs_info_fcc:
                                <1> loc_check_fcc_FSINFO_op:
                                <1>       ; 29/02/2016
                                <1>       ; EAX = Free cluster count (before this update) ; value from disk
                                <1>       ; EDX = First Free Cluster (before this update) ; value from disk
0000C0E5 803D[A25D0100]01       <1>       cmp    byte [CFS_OPType], 1
0000C0EC 7221                   <1>       jb     short loc_cfs_FAT32_get_rcalc_parms ; 0 = recalculated
0000C0EE 7406                   <1>       je     short loc_check_fcc_FSINFO_op1 ; 1 = add
                                <1> loc_check_fcc_FSINFO_op2: ; subtract
0000C0F0 F71D[A45D0100]         <1>       neg    dword [CFS_CC] ; prepare to subtract ; 2 = sub (add negative)
                                <1> loc_check_fcc_FSINFO_op1:
                                <1>       ; 01/03/2016
0000C0F6 31D2                   <1>       xor    edx, edx ; 0
0000C0F8 4A                     <1>       dec    edx ; 0FFFFFFFFh
0000C0F9 8B463A                 <1>       mov    eax, [esi+LD_BPB+BPB_Reserved]
0000C0FC 39D0                   <1>       cmp    eax, edx
0000C0FE 73D5                   <1>       jnb    short loc_put_fcc_invalid_sign
0000C100 0305[A45D0100]         <1>         add     eax, [CFS_CC] ; free cluster count on disk + current count
0000C106 72CD                   <1>       jc     short loc_put_fcc_invalid_sign
                                <1>
0000C108 A3[1B5B0100]           <1>       mov    [FreeClusterCount], eax
0000C10D EB0E                   <1>       jmp    short loc_cfs_write_FSINFO_sector
                                <1>
                                <1> loc_cfs_FAT32_get_rcalc_parms:
0000C10F 8B15[AC5D0100]         <1>       mov    edx, [CFS_FAT32FC]
0000C115 A1[1B5B0100]           <1>       mov    eax, [FreeClusterCount]
0000C11A 89563E                 <1>       mov    [esi+LD_BPB+BPB_Reserved+4], edx ; First Free Cluster
```

```
35016                                  <1> loc_cfs_write_FSINFO_sector:
35017 0000C11D 89463A                  <1>       mov   [esi+LD_BPB+BPB_Reserved], eax ; Free cluster count
35018                                  <1>       ; 01/03/2016
35019 0000C120 E8AA000000              <1>       call  set_fat32_fsinfo_sector_parms
35020 0000C125 72AE                    <1>         jc      short loc_put_fcc_invalid_sign
35021                                  <1>
35022                                  <1> loc_set_FAT32_free_sectors:
35023                                  <1>       ; 29/02/2016
35024                                  <1>       ;mov   eax, [FreeClusterCount]
35025                                  <1>       ;mov   ecx, eax
35026                                  <1>       ;cmp   eax, 0FFFFFFFFh ; Invalid !
35027                                  <1>       ;je    short loc_cfs_retn_params
35028                                  <1>       ;
35029 0000C127 8B0D[1B5B0100]          <1>       mov   ecx, [FreeClusterCount]
35030 0000C12D 0FB64613                <1>       movzx  eax, byte [esi+LD_BPB+SecPerClust]
35031 0000C131 F7E1                    <1>       mul   ecx
35032                                  <1>       ; 29/02/2016
35033 0000C133 31C9                    <1>       xor   ecx, ecx ; 0
35034 0000C135 09D2                    <1>       or    edx, edx ; 0 ?
35035 0000C137 759C                    <1>         jnz     loc_put_fcc_invalid_sign
35036 0000C139 394670                  <1>       cmp   [esi+LD_TotalSectors], eax ; Volume size in sectors
35037 0000C13C 7697                    <1>         jna     short loc_put_fcc_invalid_sign
35038                                  <1>       ;
35039                                  <1> loc_set_FAT32_free_sectors_ok:
35040 0000C13E 31D2                    <1>       xor   edx, edx ; 0
35041 0000C140 EB98                    <1>         jmp     short loc_cfs_retn_params
35042                                  <1>       ;
35043                                  <1>
35044                                  <1> get_last_cluster:
35045                                  <1>       ; 22/10/2016
35046                                  <1>       ; 27/02/2016 (TRDOS 386 =  TRDOS v2.0)
35047                                  <1>       ; 12/06/2010 (DRV_FAT.ASM, 'proc_get_last_custer')
35048                                  <1>       ; 06/06/2010
35049                                  <1>       ; INPUT ->
35050                                  <1>       ;     EAX = First Cluster Number
35051                                  <1>       ;     ESI = Logical Dos Drive Parameters Table
35052                                  <1>       ; OUTPUT ->
35053                                  <1>       ;     cf = 0 -> No Error, EAX is valid
35054                                  <1>       ;     cf = 1 -> EAX > 0 -> Error
35055                                  <1>       ;     EAX = Last Cluster Number
35056                                  <1>       ;       ECX = Previous Cluster -just before the last cluster-
35057                                  <1>       ;       ; 22/10/2016
35058                                  <1>       ;     [glc_index] = cluster index number of the last cluster
35059                                  <1>       ;
35060                                  <1>       ; (Modified registers: EAX, ECX, EBX, EDX)
35061                                  <1>
35062 0000C142 89C1                    <1>       mov   ecx, eax
35063                                  <1>
35064 0000C144 C705[B45D0100]FFFF-     <1>       mov   dword [glc_index], 0FFFFFFFFh ; 22/10/2016
35065 0000C14C FFFF                    <1>
35066                                  <1>
35067                                  <1> loc_glc_get_next_cluster_1:
35068 0000C14E 890D[B05D0100]          <1>       mov   [glc_prevcluster], ecx
35069                                  <1>       ; 22/10/2016
35070 0000C154 FF05[B45D0100]          <1>       inc   dword [glc_index]
35071                                  <1>
35072                                  <1> loc_glc_get_next_cluster_2:
35073 0000C15A E8E8F7FFFF              <1>       call  get_next_cluster
35074                                  <1>       ; ecx = current/previous cluster
35075                                  <1>       ; eax = next/last cluster
35076 0000C15F 73ED                    <1>       jnc   short loc_glc_get_next_cluster_1
35077                                  <1>
35078 0000C161 09C0                    <1>       or    eax, eax
35079 0000C163 7509                    <1>       jnz   short loc_glc_stc_retn
35080                                  <1>
35081                                  <1>       ; ecx = previous cluster
35082 0000C165 89C8                    <1>         mov  eax, ecx
35083                                  <1>
35084                                  <1>       ; previous cluster becomes last cluster (ecx -> eax)
35085                                  <1>       ; previous of previous cluster becomes previous cluster (ecx)
35086                                  <1>
35087                                  <1> loc_glc_prev_cluster_retn:
35088 0000C167 8B0D[B05D0100]          <1>       mov   ecx, [glc_prevcluster]
35089 0000C16D C3                      <1>       retn
35090                                  <1>
35091                                  <1> loc_glc_stc_retn:
35092 0000C16E F5                      <1>       cmc   ;stc
35093 0000C16F EBF6                    <1>         jmp  short loc_glc_prev_cluster_retn
35094                                  <1>
35095                                  <1> truncate_cluster_chain:
35096                                  <1>       ; 01/03/2016
35097                                  <1>       ; 28/02/2016 (TRDOS 386 =  TRDOS v2.0)
35098                                  <1>       ; 22/01/2011 (DRV_FAT.ASM, 'proc_truncate_cluster_chain')
35099                                  <1>       ; 11/09/2010
35100                                  <1>       ; INPUT ->
35101                                  <1>       ;     ESI = Logical dos drive description table address
35102                                  <1>       ;     EAX = First cluster to be truncated/unlinked
35103                                  <1>       ; OUTPUT ->
35104                                  <1>       ;     ESI = Logical dos drive description table address
35105                                  <1>       ;     ECX = Count of truncated/removed clusters
35106                                  <1>       ;     CF = 0 -> EAX = Free sectors
35107                                  <1>       ;     CF = 1 -> Error code in EAX (AL)
35108                                  <1>
35109                                  <1>       ; NOTE: This procedure does not update lm date&time !
35110                                  <1>
35111                                  <1> loc_truncate_cc:
35112 0000C171 31C9                    <1>       xor   ecx, ecx ; mov ecx, 0
35113                                  <1>       ;mov   byte [FAT_BuffValidData], 0
35114 0000C173 890D[065B0100]          <1>       mov   [FAT_ClusterCounter], ecx ; 0 ; reset
35115                                  <1>
35116                                  <1> loc_tcc_unlink_clusters:
35117 0000C179 E8F3FAFFFF              <1>       call  update_cluster
35118                                  <1>       ; EAX = Next Cluster
```

```
35119                                  <1>       ; ECX = Cluster Value
35120                                  <1>       ; Note:
35121                                  <1>       ; Returns count of unlinked clusters in
35122                                  <1>       ; dword ptr FAT_ClusterCounter
35123 0000C17E 73F9                    <1>       jnc short loc_tcc_unlink_clusters
35124                                  <1>
35125                                  <1> pass_tcc_unlink_clusters:
35126 0000C180 A2[BB5D0100]            <1>       mov    byte [TCC_FATErr], al
35127 0000C185 803D[FE5A0100]02        <1>       cmp    byte [FAT_BuffValidData], 2
35128 0000C18C 750E                    <1>       jne    short loc_tcc_calculate_FAT_freespace
35129 0000C18E E89BFDFFFF              <1>       call   save_fat_buffer
35130 0000C193 7307                    <1>       jnc    short loc_tcc_calculate_FAT_freespace
35131 0000C195 A2[BB5D0100]            <1>       mov    byte [TCC_FATErr], al ; Error
35132                                  <1>       ;mov   byte [FAT_BuffValidData], 0
35133                                  <1>
35134                                  <1>       ; 01/03/2016
35135 0000C19A EB12                    <1>       jmp    short loc_tcc_recalculate_FAT_freespace
35136                                  <1>
35137                                  <1> loc_tcc_calculate_FAT_freespace:
35138 0000C19C A1[065B0100]            <1>       mov    eax, [FAT_ClusterCounter] ; signed (+-) number
35139 0000C1A1 66BB01FF                <1>       mov    bx, 0FF01h ; BH = FFh -> ESI = Dos drv desc. table
35140                                  <1>                         ; BL = 1 -> add cluster
35141 0000C1A5 E819FEFFFF              <1>       call   calculate_fat_freespace
35142 0000C1AA 21C9                    <1>       and    ecx, ecx ; cx = 0 -> valid free sector count
35143 0000C1AC 7409                    <1>       jz     short pass_truncate_cc_recalc_FAT_freespace
35144                                  <1>
35145                                  <1> loc_tcc_recalculate_FAT_freespace:
35146 0000C1AE 66BB00FF                <1>       mov    bx, 0FF00h ; recalculate !
35147 0000C1B2 E80CFEFFFF              <1>       call   calculate_fat_freespace
35148                                  <1>
35149                                  <1> loc_tcc_calculate_FAT_freespace_err:
35150                                  <1> pass_truncate_cc_recalc_FAT_freespace:
35151 0000C1B7 8B0D[065B0100]          <1>       mov    ecx, [FAT_ClusterCounter]
35152                                  <1>
35153 0000C1BD 803D[BB5D0100]00        <1>       cmp    byte [TCC_FATErr], 0
35154 0000C1C4 7608                    <1>       jna    short loc_tcc_unlink_clusters_retn
35155                                  <1>
35156                                  <1> loc_tcc_unlink_clusters_error:
35157 0000C1C6 0FB605[BB5D0100]        <1>       movzx  eax, byte [TCC_FATErr]
35158 0000C1CD F9                      <1>       stc
35159                                  <1> loc_tcc_unlink_clusters_retn:
35160 0000C1CE C3                      <1>       retn
35161                                  <1>
35162                                  <1> set_fat32_fsinfo_sector_parms:
35163                                  <1>       ; 15/10/2016
35164                                  <1>       ; 23/03/2016
35165                                  <1>       ; 29/02/2016 (TRDOS 386 =  TRDOS v2.0)
35166                                  <1>       ; INPUT ->
35167                                  <1>       ;     ESI = Logical dos drive description table address
35168                                  <1>       ;     [esi+LD_BPB+BPB_Reserved] = Free Cluster Count
35169                                  <1>       ;     [esi+LD_BPB+BPB_Reserved+4] = First Free Cluster
35170                                  <1>       ; OUTPUT ->
35171                                  <1>       ;     ESI = Logical dos drive description table address
35172                                  <1>       ;     CF = 0 -> OK..
35173                                  <1>       ;     CF = 1 -> Error code in EAX (AL)
35174                                  <1>       ;
35175                                  <1>       ; (Modified registers: EAX, EBX, ECX, EDX)
35176                                  <1>
35177 0000C1CF E824000000              <1>       call   get_fat32_fsinfo_sector_parms
35178 0000C1D4 7221                    <1>       jc     short update_fat32_fsinfo_sector_retn
35179                                  <1>
35180 0000C1D6 8B463A                  <1>       mov    eax, [esi+LD_BPB+BPB_Reserved] ; Free Cluster Count
35181 0000C1D9 8B563E                  <1>       mov    edx, [esi+LD_BPB+BPB_Reserved+4] ; First free Cluster
35182                                  <1>
35183                                  <1>         ;mov ebx, DOSBootSectorBuff
35184 0000C1DC 8983E8010000            <1>       mov    [ebx+488], eax
35185 0000C1E2 8993EC010000            <1>       mov    [ebx+492], edx
35186                                  <1>
35187 0000C1E8 A1[A85D0100]            <1>       mov    eax, [CFS_FAT32FSINFOSEC]
35188 0000C1ED B901000000              <1>       mov    ecx, 1
35189 0000C1F2 E8D22F0000              <1>       call   disk_write
35190                                  <1>       ;jnc     short update_fat32_fsinfo_sector_retn
35191                                  <1>
35192                                  <1>       ; 15/10/2016 (1Dh -> 18)
35193                                  <1>       ; 23/03/2016 (1Dh)
35194                                  <1>       ;mov   eax, 18 ; Drive not ready or write error
35195                                  <1>
35196                                  <1> update_fat32_fsinfo_sector_retn:
35197 0000C1F7 C3                      <1>       retn
35198                                  <1>
35199                                  <1> get_fat32_fsinfo_sector_parms:
35200                                  <1>       ; 15/10/2016
35201                                  <1>       ; 23/03/2016
35202                                  <1>       ; 01/03/2016
35203                                  <1>       ; 29/02/2016 (TRDOS 386 =  TRDOS v2.0)
35204                                  <1>       ; INPUT ->
35205                                  <1>       ;     ESI = Logical dos drive description table address
35206                                  <1>       ; OUTPUT ->
35207                                  <1>       ;     ESI = Logical dos drive description table address
35208                                  <1>       ;     EBX = FSINFO sector buffer address (DOSBootSectorBuff)
35209                                  <1>       ;     CF = 0 -> OK..
35210                                  <1>       ;        EAX = FsInfo sector address
35211                                  <1>       ;        ECX = Free cluster count
35212                                  <1>       ;        EDX = First free cluster
35213                                  <1>       ;     CF = 1 -> Error code in AL (EAX)
35214                                  <1>       ;        EBX = 0
35215                                  <1>       ;
35216                                  <1>       ;     [CFS_FAT32FSINFOSEC] = FAT32 FSINFO sector address
35217                                  <1>         ;
35218                                  <1>       ; (Modified registers: EAX, EBX, ECX, EDX)
35219                                  <1>
35220 0000C1F8 0FB74636                <1>       movzx  eax, word [esi+LD_BPB+FAT32_FSInfoSec]
35221 0000C1FC 03466C                  <1>       add    eax, [esi+LD_StartSector]
```

```
35222 0000C1FF A3[A85D0100]        <1>       mov    [CFS_FAT32FSINFOSEC], eax
35223                              <1>
35224 0000C204 BB[FA580100]        <1>         mov     ebx, DOSBootSectorBuff
35225 0000C209 B901000000          <1>       mov    ecx, 1
35226 0000C20E E8C52F0000          <1>       call   disk_read
35227 0000C213 7232                <1>       jc     short loc_read_FAT32_fsinfo_sec_err
35228                              <1>
35229 0000C215 BB[FA580100]        <1>       mov    ebx, DOSBootSectorBuff
35230                              <1>
35231 0000C21A 813B52526141        <1>       cmp    dword [ebx], 41615252h
35232 0000C220 751E                <1>       jne    short loc_read_FAT32_fsinfo_sec_stc
35233                              <1>
35234 0000C222 81BBE4010000727241- <1>       cmp    dword [ebx+484], 61417272h
35235 0000C22B 61                  <1>
35236 0000C22C 7512                <1>       jne    short loc_read_FAT32_fsinfo_sec_stc
35237                              <1>
35238 0000C22E A1[A85D0100]        <1>       mov    eax, [CFS_FAT32FSINFOSEC]
35239 0000C233 8B8BE8010000        <1>       mov    ecx, [ebx+488] ; free cluster count
35240 0000C239 8B93EC010000        <1>       mov    edx, [ebx+492] ; first (next) free cluster
35241                              <1>
35242 0000C23F C3                  <1>       retn
35243                              <1>
35244                              <1> loc_read_FAT32_fsinfo_sec_stc:
35245                              <1>       ; 15/10/2016 (0Bh -> 28)
35246 0000C240 B81C000000          <1>       mov    eax, 28 ; Invalid format!
35247 0000C245 EB05                <1>       jmp    short loc_read_FAT32_fsinfo_sec_stc_retn
35248                              <1>
35249                              <1> loc_read_FAT32_fsinfo_sec_err:
35250                              <1>       ; 15/10/2016 (15h -> 17)
35251                              <1>       ; 23/03/2016 (15h)
35252 0000C247 B811000000          <1>       mov    eax, 17 ; Drive not ready or read error
35253                              <1>
35254                              <1> loc_read_FAT32_fsinfo_sec_stc_retn:
35255 0000C24C 29DB                <1>       sub    ebx, ebx ; 0
35256 0000C24E F9                  <1>       stc
35257 0000C24F C3                  <1>       retn
35258                              <1>
35259                              <1> add_new_cluster:
35260                              <1>       ; 15/10/2016
35261                              <1>       ; 16/05/2016
35262                              <1>       ; 18/03/2016, 24/03/2016
35263                              <1>       ; 11/03/2016 (TRDOS 386 =  TRDOS v2.0)
35264                              <1>       ; 30/07/2011 (DRV_FAT.ASM)
35265                              <1>       ; 11/09/2010
35266                              <1>       ; INPUT ->
35267                              <1>       ;     ESI = Logical dos drv desc. table address
35268                              <1>       ;     EAX = Last cluster
35269                              <1>       ; OUTPUT ->
35270                              <1>       ;     ESI = Logical dos drv desc. table address
35271                              <1>       ;     EAX = New Last cluster (next cluster)
35272                              <1>       ;     cf = 1 -> error code in EAX (AL)
35273                              <1>       ;     cf = 1 -> DX = sectors per cluster
35274                              <1>       ;     ECX = Free sectors
35275                              <1>       ; NOTE:
35276                              <1>       ; This procedure does not update lm date&time !
35277                              <1>       ;
35278                              <1>       ; (Modified registers: EAX, EBX, ECX, EDX, EDI)
35279                              <1>       ;
35280                              <1>
35281 0000C250 A3[D85E0100]        <1>       mov    [FAT_anc_LCluster], eax
35282                              <1>
35283 0000C255 E844F9FFFF          <1>       call   get_first_free_cluster
35284 0000C25A 720B                <1>       jc     short loc_add_new_cluster_retn
35285                              <1>       ; EAX >= 2 and EAX < FFFFFFFFh is valid
35286                              <1>
35287 0000C25C 89C2                <1>       mov    edx, eax
35288                              <1>
35289 0000C25E 42                  <1>       inc    edx
35290                              <1>       ;jnz   short loc_add_new_cluster_check_ffc_eax
35291 0000C25F 7516                <1>       jnz    short loc_add_new_cluster_save_fcc
35292                              <1>
35293                              <1> loc_add_new_cluster_no_disk_space_retn:
35294 0000C261 B827000000          <1>       mov    eax, 27h ; MSDOS err => insufficient disk space
35295                              <1> loc_add_new_cluster_stc_retn:
35296 0000C266 F9                  <1>       stc
35297                              <1> loc_add_new_cluster_retn:
35298 0000C267 0FB65E13            <1>       movzx  ebx, byte [esi+LD_BPB+SecPerClust]
35299 0000C26B 8B4E74              <1>       mov    ecx, [esi+LD_FreeSectors]
35300                              <1>       ;xor   edx, edx
35301                              <1>       ;stc
35302 0000C26E C3                  <1>       retn
35303                              <1>
35304                              <1> loc_anc_invalid_format_stc_retn:
35305 0000C26F F9                  <1>       stc
35306                              <1> loc_add_new_cluster_invalid_format_retn:
35307                              <1>       ; 15/10/2016 (0Bh -> 28)
35308 0000C270 B81C000000          <1>       mov    eax, 28 ; Invalid format
35309 0000C275 EBF0                <1>       jmp    short loc_add_new_cluster_retn
35310                              <1>
35311                              <1> ;loc_add_new_cluster_check_ffc_eax:
35312                              <1> ;     cmp    eax, 2
35313                              <1> ;     jb     short loc_add_new_cluster_invalid_format_retn
35314                              <1>
35315                              <1> loc_add_new_cluster_save_fcc:
35316 0000C277 A3[DC5E0100]        <1>       mov    [FAT_anc_FFCluster], eax
35317                              <1>
35318 0000C27C 83E802              <1>       sub    eax, 2
35319 0000C27F 0FB65E13            <1>         movzx   ebx, byte [esi+LD_BPB+SecPerClust]
35320 0000C283 F7E3                <1>       mul    ebx
35321 0000C285 09D2                <1>       or     edx, edx
35322 0000C287 75E6                <1>       jnz    short loc_anc_invalid_format_stc_retn
35323                              <1>
35324                              <1> loc_add_new_cluster_allocate_cluster:
```

```
35325                                  <1>       ; 18/03/2016
35326 0000C289 92                      <1>       xchg   edx, eax ; eax = 0
35327                                  <1>       ; 16/05/2016
35328                                  <1>       ;cmp   [ClusterBuffer_Valid], al ; 0
35329                                  <1>       ;jna   short loc_anc_clear_cluster_buffer
35330                                  <1>       ;; 'copy' command,
35331                                  <1>       ;; writing destination file clust after reading source file clust
35332                                  <1>       ;mov   [ClusterBuffer_Valid], al ; 0 ; reset
35333                                  <1>       ;jmp   short loc_add_new_cluster_write_nc_to_disk
35334                                  <1>
35335                                  <1> loc_anc_clear_cluster_buffer:
35336                                  <1>       ; 11/03/2016
35337                                  <1>       ; Clear buffer
35338 0000C28A BF00000700              <1>       mov    edi, Cluster_Buffer ; 70000h (for current TRDOS 386 version)
35339 0000C28F 89D9                    <1>       mov    ecx, ebx ; sector count
35340 0000C291 C1E107                  <1>       shl    ecx, 7 ; 1 sector = 512 bytes -> 128 double words
35341                                  <1>       ;xor   eax, eax ; 0
35342 0000C294 F3AB                    <1>       rep    stosd
35343                                  <1>
35344                                  <1> loc_add_new_cluster_write_nc_to_disk:
35345                                  <1>       ; 11/03/2016
35346                                  <1>       ;xchg  eax, edx ; edx = 0, eax = sector offset
35347 0000C296 89D0                    <1>       mov    eax, edx
35348 0000C298 034668                  <1>         add     eax, [esi+LD_DATABegin]
35349 0000C29B 72D3                    <1>       jc     short loc_add_new_cluster_invalid_format_retn
35350                                  <1>
35351 0000C29D 89D9                    <1>       mov    ecx, ebx ; ECX = sectors per cluster (<256)
35352 0000C29F BB00000700              <1>       mov    ebx, Cluster_Buffer
35353 0000C2A4 E8202F0000              <1>       call   disk_write
35354 0000C2A9 7307                    <1>       jnc    short loc_add_new_cluster_update_fat_nlc
35355                                  <1>
35356                                  <1>       ; 15/10/2016 (1Dh -> 18)
35357 0000C2AB B812000000              <1>       mov    eax, 18 ; Write Error
35358 0000C2B0 EBB4                    <1>       jmp    short loc_add_new_cluster_stc_retn
35359                                  <1>
35360                                  <1> loc_add_new_cluster_update_fat_nlc:
35361 0000C2B2 A1[DC5E0100]            <1>       mov    eax, [FAT_anc_FFCluster]
35362 0000C2B7 31C9                    <1>       xor    ecx, ecx
35363 0000C2B9 890D[065B0100]          <1>       mov    [FAT_ClusterCounter], ecx ; 0 ; reset
35364 0000C2BF 49                      <1>       dec    ecx ; 0FFFFFFFFh
35365 0000C2C0 E8ACF9FFFF              <1>       call   update_cluster
35366 0000C2C5 7304                    <1>       jnc    short loc_add_new_cluster_update_fat_plc
35367 0000C2C7 09C0                    <1>       or     eax, eax ;EAX = 0 -> cluster value is 0 or eocc
35368 0000C2C9 759B                    <1>       jnz    short loc_add_new_cluster_stc_retn
35369                                  <1>
35370                                  <1> loc_add_new_cluster_update_fat_plc:
35371 0000C2CB A1[D85E0100]            <1>       mov    eax, [FAT_anc_LCluster]
35372 0000C2D0 8B0D[DC5E0100]          <1>       mov    ecx, [FAT_anc_FFCluster]
35373 0000C2D6 E896F9FFFF              <1>       call   update_cluster
35374 0000C2DB 7314                    <1>       jnc    short loc_add_new_cluster_save_fat_buffer
35375 0000C2DD 09C0                    <1>       or     eax, eax ; EAX = 0 -> cluster value is 0 or eocc
35376 0000C2DF 7410                    <1>       jz     short loc_add_new_cluster_save_fat_buffer
35377                                  <1>
35378                                  <1> loc_anc_save_fat_buffer_err_retn:
35379                                  <1>       ;cmp   byte [FAT_ClusterCounter], 1
35380                                  <1>       ;jb    short loc_add_new_cluster_retn
35381                                  <1>
35382 0000C2E1 66BB00FF               <1>       mov    bx, 0FF00h ; recalculate free space (BL = 0)
35383                                  <1>                         ; (BH = FFh -> Use ESI as Drv Param. Tbl.)
35384 0000C2E5 50                      <1>       push   eax
35385 0000C2E6 E8D8FCFFFF              <1>       call   calculate_fat_freespace
35386 0000C2EB 58                      <1>       pop    eax
35387 0000C2EC E975FFFFFF              <1>         jmp     loc_add_new_cluster_stc_retn
35388                                  <1>
35389                                  <1> loc_add_new_cluster_save_fat_buffer:
35390                                  <1>       ;cmp   byte [FAT_BuffValidData], 2
35391                                  <1>       ;jne   short loc_add_new_cluster_calc_FAT_freespace
35392                                  <1>       ;Byte [FAT_BuffValidData] =  2
35393 0000C2F1 E838FCFFFF              <1>       call   save_fat_buffer
35394 0000C2F6 72E9                    <1>       jc     short loc_anc_save_fat_buffer_err_retn
35395                                  <1>
35396                                  <1> loc_add_new_cluster_calc_FAT_freespace:
35397                                  <1>       ;mov   eax, 1 ; Only one Cluster
35398 0000C2F8 A1[065B0100]            <1>       mov    eax, [FAT_ClusterCounter]
35399 0000C2FD 66BB01FF                <1>       mov    bx, 0FF01h ; BH = FFh -> ESI -> Dos drv desc. table
35400                                  <1>                 ; BL = 1 -> add cluster
35401 0000C301 B301                    <1>       mov    bl, 01h ; BL = 1 -> add clusters
35402                                  <1>       ; NOTE: EAX value will be added to Free Cluster Count
35403                                  <1>       ; (Free Cluster Count is decreased when EAX value is negative)
35404 0000C303 E8BBFCFFFF              <1>         call    calculate_fat_freespace
35405                                  <1>       ;ECX = 0 -> no error, ECX > 0 -> error or invalid return
35406 0000C308 21C9                    <1>       and    ecx, ecx ; ECX = 0 -> valid free sector count
35407 0000C30A 7409                    <1>       jz     short loc_add_new_cluster_return_cluster_number
35408                                  <1>
35409                                  <1> loc_add_new_cluster_recalc_FAT_freespace:
35410 0000C30C 66BB00FF               <1>       mov    bx, 0FF00h  ; recalculate free space
35411 0000C310 E8AEFCFFFF              <1>         call    calculate_fat_freespace
35412                                  <1>       ; cf = 0
35413                                  <1> loc_add_new_cluster_return_cluster_number:
35414 0000C315 89C1                    <1>       mov    ecx, eax ; Free sector count
35415 0000C317 A1[DC5E0100]            <1>       mov    eax, [FAT_anc_FFCluster]
35416 0000C31C 0FB65E13                <1>       movzx  ebx, byte [esi+LD_BPB+SecPerClust]
35417                                  <1>       ;mov   edi, Cluster_Buffer
35418 0000C320 31D2                    <1>       xor    edx, edx
35419 0000C322 C3                      <1>         retn
35420                                  <1>
35421                                  <1> write_cluster:
35422                                  <1>       ; 15/10/2016
35423                                  <1>       ; 21/03/2016 (TRDOS 386 =  TRDOS v2.0)
35424                                  <1>       ;
35425                                  <1>       ; INPUT ->
35426                                  <1>       ;     EAX = Cluster Number (Sector index for SINGLIX FS)
35427                                  <1>       ;     ESI = Logical DOS Drive Description Table address
```

```
35428                           <1>       ;     EBX = Cluster (File R/W) Buffer address (max. 64KB)
35429                           <1>       ;     Only for SINGLIX FS:
35430                           <1>       ;     EDX = File Number (The 1st FDT address)
35431                           <1>       ; OUTPUT ->
35432                           <1>       ;     cf = 1 -> Cluster can not be written onto disk
35433                           <1>       ;         EAX > 0 -> Error number
35434                           <1>       ;     cf = 0 -> Cluster has been written successfully
35435                           <1>       ;
35436                           <1>       ; (Modified registers: EAX, ECX, EBX, EDX)
35437                           <1>
35438 0000C323 0FB64E13          <1>       movzx  ecx, byte [esi+LD_BPB+BPB_SecPerClust]
35439                           <1>       ; CL = 1 = [esi+LD_FS_Reserved2] ; SectPerClust for Singlix FS
35440                           <1>
35441                           <1> write_file_sectors: ; 16/03/2016
35442 0000C327 807E0300          <1>       cmp    byte [esi+LD_FATType], 0
35443 0000C32B 761C              <1>       jna    short write_fs_cluster
35444                           <1>
35445                           <1> write_fat_file_sectors:
35446 0000C32D 83E802            <1>       sub    eax, 2 ; Beginning cluster number is always 2
35447 0000C330 0FB65613          <1>       movzx  edx, byte [esi+LD_BPB+BPB_SecPerClust] ; 18/03/2016
35448 0000C334 F7E2              <1>       mul    edx
35449 0000C336 034668            <1>       add    eax, [esi+LD_DATABegin] ; absolute address of the cluster
35450                           <1>
35451                           <1>       ; EAX = Disk sector address
35452                           <1>       ; ECX = Sector count
35453                           <1>       ; EBX = Buffer address
35454                           <1>       ; (EDX = 0)
35455                           <1>       ; ESI = Logical DOS drive description table address
35456                           <1>
35457 0000C339 E88B2E0000        <1>       call   disk_write
35458 0000C33E 7306              <1>       jnc    short wclust_retn
35459                           <1>
35460                           <1>       ; 15/10/2016 (1Dh -> 18)
35461 0000C340 B812000000        <1>       mov    eax, 18 ; Drive not ready or write error !
35462 0000C345 C3                <1>       retn
35463                           <1>
35464                           <1> wclust_retn:
35465 0000C346 29C0              <1>       sub    eax, eax ; 0
35466 0000C348 C3                <1>       retn
35467                           <1>
35468                           <1> write_fs_cluster:
35469                           <1>       ; 21/03/2016 (TRDOS 386 =  TRDOS v2.0)
35470                           <1>       ; Singlix FS
35471                           <1>
35472                           <1>       ; EAX = Cluster number is sector index number of the file (eax)
35473                           <1>
35474                           <1>       ; EDX = File number is the first File Descriptor Table address
35475                           <1>       ;     of the file. (Absolute address of the FDT).
35476                           <1>
35477                           <1>       ; eax = sector index (0 for the first sector)
35478                           <1>       ; edx = FDT0 address
35479                           <1>             ; 64 KB buffer = 128 sectors (limit)
35480 0000C349 B980000000        <1>       mov    ecx, 128 ; maximum count of sectors (before eof)
35481 0000C34E E801000000        <1>       call   write_fs_sectors
35482 0000C353 C3                <1>       retn
35483                           <1>
35484                           <1> write_fs_sectors:
35485                           <1>       ; 21/03/2016 (TRDOS 386 =  TRDOS v2.0)
35486 0000C354 F9                <1>       stc
35487 0000C355 C3                <1>       retn
35488                           <1>
35489                           <1> get_cluster_by_index:
35490                           <1>       ; 29/04/2016 (TRDOS 386 =  TRDOS v2.0)
35491                           <1>       ; INPUT ->
35492                           <1>       ;     EAX = Beginning cluster
35493                           <1>       ;     EDX = Sector index in disk/file section
35494                           <1>       ;           (Only for SINGLIX file system!)
35495                           <1>       ;     ECX = Cluster sequence number after the beginning cluster
35496                           <1>       ;     ESI = Logical DOS Drive Description Table address
35497                           <1>       ; OUTPUT ->
35498                           <1>       ;     EAX = Cluster number
35499                           <1>       ;     cf = 1 -> Error code in AL (EAX)
35500                           <1>       ;
35501                           <1>       ;(Modified registers: EAX, ECX, EBX, EDX)
35502                           <1>       ;
35503 0000C356 807E0301          <1>       cmp    byte [esi+LD_FATType], 1
35504 0000C35A 721E              <1>         jb     short get_fs_section_by_index
35505                           <1>
35506 0000C35C 3B4E78            <1>       cmp    ecx, [esi+LD_Clusters]
35507 0000C35F 7207              <1>       jb     short gcbi_1
35508                           <1> gcbi_0:
35509 0000C361 F9                <1>       stc
35510 0000C362 B823000000        <1>       mov    eax, 23h ; Cluster not available !
35511                           <1>                       ; MSDOS error code: FCB unavailable
35512 0000C367 C3                <1>       retn
35513                           <1> gcbi_1:
35514 0000C368 51                <1>       push   ecx
35515 0000C369 E8D9F5FFFF        <1>       call   get_next_cluster
35516 0000C36E 59                <1>       pop    ecx
35517 0000C36F 7203              <1>       jc     short gcbi_3
35518 0000C371 E2F5              <1>       loop   gcbi_1
35519                           <1> gcbi_2:
35520 0000C373 C3                <1>       retn
35521                           <1> gcbi_3:
35522 0000C374 09C0              <1>       or     eax, eax
35523 0000C376 74E9              <1>       jz     short gcbi_0
35524 0000C378 F5                <1>       cmc    ; stc
35525 0000C379 C3                <1>       retn
35526                           <1>
35527                           <1> get_fs_section_by_index:
35528                           <1>       ; 29/04/2016 (TRDOS 386 =  TRDOS v2.0)
35529                           <1>       ; INPUT ->
35530                           <1>       ;     EAX = Beginning FDT number/address
```

```
35531                                  <1>       ;      EDX = Sector index in disk/file section
35532                                  <1>       ;      ECX = Sector sequence number after the beginning FDT
35533                                  <1>       ;      ESI = Logical DOS Drive Description Table address
35534                                  <1>       ; OUTPUT ->
35535                                  <1>       ;      EAX = FDT number/address
35536                                  <1>       ;      EDX = Sector index of the section (0,1,2,3,4...)
35537                                  <1>       ;      cf = 1 -> Error code in AL (EAX)
35538                                  <1>       ;
35539                                  <1>       ;(Modified registers: EAX, ECX, EBX, EDX)
35540                                  <1>       ;
35541 0000C37A B8FFFFFFFF               <1>       mov    eax, 0FFFFFFFFh
35542 0000C37F C3                       <1>       retn
35543                                  <1>
35544                                  <1> get_last_section:
35545                                  <1>       ; 22/10/2016 (TRDOS 386 =  TRDOS v2.0)
35546                                  <1>       ; INPUT ->
35547                                  <1>       ;      EAX = (The 1st) FDT number/address
35548                                  <1>       ;      ESI = Logical DOS Drive Description Table address
35549                                  <1>       ; OUTPUT ->
35550                                  <1>       ;      EAX = FDT number/address of the last section
35551                                  <1>       ;      EDX = Last sector of the section (0,1,2,3,4...)
35552                                  <1>       ;      [glc_index] = sector index number of the last sector
35553                                  <1>       ;                    (for file, not for the last section)
35554                                  <1>       ;
35555                                  <1>       ;      cf = 1 -> Error code in AL (EAX)
35556                                  <1>       ;
35557                                  <1>       ;(Modified registers: EAX, ECX, EBX, EDX)
35558                                  <1>       ;
35559 0000C380 B800000000               <1>       mov    eax, 0
35560 0000C385 BA00000000               <1>       mov    edx, 0
35561 0000C38A C3                       <1>       retn
35562                                      %include 'trdosk6.s' ; 24/01/2016
35563                                  <1> ; ****************************************************************************
35564                                  <1> ; TRDOS386.ASM (TRDOS 386 Kernel - v2.0.0) - MAIN PROGRAM : trdosk6.s
35565                                  <1> ; ----------------------------------------------------------------------------
35566                                  <1> ; Last Update: 16/11/2017
35567                                  <1> ; ----------------------------------------------------------------------------
35568                                  <1> ; Beginning: 24/01/2016
35569                                  <1> ; ----------------------------------------------------------------------------
35570                                  <1> ; Assembler: NASM version 2.11 (trdos386.s)
35571                                  <1> ; ----------------------------------------------------------------------------
35572                                  <1> ; Derived from 'Retro UNIX 386 Kernel - v0.2.1.0' source code by Erdogan Tan
35573                                  <1> ; u1.s (27/17/2015), u2.s (03/01/2016)
35574                                  <1> ; ****************************************************************************
35575                                  <1> ; Derived from TRDOS Operating System v1.0 (8086) source code by Erdogan Tan
35576                                  <1> ; TRDOS2.ASM (09/11/2011)
35577                                  <1> ; ----------------------------------------------------------------------------
35578                                  <1> ; INT_21H.ASM (c) 2009-2011 Erdogan TAN  [14/11/2009] Last Update: 08/11/2011
35579                                  <1>
35580                                  <1> sysent: ; < enter to system call >
35581                                  <1>       ; 17/03/2017
35582                                  <1>       ; 03/03/2017
35583                                  <1>       ; 19/02/2017
35584                                  <1>       ; 13/01/2017
35585                                  <1>       ; 06/06/2016
35586                                  <1>       ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
35587                                  <1>       ; 16/04/2015 - 19/10/2015 (Retro UNIX 386 v1)
35588                                  <1>       ; 10/04/2013 - 18/01/2014 (Retro UNIX 8086 v1)
35589                                  <1>       ;
35590                                  <1>       ; 'unkni' or 'sysent' is sytem entry from various traps.
35591                                  <1>       ; The trap type is determined and an indirect jump is made to
35592                                  <1>       ; the appropriate system call handler. If there is a trap inside
35593                                  <1>       ; the system a jump to panic is made. All user registers are saved
35594                                  <1>       ; and u.sp points to the end of the users stack. The sys (trap)
35595                                  <1>       ; instructor is decoded to get the the system code part (see
35596                                  <1>       ; trap instruction in the PDP-11 handbook) and from this
35597                                  <1>       ; the indirect jump address is calculated. If a bad system call is
35598                                  <1>       ; made, i.e., the limits of the jump table are exceeded, 'badsys'
35599                                  <1>       ; is called. If the call is legitimate control passes to the
35600                                  <1>       ; appropriate system routine.
35601                                  <1>       ;
35602                                  <1>       ; Calling sequence:
35603                                  <1>       ;      Through a trap caused by any sys call outside the system.
35604                                  <1>       ; Arguments:
35605                                  <1>       ;      Arguments of particular system call.
35606                                  <1>       ; ..........................................................
35607                                  <1>       ;
35608                                  <1>       ; Retro UNIX 8086 v1 modification:
35609                                  <1>       ;       System call number is in EAX register.
35610                                  <1>       ;
35611                                  <1>       ;       Other parameters are in EDX, EBX, ECX, ESI, EDI, EBP
35612                                  <1>       ;      registers depending of function details.
35613                                  <1>       ;
35614                                  <1>       ; 16/04/2015
35615 0000C38B 368925[5C030300]         <1>         mov     [ss:u.sp], esp ; Kernel stack points to return address
35616                                  <1>
35617                                  <1>       ; save user registers
35618 0000C392 1E                       <1>       push   ds
35619 0000C393 06                       <1>       push   es
35620 0000C394 0FA0                     <1>       push   fs
35621 0000C396 0FA8                     <1>       push   gs
35622 0000C398 60                       <1>       pushad  ; eax, ecx, edx, ebx, esp -before pushad-, ebp, esi, edi
35623                                  <1>       ;
35624                                  <1>       ; ESPACE = [ss:u.sp] - esp ; 4*12 = 48 ; 17/09/2015 ; 06/06/2016
35625                                  <1>       ;      (ESPACE is size of space in kernel stack
35626                                  <1>       ;      for saving/restoring user registers.)
35627                                  <1>       ;
35628 0000C399 50                       <1>       push   eax ; 01/07/2015
35629 0000C39A 66B81000                 <1>       mov     ax, KDATA
35630 0000C39E 8ED8                     <1>         mov     ds, ax
35631 0000C3A0 8EC0                     <1>         mov     es, ax
35632 0000C3A2 8EE0                     <1>         mov     fs, ax
35633 0000C3A4 8EE8                     <1>         mov     gs, ax
```

```
35634 0000C3A6 A1[20520100]        <1>       mov    eax, [k_page_dir]
35635 0000C3AB 0F22D8              <1>       mov    cr3, eax
35636 0000C3AE 58                  <1>       pop    eax ; 01/07/2015
35637                              <1>       ; 19/10/2015
35638 0000C3AF FC                  <1>       cld
35639                              <1>       ;
35640 0000C3B0 FE05[5B030300]      <1>       inc    byte [sysflg]
35641                              <1>              ; incb sysflg / indicate a system routine is in progress
35642 0000C3B6 FB                  <1>         sti  ; 18/01/2014
35643 0000C3B7 0F85F49FFFFF        <1>       jnz     panic ; 24/05/2013
35644                              <1>              ; beq 1f
35645                              <1>              ; jmp panic ; / called if trap inside system
35646                              <1> ;1:
35647                              <1>       ; 17/03/2017
35648 0000C3BD 80642438FE          <1>       and    byte [esp+ESPACE+8], ~1 ; clear carry flag
35649                              <1>
35650                              <1>       ; 16/04/2015
35651 0000C3C2 A3[64030300]        <1>       mov    [u.r0], eax
35652 0000C3C7 8925[60030300]      <1>       mov    [u.usp], esp ; kernel stack points to user's registers
35653                              <1>
35654                              <1>       ; 13/01/2017 (TRDOS 386 Feaure only !)
35655 0000C3CD 803D[D4030300]00    <1>       cmp    byte [u.t_lock], 0 ; timer interrupt lock ?
35656 0000C3D4 0F879D010000        <1>       ja     sysrele                ; yes, sys release only !!!
35657                              <1>
35658                              <1>              ; mov $s.syst+2,clockp
35659                              <1>              ; mov r0,-(sp) / save user registers
35660                              <1>              ; mov sp,u.r0 / pointer to bottom of users stack
35661                              <1>                      ; / in u.r0
35662                              <1>              ; mov r1,-(sp)
35663                              <1>              ; mov r2,-(sp)
35664                              <1>              ; mov r3,-(sp)
35665                              <1>              ; mov r4,-(sp)
35666                              <1>              ; mov r5,-(sp)
35667                              <1>              ; mov ac,-(sp) / "accumulator" register for extended
35668                              <1>                       ; / arithmetic unit
35669                              <1>              ; mov mq,-(sp) / "multiplier quotient" register for the
35670                              <1>                       ; / extended arithmetic unit
35671                              <1>              ; mov sc,-(sp) / "step count" register for the extended
35672                              <1>                       ; / arithmetic unit
35673                              <1>              ; mov sp,u.sp / u.sp points to top of users stack
35674                              <1>              ; mov 18.(sp),r0 / store pc in r0
35675                              <1>              ; mov -(r0),r0 / sys inst in r0      10400xxx
35676                              <1>              ; sub $sys,r0 / get xxx code
35677 0000C3DA C1E002             <1>       shl    eax, 2
35678                              <1>              ; asl r0 / multiply by 2 to jump indirect in bytes
35679 0000C3DD 3DB8000000         <1>       cmp    eax, end_of_syscalls - syscalls
35680                              <1>              ; cmp r0,$2f-1f / limit of table (35) exceeded
35681                              <1>       ;jnb   short badsys
35682                              <1>              ; bhis badsys / yes, bad system call
35683 0000C3E2 F5                  <1>       cmc
35684 0000C3E3 9C                  <1>       pushf
35685 0000C3E4 50                  <1>       push   eax
35686 0000C3E5 8B2D[5C030300]      <1>       mov    ebp, [u.sp] ; Kernel stack at the beginning of sys call
35687 0000C3EB B0FE                <1>       mov    al, 0FEh ; 11111110b
35688 0000C3ED 1400                <1>       adc    al, 0 ; al = al + cf
35689 0000C3EF 204508              <1>       and    [ebp+8], al ; flags (reset carry flag)
35690                              <1>              ; bic $341,20.(sp) / set users processor priority to 0
35691                              <1>                           ; / and clear carry bit
35692 0000C3F2 5D                  <1>       pop    ebp ; eax
35693 0000C3F3 9D                  <1>       popf
35694 0000C3F4 0F8208020000        <1>         jc      badsys
35695 0000C3FA A1[64030300]        <1>       mov    eax, [u.r0]
35696                              <1>       ; system call registers: EAX, EDX, ECX, EBX, ESI, EDI
35697 0000C3FF FFA5[05C40000]      <1>       jmp    dword [ebp+syscalls]
35698                              <1>              ; jmp *1f(r0) / jump indirect thru table of addresses
35699                              <1>                           ; / to proper system routine.
35700                              <1> syscalls: ; 1:
35701                              <1>       ; 28/02/2017
35702                              <1>       ; 20/02/2017
35703                              <1>       ; 19/02/2017
35704                              <1>       ; 15/10/2016
35705                              <1>       ; 20/05/2016
35706                              <1>       ; 19/05/2016
35707                              <1>       ; 16/05/2016
35708                              <1>       ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
35709                              <1>       ; 21/09/2015
35710                              <1>       ; 01/07/2015
35711                              <1>       ; 16/04/2015 (32 bit address modification)
35712                              <1>       ;dd sysrele  ; / 0
35713 0000C405 [E2E70000]          <1>       dd sysver    ; 0 ; Get TRDOS 386 version number (v2.0)
35714 0000C409 [64C60000]          <1>       dd sysexit   ; / 1
35715 0000C40D [39C80000]          <1>       dd sysfork   ; / 2
35716 0000C411 [6CCC0000]          <1>       dd sysread   ; / 3
35717 0000C415 [8BCC0000]          <1>       dd syswrite  ; / 4
35718 0000C419 [22CA0000]          <1>       dd sysopen   ; / 5
35719 0000C41D [43CC0000]          <1>       dd sysclose  ; / 6
35720 0000C421 [BBC70000]          <1>       dd syswait   ; / 7
35721 0000C425 [51C90000]          <1>       dd syscreat  ; / 8
35722 0000C429 [3AD80000]          <1>       dd syslink   ; / 9
35723 0000C42D [85D80000]          <1>       dd sysunlink ; / 10
35724 0000C431 [12D90000]          <1>       dd sysexec   ; / 11
35725 0000C435 [29DD0000]          <1>       dd syschdir  ; / 12
35726 0000C439 [08DE0000]          <1>       dd systime   ; / 13
35727 0000C43D [05CC0000]          <1>       dd sysmkdir  ; / 14
35728 0000C441 [7BDD0000]          <1>       dd syschmod  ; / 15
35729 0000C445 [D8DD0000]          <1>       dd syschown  ; / 16
35730 0000C449 [3BDE0000]          <1>       dd sysbreak  ; / 17
35731 0000C44D [D9DA0000]          <1>       dd sysstat   ; / 18
35732 0000C451 [80DE0000]          <1>       dd sysseek   ; / 19
35733 0000C455 [92DE0000]          <1>       dd systell   ; / 20
35734 0000C459 [7BDF0000]          <1>       dd sysmount  ; / 21
35735 0000C45D [8FDF0000]          <1>       dd sysumount ; / 22
35736 0000C461 [04DF0000]          <1>       dd syssetuid ; / 23
```

```
35737 0000C465 [35DF0000]          <1>       dd sysgetuid ; / 24
35738 0000C469 [17DE0000]          <1>       dd sysstime  ; / 25
35739 0000C46D [F8DE0000]          <1>       dd sysquit   ; / 26
35740 0000C471 [ECDE0000]          <1>       dd sysintr   ; / 27
35741 0000C475 [EDDA0000]          <1>       dd sysfstat  ; / 28
35742 0000C479 [22CD0000]          <1>       dd sysemt    ; / 29
35743 0000C47D [D3CE0000]          <1>       dd sysmdate  ; / 30
35744                              <1>       ;dd sysstty  ; / 31
35745 0000C481 [E7CE0000]          <1>       dd sysvideo  ; 31 ; TRDOS 386 Video Functions (16/05/2016)
35746                              <1>       ;dd sysgtty  ; / 32
35747 0000C485 [E9FA0000]          <1>       dd sysaudio  ; 32 ; TRDOS 386 Audio Functions (16/05/2016)
35748                              <1>       ;dd sysilgins; / 33
35749 0000C489 [3BCD0000]          <1>       dd systimer  ; 33 ; TRDOS 386 Timer Functions (18/05/2016)
35750 0000C48D [A3DF0000]          <1>       dd syssleep  ; 34 ; Retro UNIX 8086 v1 feature only !
35751                              <1>                          ; 11/06/2014
35752 0000C491 [D2DF0000]          <1>       dd sysmsg    ; 35 ; Retro UNIX 386 v1 feature only !
35753                              <1>                          ; 01/07/2015
35754 0000C495 [A8E00000]          <1>       dd sysgeterr ; 36 ; Retro UNIX 386 v1 feature only !
35755                              <1>                          ; 21/09/2015 - get last error number
35756 0000C499 [96F10000]          <1>       dd sysfpstat ; 37 ; TRDOS 386 FPU state option (28/02/2017)
35757 0000C49D [00E80000]          <1>       dd syspri    ; 38 ; change priority - TRDOS 386 (20/05/2016)
35758 0000C4A1 [77C50000]          <1>       dd sysrele   ; 39 ; TRDOS 386 (19/05/2016) (0 -> 39)
35759 0000C4A5 [33E90000]          <1>       dd sysfff    ; 40 ; Find First File - TRDOS 386 (15/10/2016)
35760 0000C4A9 [11EA0000]          <1>       dd sysfnf    ; 41 ; Find Next File - TRDOS 386 (15/10/2016)
35761 0000C4AD [81F00000]          <1>       dd sysalloc  ; 42 ; Allocate contiguous memory block/pages
35762                              <1>                          ; TRDOS 386 (19/02/2017) DMA buff fuctions
35763 0000C4B1 [3FF10000]          <1>       dd sysdalloc ; 43 ; Deallocate contiguous memory block/pages
35764                              <1>                          ; TRDOS 386 (19/02/2017) DMA buff fuctions
35765 0000C4B5 [7AF10000]          <1>       dd syscalbac ; 44 ; IRQ Callback and Signal Response Byte
35766                              <1>                          ; service setup - TRDOS 386 (20/02/2017)
35767                              <1>                          ; 28/08/2017 (20/08/2017)
35768 0000C4B9 [6D030100]          <1>       dd sysdma    ; 45 ; TRDOS 386 - (ISA) DMA service
35769                              <1>
35770                              <1> end_of_syscalls:
35771                              <1>
35772                              <1> error:
35773                              <1>       ; 18/05/2016
35774                              <1>       ; 13/05/2016
35775                              <1>       ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
35776                              <1>       ; 16/04/2015 - 17/09/2015 (Retro UNIX 386 v1)
35777                              <1>       ; 10/04/2013 - 07/08/2013 (Retro UNIX 8086 v1)
35778                              <1>       ;
35779                              <1>       ; 'error' merely sets the error bit off the processor status (c-bit)
35780                              <1>       ; then falls right into the 'sysret', 'sysrele' return sequence.
35781                              <1>       ;
35782                              <1>       ; INPUTS -> none
35783                              <1>       ; OUTPUTS ->
35784                              <1>       ;     processor status - carry (c) bit is set (means error)
35785                              <1>       ;
35786                              <1>       ; 26/05/2013 (Stack pointer must be reset here!
35787                              <1>       ;           Because, jumps to error procedure
35788                              <1>       ;           disrupts push-pop nesting balance)
35789                              <1>       ;
35790 0000C4BD 8B2D[5C030300]      <1>       mov    ebp, [u.sp] ; interrupt (system call) return (iretd) address
35791 0000C4C3 804D0801            <1>       or     byte [ebp+8], 1  ; set carry bit of flags register
35792                              <1>                               ; (system call will return with cf = 1)
35793                              <1>             ; bis $1,20.(r1) / set c bit in processor status word below
35794                              <1>                               ; / users stack
35795                              <1>       ; 17/09/2015
35796 0000C4C7 83ED30              <1>       sub    ebp, ESPACE ; 48 ; total size of stack frame ('sysdefs.inc')
35797                              <1>                               ; for saving/restoring user registers
35798                              <1>       ;cmp   ebp, [u.usp]
35799                              <1>       ;je    short err0
35800 0000C4CA 892D[60030300]      <1>       mov    [u.usp], ebp
35801                              <1> ;err0:
35802                              <1>       ; 01/09/2015
35803 0000C4D0 8B25[60030300]      <1>       mov    esp, [u.usp]     ; Retro Unix 8086 v1 modification!
35804                              <1>                               ; 10/04/2013
35805                              <1>                               ; (If an I/O error occurs during disk I/O,
35806                              <1>                               ; related procedures will jump to 'error'
35807                              <1>                               ; procedure directly without returning to
35808                              <1>                               ; the caller procedure. So, stack pointer
35809                              <1>                                     ; must be restored here.)
35810                              <1>       ; 13/05/2016
35811                              <1>       ; NOTE: (The last) error code is in 'u.error', it can be retrieved by
35812                              <1>       ;     'get last error' system call later.
35813                              <1>
35814                              <1>       ; 03/09/2015 - 09/06/2015 - 07/08/2013
35815 0000C4D6 C605[C6030300]00    <1>       mov    byte [u.kcall], 0 ; namei_r, mkdir_w reset
35816                              <1>
35817                              <1> sysret: ; < return from system call>
35818                              <1>       ; 01/03/2017
35819                              <1>       ; 28/02/2017
35820                              <1>       ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
35821                              <1>       ; 16/04/2015 - 10/09/2015 (Retro UNIX 386 v1)
35822                              <1>       ; 10/04/2013 - 23/02/2014 (Retro UNIX 8086 v1)
35823                              <1>       ;
35824                              <1>       ; 'sysret' first checks to see if process is about to be
35825                              <1>       ; terminated (u.bsys). If it is, 'sysexit' is called.
35826                              <1>       ; If not, following happens:
35827                              <1>       ;     1) The user's stack pointer is restored.
35828                              <1>       ;     2) r1=0 and 'iget' is called to see if last mentioned
35829                              <1>       ;        i-node has been modified. If it has, it is written out
35830                              <1>       ;        via 'ppoke'.
35831                              <1>       ;     3) If the super block has been modified, it is written out
35832                              <1>       ;        via 'ppoke'.
35833                              <1>       ;     4) If the dismountable file system's super block has been
35834                              <1>       ;        modified, it is written out to the specified device
35835                              <1>       ;        via 'ppoke'.
35836                              <1>       ;     5) A check is made if user's time quantum (uquant) ran out
35837                              <1>       ;        during his execution. If so, 'tswap' is called to give
35838                              <1>       ;        another user a chance to run.
35839                              <1>       ;     6) 'sysret' now goes into 'sysrele'.
```

```
35840                                  <1>       ;         (See 'sysrele' for conclusion.)
35841                                  <1>       ;
35842                                  <1>       ; Calling sequence:
35843                                  <1>       ;     jump table or 'br sysret'
35844                                  <1>       ; Arguments:
35845                                  <1>       ;     -
35846                                  <1>       ; ..........................................................
35847                                  <1>       ;
35848                                  <1>       ; ((AX=r1 for 'iget' input))
35849                                  <1>       ;
35850 0000C4DD 31C0                   <1>       xor   eax, eax ; 28/02/2017
35851                                  <1> sysret0: ; 29/07/2015 (eax = 0, jump from sysexec)
35852 0000C4DF FEC0                   <1>       inc   al ; 04/05/2013
35853 0000C4E1 3805[B2030300]         <1>       cmp   [u.bsys], al ; 1
35854                                  <1>             ; tstb u.bsys / is a process about to be terminated because
35855 0000C4E7 0F8377010000           <1>         jnb     sysexit ; 04/05/2013
35856                                  <1>             ; bne sysexit / of an error? yes, go to sysexit
35857                                  <1>       ;mov   esp, [u.usp] ; 24/05/2013 (that is not needed here)
35858                                  <1>             ; mov u.sp,sp / no point stack to users stack
35859 0000C4ED FEC8                   <1>       dec   al ; mov ax, 0
35860                                  <1>             ; clr r1 / zero r1 to check last mentioned i-node
35861 0000C4EF E8CB2C0000             <1>       call  iget
35862                                  <1>             ; jsr r0,iget / if last mentioned i-node has been modified
35863                                  <1>                           ; / it is written out
35864                                  <1>       ; 10/01/2017
35865                                  <1>       ; 09/01/2017
35866                                  <1> ;sysrele: ; < release >
35867                                  <1>       ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
35868                                  <1>       ; 16/04/2015 - 14/10/2015 (Retro UNIX 386 v1)
35869                                  <1>       ; 10/04/2013 - 07/03/2014 (Retro UNIX 8086 v1)
35870                                  <1>       ;
35871                                  <1>       ; 'sysrele' first calls 'tswap' if the time quantum for a user is
35872                                  <1>       ;  zero (see 'sysret'). It then restores the user's registers and
35873                                  <1>       ; turns off the system flag. It then checked to see if there is
35874                                  <1>       ; an interrupt from the user by calling 'isintr'. If there is,
35875                                  <1>       ; the output gets flashed (see isintr) and interrupt action is
35876                                  <1>       ; taken by a branch to 'intract'. If there is no interrupt from
35877                                  <1>       ; the user, a rti is made.
35878                                  <1>       ;
35879                                  <1>       ; Calling sequence:
35880                                  <1>       ;     Fall through a 'bne' in 'sysret' & ?
35881                                  <1>       ; Arguments:
35882                                  <1>       ;     -
35883                                  <1>       ; ..........................................................
35884                                  <1>       ;
35885                                  <1>       ; 23/02/2014 (swapret)
35886                                  <1>       ; 22/09/2013
35887                                  <1> sysrel0: ;1:
35888 0000C4F4 803D[A8030300]00       <1>       cmp   byte [u.quant], 0 ; 16/05/2013
35889                                  <1>             ; tstb uquant / is the time quantum 0?
35890 0000C4FB 7705                   <1>         ja      short swapret
35891                                  <1>             ; bne 1f / no, don't swap it out
35892                                  <1> sysrelease: ; 07/12/2013 (jump from 'clock')
35893 0000C4FD E8A8210000             <1>       call  tswap
35894                                  <1>             ; jsr r0,tswap / yes, swap it out
35895                                  <1>
35896                                  <1> ; Retro Unix 8086 v1 feature: return from 'swap' to 'swapret' address.
35897                                  <1> swapret: ;1:
35898                                  <1>       ; 10/09/2015
35899                                  <1>       ; 01/09/2015
35900                                  <1>       ; 14/05/2015
35901                                  <1>       ; 16/04/2015 (Retro UNIX 386 v1 - 32 bit, pm modifications)
35902                                  <1>       ; 26/05/2013 (Retro UNIX 8086 v1)
35903                                  <1>       ; cli
35904                                  <1>       ; 24/07/2015
35905                                  <1>       ;
35906                                  <1>       ;; 'esp' must be already equal to '[u.usp]' here !
35907                                  <1>       ;; mov esp, [u.usp]
35908                                  <1>
35909                                  <1>       ; 22/09/2013
35910 0000C502 E8B92C0000             <1>       call  isintr
35911                                  <1>       ; 20/10/2013
35912 0000C507 7405                   <1>       jz    short sysrel1
35913 0000C509 E83F010000             <1>       call  intract
35914                                  <1>             ; jsr r0,isintr / is there an interrupt from the user
35915                                  <1>             ;     br intract / yes, output gets flushed, take interrupt
35916                                  <1>                           ; / action
35917                                  <1> sysrel1:
35918 0000C50E FA                     <1>       cli   ; 14/10/2015
35919                                  <1> sysrel2:
35920                                  <1>       ; 28/02/2017
35921                                  <1>       ; Check if there is a (delayed) callback for current user/process
35922 0000C50F A0[D7030300]           <1>       mov   al, [u.irqwait]
35923 0000C514 240F                   <1>       and   al, 0Fh ; is there a waiting IRQ callback service ?
35924 0000C516 7444                   <1>       jz    short sysrel8 ; no
35925                                  <1>
35926                                  <1>       ; Set return to IRQ callback service and return from the service
35927 0000C518 0FB6D8                 <1>       movzx ebx, al
35928 0000C51B 883D[D7030300]         <1>       mov   [u.irqwait], bh ; 0 ; reset
35929 0000C521 8A9B[08100100]         <1>       mov   bl, [ebx+IRQenum] ; (available) IRQ index +1 (1 to 9)
35930                                  <1>       ; 01/03/2017
35931 0000C527 FECB                   <1>       dec   bl ; IRQ index number, 0 to 8
35932 0000C529 7831                   <1>       js    short sysrel8 ; 0 -> FFh (not in use!?)
35933                                  <1>       ;
35934 0000C52B A0[B3030300]           <1>       mov   al, [u.uno] ; current process (user) number
35935 0000C530 3883[56650100]         <1>       cmp   [ebx+IRQ.owner], al
35936 0000C536 7524                   <1>       jne   short sysrel8 ; it is not the current user/process !?
35937 0000C538 F683[68650100]01       <1>       test  byte [ebx+IRQ.method], 1 ; callback ?
35938 0000C53F 741B                   <1>       jz    short sysrel8 ; not a callback method !?
35939                                  <1>
35940 0000C541 8B93[7A650100]         <1>       mov   edx, [ebx+IRQ.addr] ; IRQ callback service address (virtual)
35941 0000C547 C605[D8030300]01       <1>       mov   byte [u.r_lock], 1 ; IRQ callback service in progress flag
35942                                  <1>
```

```
35943 0000C54E E8FF210000          <1>       call   wswap ; save user's registers & status
35944                              <1>                      ;      (for return from IRQ callback service)
35945                              <1>
35946 0000C553 8B2D[5C030300]      <1>       mov    ebp, [u.sp]; kernel's stack, points to EIP (user)
35947 0000C559 895500              <1>       mov    [ebp], edx ; IRQ call back service address
35948                              <1> sysrel8:
35949 0000C55C FE0D[5B030300]      <1>       dec    byte [sysflg]
35950                              <1>              ; decb sysflg / turn system flag off
35951                              <1>
35952 0000C562 A1[B8030300]        <1>       mov    eax, [u.pgdir]
35953 0000C567 0F22D8              <1>       mov    cr3, eax  ; 1st PDE points to Kernel Page Table 0 (1st 4 MB)
35954                              <1>                        ; (others are different than kernel page tables)
35955                              <1>       ; 10/09/2015
35956 0000C56A 61                  <1>       popad ; edi, esi, ebp, temp (icrement esp by 4), ebx, edx, ecx, eax
35957                              <1>              ; mov (sp)+,sc / restore user registers
35958                              <1>              ; mov (sp)+,mq
35959                              <1>              ; mov (sp)+,ac
35960                              <1>              ; mov (sp)+,r5
35961                              <1>              ; mov (sp)+,r4
35962                              <1>              ; mov (sp)+,r3
35963                              <1>              ; mov (sp)+,r2
35964                              <1>       ;
35965 0000C56B A1[64030300]        <1>       mov    eax, [u.r0]  ; ((return value in EAX))
35966 0000C570 0FA9                <1>       pop    gs
35967 0000C572 0FA1                <1>       pop    fs
35968 0000C574 07                  <1>       pop    es
35969 0000C575 1F                  <1>       pop    ds
35970                              <1>       ;or    word [esp+8], 200h ; 22/01/2017 ; force enabling interrupts
35971 0000C576 CF                  <1>       iretd
35972                              <1>              ; rti / no, return from interrupt
35973                              <1>
35974                              <1> sysrele:
35975                              <1>       ; 24/03/2017
35976                              <1>       ; 28/02/2017
35977                              <1>       ; 27/02/2017
35978                              <1>       ; 29/01/2017
35979                              <1>       ; 14/01/2017
35980                              <1>       ; 13/01/2017
35981                              <1>       ; 09/01/2017, 10/01/2017, 12/01/2017
35982                              <1>       ; Major modification for TRDOS 386 (CallBack return)
35983                              <1>       ;
35984                              <1>       ; 'sysrele' system call restores previously saved
35985                              <1>       ; registers and addresses of the process
35986                              <1>       ; (Main purpose -in TRDOS 386- is to return from
35987                              <1>       ; timer callback service routine in ring 3 -user mode-.)
35988                              <1>       ;
35989                              <1>       ; check if the process is in timer callback phase
35990 0000C577 803D[D4030300]00    <1>       cmp    byte [u.t_lock], 0 ; TIMER INT LOCK
35991                              <1>       ;je    short sysrel0 ; classic (Retro UNIX 386 type) sysrele
35992 0000C57E 7734                <1>       ja     short sysrel3
35993                              <1>       ; 27/02/2017
35994 0000C580 803D[D8030300]00    <1>       cmp    byte [u.r_lock], 0 ; IRQ callback lock
35995 0000C587 0F8667FFFFFF        <1>       jna    sysrel0 ; classic sysrele ; 24/03/2017
35996 0000C58D E859000000          <1>       call   sysrel7
35997 0000C592 803D[D8030300]00    <1>       cmp    byte [u.r_lock], 0 ; IRQ callback service lock
35998 0000C599 7628                <1>       jna    short sysrel4
35999 0000C59B C605[D8030300]00    <1>       mov    byte [u.r_lock], 0 ; reset
36000                              <1>       ;mov   byte [u.irqwait], 0 ; reset ; 28/02/2017
36001 0000C5A2 A0[D9030300]        <1>       mov    al, [u.r_mode]
36002 0000C5A7 08C0                <1>       or     al, al
36003 0000C5A9 7518                <1>       jnz    short sysrel4
36004 0000C5AB FEC8                <1>       dec    al
36005 0000C5AD A2[D9030300]        <1>       mov    [u.r_mode], al ; 0FFh ; not necessary !?
36006 0000C5B2 EB32                <1>       jmp    short sysrel6
36007                              <1> sysrel3:
36008                              <1>       ; 27/02/2017
36009 0000C5B4 E832000000          <1>       call   sysrel7
36010                              <1>       ; 14/01/2017
36011 0000C5B9 28C0                <1>       sub    al, al
36012 0000C5BB 3805[D4030300]      <1>       cmp    [u.t_lock], al ; 0 ; TIMER INT LOCK
36013 0000C5C1 770E                <1>       ja     short sysrel5 ; yes
36014                              <1> sysrel4:
36015                              <1>       ; 29/01/2017
36016 0000C5C3 8B44241C            <1>       mov    eax, [esp+28] ; eax
36017 0000C5C7 A3[64030300]        <1>       mov    [u.r0], eax
36018 0000C5CC E93EFFFFFF          <1>       jmp    sysrel2
36019                              <1> sysrel5:
36020 0000C5D1 A2[D4030300]        <1>       mov    [u.t_lock], al ; 0 ; reset
36021 0000C5D6 A0[D5030300]        <1>       mov    al, [u.t_mode]
36022 0000C5DB 20C0                <1>       and    al, al
36023                              <1>       ;jnz   short sysrel2 ; 0FFh ; user mode
36024 0000C5DD 75E4                <1>       jnz    short sysrel4 ; 29/01/2017
36025 0000C5DF FEC8                <1>       dec    al
36026 0000C5E1 A2[D5030300]        <1>       mov    [u.t_mode], al ; 0FFh ; not necessary !?
36027                              <1> sysrel6:
36028                              <1>       ; cpu will continue from the interrupted sytem call addr
36029 0000C5E6 61                  <1>       popad         ; edi, esi, ebp, esp, ebx, edx, ecx, eax
36030 0000C5E7 83C410              <1>       add    esp, 16      ; pass segment segisters: ds, es, fs, gs
36031 0000C5EA CF                  <1>       iretd         ; eip, cs, eflags
36032                              <1>
36033                              <1> sysrel7:
36034 0000C5EB 0FB61D[B3030300]    <1>       movzx  ebx, byte [u.uno] ; current process number
36035 0000C5F2 66C1E302            <1>       shl    bx, 2
36036                              <1>       ;cmp   [ebx+p.tcb-4], eax ; 0 ; is there callback address ?
36037                              <1>       ;jna   short sysrel0
36038                              <1>       ; yes, reset callback address then restore process registers
36039                              <1>       ;mov   [ebx+p.tcb-4], eax ; 0 ; reset
36040 0000C5F6 8B83[BC000300]      <1>       mov     eax, [ebx+p.upage-4] ; UPAGE address
36041 0000C5FC FA                  <1>       cli    ; disable interrupts till 'iretd'
36042 0000C5FD E988210000          <1>       jmp    rswap ; restore process 'u' structure
36043                              <1>
36044                              <1> badsys:
36045                              <1>       ; 25/12/2016
```

```
                                <1>       ; 18/04/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>       ; 17/04/2011 (TRDOS v1.0, 'IFC.ASM')
                                <1>       ; 03/02/2011 ('trdos_ifc_routine')
                                <1>       ;
                                <1>       ; 16/04/2015 (Retro UNIX 386 v1, 'badsys')
                                <1>       ; (EIP, EAX values will be shown on screen with error message)
                                <1>       ; (EIP = 'CD 40h' instruction address -INT 40h-)
                                <1>       ; (EAX = Function number)
                                <1>       ;
0000C602 FE05[B2030300]         <1>       inc    byte [u.bsys]
                                <1>       ;
0000C608 8B1D[5C030300]         <1>       mov    ebx, [u.sp] ; esp at the beginning of 'sysent'
0000C60E 8B03                   <1>       mov    eax, [ebx] ; EIP (return address, not 'INT 30h' address)
0000C610 83E802                 <1>       sub    eax, 2 ; CDh, ##h
0000C613 E8F16CFFFF             <1>       call   dwordtohex
0000C618 8915[E10D0100]         <1>       mov    [eip_str], edx
0000C61E A3[E50D0100]           <1>       mov    [eip_str+4], eax
0000C623 A1[64030300]           <1>       mov    eax, [u.r0]
0000C628 E8DC6CFFFF             <1>       call   dwordtohex
0000C62D 8915[D00D0100]         <1>       mov    [eax_str], edx
0000C633 A3[D40D0100]           <1>       mov    [eax_str+4], eax
                                <1>
0000C638 66C705[C50D0100]34-    <1>       mov    word [int_num_str], SYSCALL_INT_NUM ; 25/12/2016
0000C640 30                     <1>
                                <1>
0000C641 BE[970D0100]           <1>       mov    esi, ifc_msg ; "invalid funtion call !" msg (trdosk9.s)
0000C646 E8129DFFFF             <1>       call   print_msg
                                <1>
0000C64B EB17                   <1>       jmp    sysexit
                                <1>
                                <1> intract: ; / interrupt action
                                <1>       ; 14/10/2015
                                <1>       ; 16/04/2015 (Retro UNIX 386 v1 - Beginning)
                                <1>       ; 09/05/2013 - 07/12/2013 (Retro UNIX 8086 v1)
                                <1>       ;
                                <1>       ; Retro UNIX 8086 v1 modification !
                                <1>       ; (Process/task switching and quit routine by using
                                <1>       ; Retro UNIX 8086 v1 keyboard interrupt output.))
                                <1>       ;
                                <1>       ; input -> 'u.quit'  (also value of 'u.intr' > 0)
                                <1>       ; output -> If value of 'u.quit' = FFFFh ('ctrl+brk' sign)
                                <1>       ;            'intract' will jump to 'sysexit'.
                                <1>       ;          Intract will return to the caller
                                <1>       ;            if value of 'u.quit' <> FFFFh.
                                <1>       ; 14/10/2015
0000C64D FB                     <1>       sti
                                <1>       ; 07/12/2013
0000C64E 66FF05[AC030300]       <1>       inc    word [u.quit]
0000C655 7408                   <1>       jz     short intrct0 ; FFFFh -> 0
0000C657 66FF0D[AC030300]       <1>       dec    word [u.quit]
                                <1>       ; 16/04/2015
0000C65E C3                     <1>       retn
                                <1> intrct0:
0000C65F 58                     <1>       pop    eax ; call intract -> retn
                                <1>       ;
0000C660 31C0                   <1>       xor    eax, eax
0000C662 FEC0                   <1>       inc    al  ; mov ax, 1
                                <1> ;;;
                                <1>       ; UNIX v1 original 'intract' routine...
                                <1>       ; / interrupt action
                                <1>             ;cmp *(sp),$rti / are you in a clock interrupt?
                                <1>             ; bne 1f / no, 1f
                                <1>             ; cmp (sp)+,(sp)+ / pop clock pointer
                                <1>       ; 1: / now in user area
                                <1>             ; mov r1,-(sp) / save r1
                                <1>             ; mov u.ttyp,r1
                                <1>                   ; / pointer to tty buffer in control-to r1
                                <1>             ; cmpb 6(r1),$177
                                <1>                   ; / is the interrupt char equal to "del"
                                <1>             ; beq 1f / yes, 1f
                                <1>             ; clrb 6(r1)
                                <1>                   ; / no, clear the byte
                                <1>                  ; / (must be a quit character)
                                <1>             ; mov (sp)+,r1 / restore r1
                                <1>             ; clr u.quit / clear quit flag
                                <1>             ; bis $20,2(sp)
                                <1>                   ; / set trace for quit (sets t bit of
                                <1>                   ; / ps-trace trap)
                                <1>             ; rti   ;  / return from interrupt
                                <1>       ; 1: / interrupt char = del
                                <1>             ; clrb 6(r1) / clear the interrupt byte
                                <1>                     ; / in the buffer
                                <1>             ; mov (sp)+,r1 / restore r1
                                <1>             ; cmp u.intr,$core / should control be
                                <1>                         ; / transferred to loc core?
                                <1>             ; blo 1f
                                <1>             ; jmp *u.intr / user to do rti yes,
                                <1>                         ; / transfer to loc core
                                <1>       ; 1:
                                <1>             ; sys 1 / exit
                                <1>
                                <1> sysexit: ; <terminate process>
                                <1>       ; 14/11/2017
                                <1>       ; 27/05/2017
                                <1>       ; 10/04/2017
                                <1>       ; 26/02/2017, 28/02/2017
                                <1>       ; 02/01/2017, 23/01/2017
                                <1>       ; 06/06/2016, 10/06/2016
                                <1>       ; 19/05/2016, 23/05/2016
                                <1>       ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
                                <1>       ; 16/04/2015 - 01/09/2015 (Retro UNIX 386 v1)
                                <1>       ; 19/04/2013 - 14/02/2014 (Retro UNIX 8086 v1)
                                <1>       ;
```

```
                                <1>       ; 'sysexit' terminates a process. First each file that
                                <1>       ; the process has opened is closed by 'flose'. The process
                                <1>       ; status is then set to unused. The 'p.pid' table is then
                                <1>       ; searched to find children of the dying process. If any of
                                <1>       ; children are zombies (died by not waited for), they are
                                <1>       ; set free. The 'p.pid' table is then searched to find the
                                <1>       ; dying process's parent. When the parent is found, it is
                                <1>       ; checked to see if it is free or it is a zombie. If it is
                                <1>       ; one of these, the dying process just dies. If it is waiting
                                <1>       ; for a child process to die, it notified that it doesn't
                                <1>       ; have to wait anymore by setting it's status from 2 to 1
                                <1>       ; (waiting to active). It is awakened and put on runq by
                                <1>       ; 'putlu'. The dying process enters a zombie state in which
                                <1>       ; it will never be run again but stays around until a 'wait'
                                <1>       ; is completed by it's parent process. If the parent is not
                                <1>       ; found, process just dies. This means 'swap' is called with
                                <1>       ; 'u.uno=0'. What this does is the 'wswap' is not called
                                <1>       ; to write out the process and 'rswap' reads the new process
                                <1>       ; over the one that dies..i.e., the dying process is
                                <1>       ; overwritten and destroyed.
                                <1>       ;
                                <1>       ; Calling sequence:
                                <1>       ;     sysexit or conditional branch.
                                <1>       ; Arguments:
                                <1>       ;     -
                                <1>       ; ............................................................
                                <1>       ;
                                <1>       ; Retro UNIX 8086 v1 modification:
                                <1>       ;      System call number (=1) is in EAX register.
                                <1>       ;
                                <1>       ;      Other parameters are in EDX, EBX, ECX, ESI, EDI, EBP
                                <1>       ;      registers depending of function details.
                                <1>       ;
                                <1>       ; ('swap' procedure is mostly different than original UNIX v1.)
                                <1>       ;
                                <1> ; / terminate process
                                <1>       ; AX = 1
0000C664 6648                   <1>       dec   ax ; 0
0000C666 66A3[AA030300]         <1>       mov   [u.intr], ax ; 0
                                <1>             ; clr u.intr / clear interrupt control word
                                <1>             ; clr r1 / clear r1
                                <1> sysexit_0:
                                <1>       ; 23/01/2017
                                <1>       ; 02/01/2017
                                <1>       ; 10/06/2016
                                <1>       ; 06/06/2016
                                <1>       ; 23/05/2016
                                <1>       ; 19/05/2016 - TRDOS 386 (TRDOS v2.0)
                                <1>       ; Check and stop/clear timer event(s) of this (dying) process
                                <1>       ; if there is.
                                <1>
                                <1>       ; 02/01/2017
0000C66C FA                     <1>       cli   ; disable interrupts
                                <1>       ; 23/01/2017 - reset timer frequency (to 18.2Hz)
0000C66D B036                   <1>       mov   al, 00110110b ; 36h
0000C66F E643                   <1>       out   43h, al
0000C671 28C0                   <1>       sub   al, al ; 0
0000C673 E640                   <1>       out   40h, al ; LB
0000C675 E640                   <1>       out   40h, al ; HB
                                <1>       ;
0000C677 0FB61D[B3030300]       <1>       movzx ebx, byte [u.uno]
                                <1>       ;mov  bl, [u.uno] ; process number of dying process
0000C67E 3883[FF000300]         <1>       cmp   byte [ebx+p.timer-1], al ; 0
0000C684 763A                   <1>       jna   short sysexit_12 ; no timer events for this process
0000C686 8883[FF000300]         <1>       mov   byte [ebx+p.timer-1], al ; 0 ; reset
                                <1>       ;mov  al, [timer_events]
                                <1>       ;or   al, al
                                <1>       ;jz   short sysexit_12 ; no timer events
                                <1>       ;mov  cl, al
0000C68C 8A0D[B75F0100]         <1>       mov   cl, [timer_events] ; 14/11/2017
                                <1>       ;cli  ; disable interrupts
0000C692 B410                   <1>       mov   ah, 16 ; number of available timer events
0000C694 BE[60040300]           <1>       mov   esi, timer_set ; beginning address of timer events
                                <1> sysexit_7:
0000C699 8A06                   <1>       mov   al, [esi] ; process number (of timer event)
0000C69B 38D8                   <1>       cmp   al, bl ; process number comparison
0000C69D 7411                   <1>       je    short sysexit_10
0000C69F 20C0                   <1>       and   al, al
0000C6A1 7404                   <1>       jz    short sysexit_9
                                <1> sysexit_8:
0000C6A3 FEC9                   <1>       dec   cl
0000C6A5 7416                   <1>       jz    short sysexit_11
                                <1> sysexit_9:
0000C6A7 FECC                   <1>       dec   ah
0000C6A9 7415                   <1>       jz    short sysexit_12
0000C6AB 83C610                 <1>       add   esi, 16
0000C6AE EBE9                   <1>       jmp   short sysexit_7
                                <1>
                                <1> sysexit_10:
                                <1>       ;mov  byte [esi], 0
0000C6B0 66C7060000             <1>       mov   word [esi], 0
                                <1>       ;mov  dword [esi+12], 0
                                <1>       ;
0000C6B5 FE0D[B75F0100]         <1>       dec   byte [timer_events] ; 02/01/2017
                                <1>       ;
0000C6BB EBE6                   <1>       jmp   short sysexit_8
                                <1>
                                <1> sysexit_11:
0000C6BD 6629C0                 <1>       sub   ax, ax ; 0 ; 26/02/2017
                                <1> sysexit_12:
                                <1>       ; 26/02/2017 (Unlink IRQ callbacks belong to the user)
0000C6C0 803D[D6030300]00       <1>       cmp   byte [u.irqc], 0 ; Count of IRQ callbacks
0000C6C7 7E2E                   <1>       jng   short sysexit_16 ; zero or invalid
```

```
36252                                  <1>       ; 28/02/2017
36253                                  <1>       ; clear IRQ callback flags (for 'sysrele' and 'sysret')
36254 0000C6C9 A2[D7030300]            <1>       mov    [u.irqwait], al ; 0 ; force to clear waiting flag
36255 0000C6CE A2[D8030300]            <1>       mov    [u.r_lock], al ; 0 ; force to clear busy flag
36256 0000C6D3 BE[56650100]            <1>       mov    esi, IRQ.owner
36257                                  <1> sysexit_13:
36258 0000C6D8 AC                      <1>       lodsb
36259 0000C6D9 3A05[B3030300]          <1>       cmp    al, [u.uno] ; owner  = current user ?
36260 0000C6DF 750C                    <1>       jne    short sysexit_14
36261 0000C6E1 C646FF00                <1>       mov    byte [esi-1], 0 ; owner = 0 : Free
36262 0000C6E5 FE0D[D6030300]          <1>       dec    byte [u.irqc]
36263 0000C6EB 7408                    <1>       jz     short sysexit_15
36264                                  <1> sysexit_14:
36265 0000C6ED 81FE[5E650100]          <1>       cmp    esi, IRQ.owner + 8 ; the last IRQ index number ?
36266 0000C6F3 76E3                    <1>       jna    short sysexit_13 ; no
36267                                  <1> sysexit_15:
36268 0000C6F5 30C0                    <1>       xor    al, al ; 0
36269                                  <1> sysexit_16: ; 2:
36270 0000C6F7 FB                      <1>       sti    ; enable interrupts
36271                                  <1>       ;
36272                                  <1>       ; AX = 0
36273                                  <1> sysexit_1: ; 1:
36274                                  <1>       ; AX = File descriptor
36275                                  <1>             ; / r1 has file descriptor (index to u.fp list)
36276                                  <1>             ; / Search the whole list
36277 0000C6F8 E804140000              <1>       call   fclose
36278                                  <1>             ; jsr r0,fclose / close all files the process opened
36279                                  <1>       ;; ignore error return
36280                                  <1>             ; br .+2 / ignore error return
36281                                  <1>       ;inc   ax
36282 0000C6FD FEC0                    <1>       inc    al
36283                                  <1>             ; inc r1 / increment file descriptor
36284                                  <1>       ;cmp   ax, 10
36285 0000C6FF 3C0A                    <1>       cmp    al, 10
36286                                  <1>             ; cmp r1,$10. / end of u.fp list?
36287 0000C701 72F5                    <1>       jb     short sysexit_1
36288                                  <1>             ; blt 1b / no, go back
36289                                  <1>       ;movzx ebx, byte [u.uno]
36290 0000C703 8A1D[B3030300]          <1>       mov    bl, [u.uno] ; 02/01/2017
36291                                  <1>             ; movb u.uno,r1 / yes, move dying process's number to r1
36292 0000C709 88A3[AF000300]          <1>       mov    [ebx+p.stat-1], ah ; 0, SFREE
36293                                  <1>             ; clrb p.stat-1(r1) / free the process
36294                                  <1>       ; 10/04/2017
36295 0000C70F 381D[CD650100]          <1>       cmp    [audio_user], bl
36296 0000C715 7518                    <1>       jne    short sysexit_17
36297                                  <1>       ; reset audio device (current) owner and 'initializated' flag
36298 0000C717 883D[CD650100]          <1>       mov    [audio_user], bh ; 0
36299                                  <1>       ; 27/05/2017
36300 0000C71D 8B0D[B8650100]          <1>       mov    ecx,  [audio_buffer]
36301 0000C723 09C9                    <1>       or     ecx, ecx
36302 0000C725 7408                    <1>       jz     short sysexit_17
36303                                  <1>       ; 'deallocate_user_pages' is not necessary in sysexit !!!
36304                                  <1>       ;push  ebx
36305                                  <1>       ;mov   ebx, ecx
36306                                  <1>       ;mov   ecx, [audio_buff_size]
36307                                  <1>       ;call  deallocate_user_pages
36308                                  <1>       ;; (Modified Registers -> EAX, EDX, ESI, EDI, EBX, ECX, EBP)
36309 0000C727 29C9                    <1>       sub    ecx, ecx
36310 0000C729 890D[B8650100]          <1>       mov    [audio_buffer], ecx ; 0
36311                                  <1>       ;pop   ebx
36312                                  <1> sysexit_17:
36313                                  <1>       ;shl   bx, 1
36314 0000C72F D0E3                    <1>       shl    bl, 1
36315                                  <1>             ; asl r1 / use r1 for index into the below tables
36316 0000C731 668B8B[1E000300]        <1>       mov    cx, [ebx+p.pid-2]
36317                                  <1>             ; mov p.pid-2(r1),r3 / move dying process's name to r3
36318 0000C738 668B93[3E000300]        <1>       mov    dx, [ebx+p.ppid-2]
36319                                  <1>             ; mov p.ppid-2(r1),r4 / move its parents name to r4
36320                                  <1>       ; xor  bx, bx ; 0
36321 0000C73F 30DB                    <1>       xor    bl, bl ; 0
36322                                  <1>             ; clr r2
36323 0000C741 31F6                    <1>       xor    esi, esi ; 0
36324                                  <1>             ; clr r5 / initialize reg
36325                                  <1> sysexit_2: ; 1:
36326                                  <1>              ; / find children of this dying process,
36327                                  <1>             ; / if they are zombies, free them
36328                                  <1>       ;add   bx, 2
36329 0000C743 80C302                  <1>       add    bl, 2
36330                                  <1>             ; add $2,r2 / search parent process table
36331                                  <1>                       ; / for dying process's name
36332 0000C746 66398B[3E000300]        <1>       cmp    [ebx+p.ppid-2], cx
36333                                  <1>             ; cmp p.ppid-2(r2),r3 / found it?
36334 0000C74D 7513                    <1>       jne    short sysexit_4
36335                                  <1>             ; bne 3f / no
36336                                  <1>       ;shr   bx, 1
36337 0000C74F D0EB                    <1>       shr    bl, 1
36338                                  <1>             ; asr r2 / yes, it is a parent
36339 0000C751 80BB[AF000300]03        <1>       cmp    byte [ebx+p.stat-1], 3 ; SZOMB
36340                                  <1>             ; cmpb p.stat-1(r2),$3 / is the child of this
36341                                  <1>                                    ; / dying process a zombie
36342 0000C758 7506                    <1>       jne    short sysexit_3
36343                                  <1>             ; bne 2f / no
36344 0000C75A 88A3[AF000300]          <1>       mov    [ebx+p.stat-1], ah ; 0, SFREE
36345                                  <1>             ; clrb p.stat-1(r2) / yes, free the child process
36346                                  <1> sysexit_3: ; 2:
36347                                  <1>       ;shr   bx, 1
36348 0000C760 D0E3                    <1>       shl    bl, 1
36349                                  <1>             ; asl r2
36350                                  <1> sysexit_4: ; 3:
36351                                  <1>             ; / search the process name table
36352                                  <1>             ; / for the dying process's parent
36353 0000C762 663993[1E000300]        <1>       cmp    [ebx+p.pid-2], dx
36354                                  <1>             ; cmp p.pid-2(r2),r4 / found it?
```

```
36355 0000C769 7502                <1>       jne    short sysexit_5
36356                              <1>              ; bne 3f / no
36357 0000C76B 89DE                <1>       mov    esi, ebx
36358                              <1>              ; mov r2,r5 / yes, put index to p.pid table (parents
36359                              <1>                           ; / process # x2) in r5
36360                              <1> sysexit_5: ; 3:
36361                              <1>       ;cmp   bx, nproc + nproc
36362 0000C76D 80FB20              <1>       cmp    bl, nproc + nproc
36363                              <1>              ; cmp r2,$nproc+nproc / has whole table been searched?
36364 0000C770 72D1                <1>       jb     short sysexit_2
36365                              <1>              ; blt 1b / no, go back
36366                              <1>              ; mov r5,r1 / yes, r1 now has parents process # x2
36367 0000C772 21F6                <1>       and    esi, esi ; r5=r1
36368 0000C774 7436                <1>       jz     short sysexit_6
36369                              <1>              ; beq 2f / no parent has been found.
36370                              <1>                     ; / The process just dies
36371 0000C776 66D1EE              <1>       shr    si, 1
36372                              <1>              ; asr r1 / set up index to p.stat
36373 0000C779 8A86[AF000300]      <1>       mov    al, [esi+p.stat-1]
36374                              <1>              ; movb p.stat-1(r1),r2 / move status of parent to r2
36375 0000C77F 20C0                <1>       and    al, al
36376 0000C781 7429                <1>       jz     short sysexit_6
36377                              <1>              ; beq 2f / if its been freed, 2f
36378 0000C783 3C03                <1>       cmp    al, 3
36379                              <1>              ; cmp r2,$3 / is parent a zombie?
36380 0000C785 7425                <1>       je     short sysexit_6
36381                              <1>              ; beq 2f / yes, 2f
36382                              <1>       ; BH = 0
36383 0000C787 8A1D[B3030300]      <1>       mov    bl, [u.uno]
36384                              <1>              ; movb u.uno,r3 / move dying process's number to r3
36385 0000C78D C683[AF000300]03    <1>       mov    byte [ebx+p.stat-1], 3  ; SZOMB
36386                              <1>              ; movb $3,p.stat-1(r3) / make the process a zombie
36387 0000C794 3C01                <1>       cmp    al, 1 ; SRUN
36388 0000C796 7414                <1>       je     short sysexit_6
36389                              <1>       ;cmp   al, 2
36390                              <1>              ; cmp r2,$2 / is the parent waiting for
36391                              <1>                      ; / this child to die
36392                              <1>       ;jne   short sysexit_6
36393                              <1>              ; bne 2f / yes, notify parent not to wait any more
36394                              <1>       ; p.stat = 2 --> waiting
36395                              <1>       ; p.stat = 4 --> sleeping
36396 0000C798 C686[AF000300]01    <1>       mov    byte [esi+p.stat-1], 1 ; SRUN
36397                              <1>       ;dec   byte [esi+p.stat-1]
36398                              <1>              ; decb p.stat-1(r1) / awaken it by putting it (parent)
36399 0000C79F 6689F0              <1>       mov    ax, si ; r1  (process number in AL)
36400                              <1>       ;
36401                              <1>       ;mov   ebx, runq + 4
36402                              <1>              ; mov $runq+4,r2 / on the runq
36403 0000C7A2 BB[54030300]        <1>       mov    ebx, runq+2 ; normal run queue ; 02/01/2017
36404 0000C7A7 E816200000          <1>       call   putlu
36405                              <1>              ; jsr r0, putlu
36406                              <1> sysexit_6:
36407                              <1>              ; / the process dies
36408 0000C7AC C605[B3030300]00    <1>       mov    byte [u.uno], 0
36409                              <1>              ; clrb u.uno / put zero as the process number,
36410                              <1>                  ; / so "swap" will
36411 0000C7B3 E80C1F0000          <1>       call   swap
36412                              <1>              ; jsr r0,swap / overwrite process with another process
36413                              <1> hlt_sys:
36414                              <1>       ;sti
36415                              <1> hlts0:
36416 0000C7B8 F4                  <1>       hlt
36417 0000C7B9 EBFD                <1>       jmp    short hlts0
36418                              <1>              ; 0 / and thereby kill it; halt?
36419                              <1>
36420                              <1> syswait: ; < wait for a processs to die >
36421                              <1>       ; 17/09/2015
36422                              <1>       ; 02/09/2015
36423                              <1>       ; 01/09/2015
36424                              <1>       ; 16/04/2015 (Retro UNIX 386 v1 - Beginning)
36425                              <1>       ; 24/05/2013 - 05/02/2014 (Retro UNIX 8086 v1)
36426                              <1>       ;
36427                              <1>       ; 'syswait' waits for a process die.
36428                              <1>       ; It works in following way:
36429                              <1>       ;   1) From the parent process number, the parent's
36430                              <1>       ;     process name is found. The p.ppid table of parent
36431                              <1>       ;     names is then searched for this process name.
36432                              <1>       ;     If a match occurs, r2 contains child's process
36433                              <1>       ;     number. The child status is checked to see if it is
36434                              <1>       ;     a zombie, i.e; dead but not waited for (p.stat=3)
36435                              <1>       ;     If it is, the child process is freed and it's name
36436                              <1>       ;     is put in (u.r0). A return is then made via 'sysret'.
36437                              <1>       ;     If the child is not a zombie, nothing happens and
36438                              <1>       ;     the search goes on through the p.ppid table until
36439                              <1>       ;     all processes are checked or a zombie is found.
36440                              <1>       ;   2) If no zombies are found, a check is made to see if
36441                              <1>       ;     there are any children at all. If there are none,
36442                              <1>       ;     an error return is made. If there are, the parent's
36443                              <1>       ;     status is set to 2 (waiting for child to die),
36444                              <1>       ;     the parent is swapped out, and a branch to 'syswait'
36445                              <1>       ;     is made to wait on the next process.
36446                              <1>       ;
36447                              <1>       ; Calling sequence:
36448                              <1>       ;     ?
36449                              <1>       ; Arguments:
36450                              <1>       ;     -
36451                              <1>       ; Inputs: -
36452                              <1>       ; Outputs: if zombie found, it's name put in u.r0.
36453                              <1>       ; ..........................................................
36454                              <1>       ;
36455                              <1>
36456                              <1> ; / wait for a process to die
36457                              <1>
```

```
36458                                  <1> syswait_0:
36459 0000C7BB 0FB61D[B3030300]        <1>       movzx  ebx, byte [u.uno] ; 01/09/2015
36460                                  <1>              ; movb u.uno,r1 / put parents process number in r1
36461 0000C7C2 D0E3                    <1>       shl    bl, 1
36462                                  <1>       ;shl   bx, 1
36463                                  <1>              ; asl r1 / x2 to get index into p.pid table
36464 0000C7C4 668B83[1E000300]        <1>       mov    ax, [ebx+p.pid-2]
36465                                  <1>              ; mov p.pid-2(r1),r1 / get the name of this process
36466 0000C7CB 31F6                    <1>       xor    esi, esi
36467                                  <1>              ; clr r2
36468 0000C7CD 31C9                    <1>       xor    ecx, ecx ; 30/10/2013
36469                                  <1>       ;xor   cl, cl
36470                                  <1>              ; clr r3 / initialize reg 3
36471                                  <1> syswait_1: ; 1:
36472 0000C7CF 6683C602                <1>       add    si, 2
36473                                  <1>              ; add $2,r2 / use r2 for index into p.ppid table
36474                                  <1>                      ; / search table of parent processes
36475                                  <1>                      ; / for this process name
36476 0000C7D3 663B86[3E000300]        <1>       cmp    ax, [esi+p.ppid-2]
36477                                  <1>              ; cmp p.ppid-2(r2),r1 / r2 will contain the childs
36478                                  <1>                                   ; / process number
36479 0000C7DA 7535                    <1>       jne    short syswait_3
36480                                  <1>              ;bne 3f / branch if no match of parent process name
36481                                  <1>       ;inc   cx
36482 0000C7DC FEC1                    <1>       inc    cl
36483                                  <1>              ;inc r3 / yes, a match, r3 indicates number of children
36484 0000C7DE 66D1EE                  <1>       shr    si, 1
36485                                  <1>              ; asr r2 / r2/2 to get index to p.stat table
36486                                  <1>       ; The possible states ('p.stat' values) of a process are:
36487                                  <1>       ;      0 = free or unused
36488                                  <1>       ;      1 = active
36489                                  <1>       ;      2 = waiting for a child process to die
36490                                  <1>       ;      3 = terminated, but not yet waited for (zombie).
36491 0000C7E1 80BE[AF000300]03        <1>       cmp    byte [esi+p.stat-1], 3 ; SZOMB, 05/02/2014
36492                                  <1>              ; cmpb p.stat-1(r2),$3 / is the child process a zombie?
36493 0000C7E8 7524                    <1>       jne    short syswait_2
36494                                  <1>              ; bne 2f / no, skip it
36495 0000C7EA 88BE[AF000300]          <1>       mov    [esi+p.stat-1], bh ; 0
36496                                  <1>              ; clrb p.stat-1(r2) / yes, free it
36497 0000C7F0 66D1E6                  <1>       shl    si, 1
36498                                  <1>              ; asl r2 / r2x2 to get index into p.pid table
36499 0000C7F3 0FB786[1E000300]        <1>       movzx  eax, word [esi+p.pid-2]
36500 0000C7FA A3[64030300]            <1>       mov    [u.r0], eax
36501                                  <1>              ; mov p.pid-2(r2),*u.r0
36502                                  <1>                           ; / put childs process name in (u.r0)
36503                                  <1>       ;
36504                                  <1>       ; Retro UNIX 386 v1 modification ! (17/09/2015)
36505                                  <1>       ;
36506                                  <1>       ; Parent process ID -p.ppid- field (of the child process)
36507                                  <1>       ; must be cleared in order to prevent infinitive 'syswait'
36508                                  <1>       ; system call loop from the application/program if it calls
36509                                  <1>       ; 'syswait' again (mistakenly) while there is not a zombie
36510                                  <1>       ; or running child process to wait. ('forktest.s', 17/09/2015)
36511                                  <1>       ;
36512                                  <1>       ; Note: syswait will return with error if there is not a
36513                                  <1>       ;       zombie or running process to wait.
36514                                  <1>       ;
36515 0000C7FF 6629C0                  <1>       sub    ax, ax
36516 0000C802 668986[3E000300]        <1>       mov    [esi+p.ppid-2], ax ; 0 ; 17/09/2015
36517 0000C809 E9D1FCFFFF              <1>       jmp    sysret0 ; ax = 0
36518                                  <1>       ;
36519                                  <1>       ;jmp   sysret
36520                                  <1>              ; br sysret1 / return cause child is dead
36521                                  <1> syswait_2: ; 2:
36522 0000C80E 66D1E6                  <1>       shl    si, 1
36523                                  <1>              ; asl r2 / r2x2 to get index into p.ppid table
36524                                  <1> syswait_3: ; 3:
36525 0000C811 6683FE20                <1>       cmp    si, nproc+nproc
36526                                  <1>              ; cmp r2,$nproc+nproc / have all processes been checked?
36527 0000C815 72B8                    <1>       jb     short syswait_1
36528                                  <1>              ; blt 1b / no, continue search
36529                                  <1>       ;and   cx, cx
36530 0000C817 20C9                    <1>       and    cl, cl
36531                                  <1>              ; tst r3 / one gets here if there are no children
36532                                  <1>                     ; / or children that are still active
36533                                  <1>       ; 30/10/2013
36534 0000C819 750B                    <1>       jnz    short syswait_4
36535                                  <1>       ;jz    error
36536                                  <1>              ; beq error1 / there are no children, error
36537 0000C81B 890D[64030300]          <1>       mov    [u.r0], ecx ; 0
36538 0000C821 E997FCFFFF              <1>       jmp    error
36539                                  <1> syswait_4:
36540 0000C826 8A1D[B3030300]          <1>       mov    bl, [u.uno]
36541                                  <1>              ; movb u.uno,r1 / there are children so put
36542                                  <1>                           ; / parent process number in r1
36543 0000C82C FE83[AF000300]          <1>       inc    byte [ebx+p.stat-1] ; 2, SWAIT, 05/02/2014
36544                                  <1>              ; incb p.stat-1(r1) / it is waiting for
36545                                  <1>                             ; / other children to die
36546                                  <1>       ; 04/11/2013
36547 0000C832 E88D1E0000              <1>       call   swap
36548                                  <1>              ; jsr r0,swap / swap it out, because it's waiting
36549 0000C837 EB82                    <1>       jmp    syswait_0
36550                                  <1>              ; br syswait / wait on next process
36551                                  <1>
36552                                  <1> sysfork: ; < create a new process >
36553                                  <1>       ; 02/01/2017 (TRDOS 386 modification)
36554                                  <1>       ; 04/09/2015, 18/05/2015
36555                                  <1>       ; 28/08/2015, 01/09/2015, 02/09/2015
36556                                  <1>       ; 09/05/2015, 10/05/2015, 14/05/2015
36557                                  <1>       ; 06/05/2015 (Retro UNIX 386 v1 - Beginning)
36558                                  <1>       ; 24/05/2013 - 14/02/2014 (Retro UNIX 8086 v1)
36559                                  <1>       ;
36560                                  <1>       ; 'sysfork' creates a new process. This process is referred
```

```
36561                                  <1>       ; to as the child process. This new process core image is
36562                                  <1>       ; a copy of that of the caller of 'sysfork'. The only
36563                                  <1>       ; distinction is the return location and the fact that (u.r0)
36564                                  <1>       ; in the old process (parent) contains the process id (p.pid)
36565                                  <1>       ; of the new process (child). This id is used by 'syswait'.
36566                                  <1>       ; 'sysfork' works in the following manner:
36567                                  <1>       ;    1) The process status table (p.stat) is searched to find
36568                                  <1>       ;       a process number that is unused. If none are found
36569                                  <1>       ;       an error occurs.
36570                                  <1>       ;    2) when one is found, it becomes the child process number
36571                                  <1>       ;       and it's status (p.stat) is set to active.
36572                                  <1>       ;    3) If the parent had a control tty, the interrupt
36573                                  <1>       ;       character in that tty buffer is cleared.
36574                                  <1>       ;    4) The child process is put on the lowest priority run
36575                                  <1>       ;       queue via 'putlu'.
36576                                  <1>       ;    5) A new process name is gotten from 'mpid' (actually
36577                                  <1>       ;       it is a unique number) and is put in the child's unique
36578                                  <1>       ;       identifier; process id (p.pid).
36579                                  <1>       ;    6) The process name of the parent is then obtained and
36580                                  <1>       ;       placed in the unique identifier of the parent process
36581                                  <1>       ;       name is then put in 'u.r0'.
36582                                  <1>       ;    7) The child process is then written out on disk by
36583                                  <1>       ;       'wswap',i.e., the parent process is copied onto disk
36584                                  <1>       ;       and the child is born. (The child process is written
36585                                  <1>       ;       out on disk/drum with 'u.uno' being the child process
36586                                  <1>       ;       number.)
36587                                  <1>       ;    8) The parent process number is then restored to 'u.uno'.
36588                                  <1>       ;    9) The child process name is put in 'u.r0'.
36589                                  <1>       ;   10) The pc on the stack sp + 18 is incremented by 2 to
36590                                  <1>       ;       create the return address for the parent process.
36591                                  <1>       ;   11) The 'u.fp' list as then searched to see what files
36592                                  <1>       ;       the parent has opened. For each file the parent has
36593                                  <1>       ;       opened, the corresponding 'fsp' entry must be updated
36594                                  <1>       ;       to indicate that the child process also has opened
36595                                  <1>       ;       the file. A branch to 'sysret' is then made.
36596                                  <1>       ;
36597                                  <1>       ; Calling sequence:
36598                                  <1>       ;       from shell ?
36599                                  <1>       ; Arguments:
36600                                  <1>       ;       -
36601                                  <1>       ; Inputs: -
36602                                  <1>       ; Outputs: *u.r0 - child process name
36603                                  <1>       ; ..........................................................
36604                                  <1>       ;
36605                                  <1>       ; Retro UNIX 8086 v1 modification:
36606                                  <1>       ;       AX = r0 = PID (>0) (at the return of 'sysfork')
36607                                  <1>       ;       = process id of child a parent process returns
36608                                  <1>       ;       = process id of parent when a child process returns
36609                                  <1>       ;
36610                                  <1>       ;         In original UNIX v1, sysfork is called and returns as
36611                                  <1>       ;       in following manner: (with an example: c library, fork)
36612                                  <1>       ;
36613                                  <1>       ;       1:
36614                                  <1>       ;               sys     fork
36615                                  <1>       ;                       br 1f  / child process returns here
36616                                  <1>       ;               bes     2f     / parent process returns here
36617                                  <1>       ;               / pid of new process in r0
36618                                  <1>       ;               rts     pc
36619                                  <1>       ;       2: / parent process condionally branches here
36620                                  <1>       ;               mov     $-1,r0 / pid = -1 means error return
36621                                  <1>       ;               rts     pc
36622                                  <1>       ;
36623                                  <1>       ;       1: / child process brances here
36624                                  <1>       ;               clr     r0   / pid = 0 in child process
36625                                  <1>       ;               rts     pc
36626                                  <1>       ;
36627                                  <1>       ;       In UNIX v7x86 (386) by Robert Nordier (1999)
36628                                  <1>       ;               // pid = fork();
36629                                  <1>       ;               //
36630                                  <1>       ;               // pid == 0 in child process;
36631                                  <1>       ;               // pid == -1 means error return
36632                                  <1>       ;               // in child,
36633                                  <1>       ;               //     parents id is in par_uid if needed
36634                                  <1>       ;
36635                                  <1>       ;               _fork:
36636                                  <1>       ;                       mov     $.fork,eax
36637                                  <1>       ;                       int     $0x30
36638                                  <1>       ;                       jmp     1f
36639                                  <1>       ;                       jnc     2f
36640                                  <1>       ;                       jmp     cerror
36641                                  <1>       ;               1:
36642                                  <1>       ;                       mov     eax,_par_uid
36643                                  <1>       ;                       xor     eax,eax
36644                                  <1>       ;               2:
36645                                  <1>       ;                       ret
36646                                  <1>       ;
36647                                  <1>       ;       In Retro UNIX 8086 v1,
36648                                  <1>       ;       'sysfork' returns in following manner:
36649                                  <1>       ;
36650                                  <1>       ;               mov     ax, sys_fork
36651                                  <1>       ;               mov     bx, offset @f ; routine for child
36652                                  <1>       ;               int     20h
36653                                  <1>       ;               jc      error
36654                                  <1>       ;
36655                                  <1>       ;       ; Routine for parent process here (just after 'jc')
36656                                  <1>       ;               mov     word ptr [pid_of_child], ax
36657                                  <1>       ;               jmp     next_routine_for_parent
36658                                  <1>       ;
36659                                  <1>       ;       @@: ; routine for child process here
36660                                  <1>       ;               ....
36661                                  <1>       ;       NOTE: 'sysfork' returns to specified offset
36662                                  <1>       ;              for child process by using BX input.
```

```
36663                                  <1>       ;           (at first, parent process will return then
36664                                  <1>       ;           child process will return -after swapped in-
36665                                  <1>       ;           'syswait' is needed in parent process
36666                                  <1>       ;           if return from child process will be waited for.)
36667                                  <1>       ;
36668                                  <1>
36669                                  <1> ; / create a new process
36670                                  <1>       ; EBX = return address for child process
36671                                  <1>           ; (Retro UNIX 8086 v1 modification !)
36672 0000C839 31F6                   <1>       xor   esi, esi
36673                                  <1>             ; clr r1
36674                                  <1> sysfork_1: ; 1: / search p.stat table for unused process number
36675 0000C83B 46                     <1>       inc   esi
36676                                  <1>             ; inc r1
36677 0000C83C 80BE[AF000300]00       <1>       cmp   byte [esi+p.stat-1], 0 ; SFREE, 05/02/2014
36678                                  <1>             ; tstb p.stat-1(r1) / is process active, unused, dead
36679 0000C843 760B                   <1>       jna   short sysfork_2
36680                                  <1>             ; beq 1f / it's unused so branch
36681 0000C845 6683FE10               <1>       cmp   si, nproc
36682                                  <1>             ; cmp r1,$nproc / all processes checked
36683 0000C849 72F0                   <1>       jb    short sysfork_1
36684                                  <1>             ; blt 1b / no, branch back
36685                                  <1>       ;
36686                                  <1>       ; Retro UNIX 8086 v1. modification:
36687                                  <1>       ;     Parent process returns from 'sysfork' to address
36688                                  <1>       ;     which is just after 'sysfork' system call in parent
36689                                  <1>       ;     process. Child process returns to address which is put
36690                                  <1>       ;     in BX register by parent process for 'sysfork'.
36691                                  <1>       ;
36692                                  <1>             ;add $2,18.(sp) / add 2 to pc when trap occured, points
36693                                  <1>                         ; / to old process return
36694                                  <1>             ; br error1 / no room for a new process
36695 0000C84B E96DFCFFFF             <1>       jmp   error
36696                                  <1> sysfork_2: ; 1:
36697 0000C850 E82583FFFF             <1>       call  allocate_page
36698 0000C855 0F8262FCFFFF           <1>       jc    error
36699 0000C85B 50                     <1>       push  eax   ; UPAGE (user structure page) address
36700                                  <1>       ; Retro UNIX 386 v1 modification!
36701 0000C85C E82885FFFF             <1>       call  duplicate_page_dir
36702                                  <1>             ; EAX = New page directory
36703 0000C861 730B                   <1>       jnc   short sysfork_3
36704 0000C863 58                     <1>       pop   eax   ; UPAGE (user structure page) address
36705 0000C864 E8EF84FFFF             <1>       call  deallocate_page
36706 0000C869 E94FFCFFFF             <1>       jmp   error
36707                                  <1> sysfork_3:
36708                                  <1>       ; Retro UNIX 386 v1 modification !
36709 0000C86E 56                     <1>       push  esi
36710 0000C86F E8DE1E0000             <1>       call  wswap ; save current user (u) structure, user registers
36711                                  <1>                   ; and interrupt return components (for IRET)
36712 0000C874 8705[B8030300]         <1>       xchg  eax, [u.pgdir] ; page directory of the child process
36713 0000C87A A3[BC030300]           <1>       mov   [u.ppgdir], eax ; page directory of the parent process
36714 0000C87F 5E                     <1>       pop   esi
36715 0000C880 58                     <1>       pop   eax   ; UPAGE (user structure page) address
36716                                  <1>             ; [u.usp] = esp
36717 0000C881 89F7                   <1>       mov   edi, esi
36718 0000C883 66C1E702               <1>       shl   di, 2
36719 0000C887 8987[BC000300]         <1>       mov   [edi+p.upage-4], eax ; memory page for 'user' struct
36720 0000C88D A3[B4030300]           <1>       mov   [u.upage], eax ; memory page for 'user' struct (child)
36721                                  <1>       ; 28/08/2015
36722 0000C892 0FB605[B3030300]       <1>       movzx eax, byte [u.uno] ; parent process number
36723                                  <1>             ; movb u.uno,-(sp) / save parent process number
36724 0000C899 89C7                   <1>       mov   edi, eax
36725 0000C89B 50                     <1>         push eax ; **
36726 0000C89C 8A87[7F000300]         <1>       mov    al, [edi+p.ttyc-1] ; console tty (parent)
36727                                  <1>       ; 18/09/2015
36728                                  <1>       ;mov    [esi+p.ttyc-1], al ; set child's console tty
36729                                  <1>       ;mov    [esi+p.waitc-1], ah ; 0 ; reset child's wait channel
36730 0000C8A2 668986[7F000300]       <1>       mov    [esi+p.ttyc-1], ax ; al - set child's console tty
36731                                  <1>                               ; ah - reset child's wait channel
36732 0000C8A9 89F0                   <1>       mov   eax, esi
36733 0000C8AB A2[B3030300]           <1>       mov   [u.uno], al ; child process number
36734                                  <1>             ;movb r1,u.uno / set child process number to r1
36735 0000C8B0 FE86[AF000300]         <1>         inc    byte [esi+p.stat-1] ; 1, SRUN, 05/02/2014
36736                                  <1>             ; incb p.stat-1(r1) / set p.stat entry for child
36737                                  <1>                          ; / process to active status
36738                                  <1>             ; mov u.ttyp,r2 / put pointer to parent process'
36739                                  <1>                         ; / control tty buffer in r2
36740                                  <1>                ; beq 2f / branch, if no such tty assigned
36741                                  <1>             ; clrb 6(r2) / clear interrupt character in tty buffer
36742                                  <1>       ; 2:
36743 0000C8B6 53                     <1>       push  ebx  ; * return address for the child process
36744                                  <1>                  ; * Retro UNIX 8086 v1 feature only !
36745                                  <1>       ; (Retro UNIX 8086 v1 modification!)
36746                                  <1>             ; mov $runq+4,r2
36747 0000C8B7 BB[54030300]           <1>       mov   ebx, runq+2 ; normal run queue ; 02/01/2017
36748 0000C8BC E8011F0000             <1>       call  putlu
36749                                  <1>             ; jsr r0,putlu / put child process on lowest priority
36750                                  <1>                        ; / run queue
36751 0000C8C1 66D1E6                 <1>       shl   si, 1
36752                                  <1>             ; asl r1 / multiply r1 by 2 to get index
36753                                  <1>                    ; / into p.pid table
36754 0000C8C4 66FF05[4E030300]       <1>       inc   word [mpid]
36755                                  <1>             ; inc mpid / increment m.pid; get a new process name
36756 0000C8CB 66A1[4E030300]         <1>       mov   ax, [mpid]
36757 0000C8D1 668986[1E000300]       <1>       mov   [esi+p.pid-2], ax
36758                                  <1>             ;mov mpid,p.pid-2(r1) / put new process name
36759                                  <1>                                  ; / in child process' name slot
36760 0000C8D8 5A                     <1>       pop   edx  ; * return address for the child process
36761                                  <1>                  ; * Retro UNIX 8086 v1 feature only !
36762 0000C8D9 5B                     <1>       pop   ebx  ; **
36763                                  <1>       ;mov  ebx, [esp] ; ** parent process number
36764                                  <1>             ; movb (sp),r2 / put parent process number in r2
36765 0000C8DA 66D1E3                 <1>       shl   bx, 1
```

```
36766                                  <1>         ;asl r2 / multiply by 2 to get index into below tables
36767                                  <1>       ;movzx eax, word [ebx+p.pid-2]
36768 0000C8DD 668B83[1E000300]        <1>       mov   ax, [ebx+p.pid-2]
36769                                  <1>             ; mov p.pid-2(r2),r2 / get process name of parent
36770                                  <1>                                  ; / process
36771 0000C8E4 668986[3E000300]        <1>       mov   [esi+p.ppid-2], ax
36772                                  <1>             ; mov r2,p.ppid-2(r1) / put parent process name
36773                                  <1>                                   ; / in parent process slot for child
36774 0000C8EB A3[64030300]            <1>       mov   [u.r0], eax
36775                                  <1>             ; mov r2,*u.r0 / put parent process name on stack
36776                                  <1>                                ; / at location where r0 was saved
36777 0000C8F0 8B2D[5C030300]          <1>       mov   ebp, [u.sp] ; points to return address (EIP for IRET)
36778 0000C8F6 895500                  <1>       mov   [ebp], edx ; *, CS:EIP -> EIP
36779                                  <1>                          ; * return address for the child process
36780                                  <1>             ; mov $sysret1,-(sp) /
36781                                  <1>             ; mov sp,u.usp / contents of sp at the time when
36782                                  <1>                              ; / user is swapped out
36783                                  <1>             ; mov $sstack,sp / point sp to swapping stack space
36784                                  <1>       ; 04/09/2015 - 01/09/2015
36785                                  <1>       ; [u.usp] = esp
36786 0000C8F9 68[DDC40000]            <1>       push  sysret ; ***
36787 0000C8FE 8925[60030300]          <1>       mov   [u.usp], esp ; points to 'sysret' address (***)
36788                                  <1>                          ; (for child process)
36789 0000C904 31C0                    <1>       xor   eax, eax
36790 0000C906 66A3[94030300]          <1>       mov   [u.ttyp], ax ; 0
36791                                  <1>       ;
36792 0000C90C E8411E0000              <1>       call  wswap ; Retro UNIX 8086 v1 modification !
36793                                  <1>             ;jsr r0,wswap / put child process out on drum
36794                                  <1>             ;jsr r0,unpack / unpack user stack
36795                                  <1>             ;mov u.usp,sp / restore user stack pointer
36796                                  <1>             ; tst (sp)+ / bump stack pointer
36797                                  <1>       ; Retro UNIX 386 v1 modification !
36798 0000C911 58                      <1>       pop   eax ; ***
36799 0000C912 66D1E3                  <1>       shl   bx, 1
36800 0000C915 8B83[BC000300]          <1>       mov    eax, [ebx+p.upage-4] ; UPAGE address ; 14/05/2015
36801 0000C91B E86A1E0000              <1>       call  rswap ; restore parent process 'u' structure,
36802                                  <1>                   ; registers and return address (for IRET)
36803                                  <1>             ;movb (sp)+,u.uno / put parent process number in u.uno
36804 0000C920 0FB705[4E030300]        <1>         movzx   eax, word [mpid]
36805 0000C927 A3[64030300]            <1>       mov   [u.r0], eax
36806                                  <1>             ; mov mpid,*u.r0 / put child process name on stack
36807                                  <1>                              ; / where r0 was saved
36808                                  <1>             ; add $2,18.(sp) / add 2 to pc on stack; gives parent
36809                                  <1>                                ; / process return
36810                                  <1>       ;xor   ebx, ebx
36811 0000C92C 31F6                    <1>       xor    esi, esi
36812                                  <1>             ;clr r1
36813                                  <1> sysfork_4: ; 1: / search u.fp list to find the files
36814                                  <1>           ; / opened by the parent process
36815                                  <1>       ; 01/09/2015
36816                                  <1>       ;xor   bh, bh
36817                                  <1>       ;mov   bl, [esi+u.fp]
36818 0000C92E 8A86[6A030300]          <1>       mov   al, [esi+u.fp]
36819                                  <1>             ; movb u.fp(r1),r2 / get an open file for this process
36820                                  <1>         ;or      bl, bl
36821 0000C934 08C0                    <1>       or    al, al
36822 0000C936 740D                    <1>       jz    short sysfork_5
36823                                  <1>             ; beq 2f / file has not been opened by parent,
36824                                  <1>                    ; / so branch
36825 0000C938 B40A                    <1>       mov   ah, 10 ; Retro UNIX 386 v1 fsp structure size = 10 bytes
36826 0000C93A F6E4                    <1>       mul   ah
36827                                  <1>       ;movzx ebx, ax
36828 0000C93C 6689C3                  <1>       mov   bx, ax
36829                                  <1>       ;shl   bx, 3
36830                                  <1>             ; asl r2 / multiply by 8
36831                                  <1>                   ; asl r2 / to get index into fsp table
36832                                  <1>                   ; asl r2
36833 0000C93F FE83[4E010300]          <1>       inc    byte [ebx+fsp-2]
36834                                  <1>             ; incb fsp-2(r2) / increment number of processes
36835                                  <1>                        ; / using file, because child will now be
36836                                  <1>                        ; / using this file
36837                                  <1> sysfork_5: ; 2:
36838 0000C945 46                      <1>         inc    esi
36839                                  <1>             ; inc r1 / get next open file
36840 0000C946 6683FE0A                <1>         cmp    si, 10
36841                                  <1>             ; cmp r1,$10. / 10. files is the maximum number which
36842                                  <1>                     ; / can be opened
36843 0000C94A 72E2                    <1>       jb    short sysfork_4
36844                                  <1>             ; blt 1b / check next entry
36845 0000C94C E98CFBFFFF              <1>       jmp   sysret
36846                                  <1>             ; br sysret1
36847                                  <1>
36848                                  <1> syscreat: ; < create file >
36849                                  <1>       ; 13/11/2017
36850                                  <1>       ; 27/10/2016
36851                                  <1>       ; 25/10/2016, 26/10/2016
36852                                  <1>       ; 15/10/2016, 16/10/2016, 17/10/2016
36853                                  <1>       ; 10/10/2016 (TRDOS 386 = TRDOS v2.0)
36854                                  <1>       ;          -derived from INT_21H.ASM-
36855                                  <1>       ;           ("loc_INT21h_create_file")
36856                                  <1>         ;   10/07/2011 (12/03/2011)
36857                                  <1>         ;   INT 21h Function AH = 3Ch
36858                                  <1>         ;   Create File
36859                                  <1>         ;   INPUT
36860                                  <1>         ;      CX = Attributes
36861                                  <1>         ;         DS:DX= Address of zero terminaned path name
36862                                  <1>         ;
36863                                  <1>       ; 27/12/2015 (Retro UNIX 386 v1.1)
36864                                  <1>       ; 14/05/2015 (Retro UNIX 386 v1 - Beginning)
36865                                  <1>       ; 27/05/2013 (Retro UNIX 8086 v1)
36866                                  <1>       ;
36867                                  <1>       ; 'syscreat' called with two arguments; name and mode.
36868                                  <1>       ; u.namep points to name of the file and mode is put
```

```
36869                                  <1>       ; on the stack. 'namei' is called to get i-number of the file.
36870                                  <1>       ; If the file aready exists, it's mode and owner remain
36871                                  <1>       ; unchanged, but it is truncated to zero length. If the file
36872                                  <1>       ; did not exist, an i-node is created with the new mode via
36873                                  <1>       ; 'maknod' whether or not the file already existed, it is
36874                                  <1>       ; open for writing. The fsp table is then searched for a free
36875                                  <1>       ; entry. When a free entry is found, proper data is placed
36876                                  <1>       ; in it and the number of this entry is put in the u.fp list.
36877                                  <1>       ; The index to the u.fp (also know as the file descriptor)
36878                                  <1>       ; is put in the user's r0.
36879                                  <1>       ;
36880                                  <1>       ; Calling sequence:
36881                                  <1>       ;     syscreate; name; mode
36882                                  <1>       ; Arguments:
36883                                  <1>       ;     name - name of the file to be created
36884                                  <1>       ;     mode - mode of the file to be created
36885                                  <1>       ; Inputs: (arguments)
36886                                  <1>       ; Outputs: *u.r0 - index to u.fp list
36887                                  <1>       ;              (the file descriptor of new file)
36888                                  <1>       ; ..........................................................
36889                                  <1>       ;
36890                                  <1>       ; Retro UNIX 8086 v1 modification:
36891                                  <1>       ;      'syscreate' system call has two arguments; so,
36892                                  <1>       ;     * 1st argument, name is pointed to by BX register
36893                                  <1>       ;     * 2nd argument, mode is in CX register
36894                                  <1>       ;
36895                                  <1>       ;     AX register (will be restored via 'u.r0') will return
36896                                  <1>       ;     to the user with the file descriptor/number
36897                                  <1>       ;     (index to u.fp list).
36898                                  <1>       ;
36899                                  <1>       ;call  arg2
36900                                  <1>       ; * name - 'u.namep' points to address of file/path name
36901                                  <1>       ;          in the user's program segment ('u.segmnt')
36902                                  <1>       ;          with offset in BX register (as sysopen argument 1).
36903                                  <1>       ; * mode - sysopen argument 2 is in CX register
36904                                  <1>       ;          which is on top of stack.
36905                                  <1>       ;
36906                                  <1>       ; TRDOS 386 (10/10/2016)
36907                                  <1>       ;
36908                                  <1>         ; INPUT ->
36909                                  <1>         ;      CL = File Attributes
36910                                  <1>       ;           bit 0 (1) - Read only file (R)
36911                                  <1>       ;            bit 1 (1) - Hidden file (H)
36912                                  <1>         ;            bit 2 (1) - System file (R)
36913                                  <1>       ;            bit 3 (1) - Volume label/name (V)
36914                                  <1>         ;            bit 4 (1) - Subdirectory (D)
36915                                  <1>       ;           bit 5 (1) - File has been archived (A)
36916                                  <1>         ;          EBX = Pointer to filename (ASCIIZ) -path-
36917                                  <1>       ;
36918                                  <1>       ; OUTPUT ->
36919                                  <1>       ;          eax = File/Device Handle/Number (index) (AL)
36920                                  <1>       ;          cf = 1 -> Error code in AL
36921                                  <1>       ;
36922                                  <1>       ; Modified Registers: EAX (at the return of system call)
36923                                  <1>       ;
36924                                  <1>       ; Note: If the file is existing and it has not any one
36925                                  <1>       ;     of S,H,R,V,D attributes, it will be truncated
36926                                  <1>       ;     to zero length; otherwise, access error will be
36927                                  <1>       ;     returned.
36928                                  <1>
36929                                  <1> sysmkdir_0:
36930 0000C951 F6C108                <1>       test   cl, 08h ; Volume name
36931 0000C954 740A                  <1>       jz     short syscreat_0
36932                                  <1>
36933                                  <1>       ; Volume name or long name creation
36934                                  <1>       ; is not permitted (in TRDOS 386)!
36935 0000C956 B80B000000            <1>       mov    eax, ERR_FILE_ACCESS  ; 11 ; 'permission denied !'
36936 0000C95B E926020000            <1>         jmp  sysopen_dev_err
36937                                  <1>
36938                                  <1> syscreat_0:
36939                                  <1>         ;mov [u.namep], ebx
36940 0000C960 51                    <1>       push   ecx
36941 0000C961 89DE                  <1>       mov    esi, ebx
36942                                  <1>       ; file name is forced, change directory as temporary
36943                                  <1>       ;mov   ax, 1
36944                                  <1>       ;mov   [FFF_Valid], ah ; 0 ; reset ; 17/10/2016
36945                                  <1>       ;call  set_working_path
36946 0000C963 E8932C0000            <1>       call   set_working_path_x ; 17/10/2016
36947 0000C968 0F82D7000000          <1>       jc     syscreat_err
36948                                  <1>
36949                                  <1>       ; 16/10/2016
36950 0000C96E 803D[DB5F0100]00      <1>       cmp    byte [SWP_inv_fname], 0
36951 0000C975 776C                  <1>       ja     short  syscreat_inv_fname ; invalid file name !
36952                                  <1>
36953                                  <1>       ; Here, we have a valid path and also a valid file name
36954                                  <1>       ; (Working dir has been changed if the path
36955                                  <1>       ;  -file name string- had contained a dir name.)
36956                                  <1>
36957 0000C977 6631C0                <1>       xor    ax, ax
36958                                  <1>       ;mov   esi, FindFile_Name
36959 0000C97A E88CB6FFFF            <1>       call   find_first_file
36960 0000C97F 59                    <1>       pop    ecx
36961                                  <1>              ; ESI = Directory Entry (FindFile_DirEntry) Location
36962                                  <1>              ; EDI = Directory Buffer Directory Entry Location
36963                                  <1>              ; EAX = File Size
36964                                  <1>              ;  BL = Attributes of The File/Directory
36965                                  <1>              ;  BH = Long Name Yes/No Status (>0 is YES)
36966                                  <1>              ;  DX > 0 : Ambiguous filename chars are used
36967 0000C980 7269                  <1>       jc     short syscreat_1 ; file not found (the good!)
36968                                  <1>                             ; or another error (the bad')
36969                                  <1>
36970                                  <1>       ; (& the uggly!) truncate file to zero length before open
36971                                  <1>
```

```
                                <1>       ;'*' and '?' already checked at 'set_working_path' stage
                                <1>       ;and  dx, dx
                                <1>       ;jnz  short sysmkdir_err ; permission denied
                                <1>                                ; invalid filename chars
                                <1>
                                <1>       ;test  cl, 10h ; subdirectory ?
                                <1>       ;jnz  short sysmkdir_err
                                <1>
                                <1>       ; BL = File Attributes:
                                <1>       ;           bit 0 (1) - Read only file (R)
                                <1>       ;             bit 1 (1) - Hidden file (H)
                                <1>         ;             bit 2 (1) - System file (R)
                                <1>       ;             bit 3 (1) - Volume label/name (V)
                                <1>         ;             bit 4 (1) - Subdirectory (D)
                                <1>       ;           bit 5 (1) - File has been archived
                                <1>
                                <1>       ; * existing directory must not be truncated
                                <1>       ;   (we don't know it is empty or not, at this stage)
                                <1>       ; * existing volume name (or a long name) can not be
                                <1>       ;   re-created or truncated by 'syscreat'
                                <1>       ; * A file with S, H, R attributes must not be truncated
                                <1>       ;   (change attributes to normal, if you need truncate it)
                                <1>
0000C982 F6C31F                 <1>       test   bl, 00011111b  ; check attributes of existing file
0000C985 754E                   <1>       jnz    short sysmkdir_err
                                <1>
                                <1>       ;; normal file, OK to continue...
                                <1>
                                <1>       ; ESI = FindFile_DirEntry
0000C987 668B4614               <1>       mov    ax, [esi+DirEntry_FstClusHI] ; 20
0000C98B C1E010                 <1>       shl    eax, 16 ; 13/11/2017
0000C98E 668B461A               <1>       mov    ax, [esi+DirEntry_FstClusLO] ; 26
                                <1>       ; EAX = First cluster to be truncated/unlinked
0000C992 57                     <1>       push   edi
0000C993 51                     <1>       push   ecx
0000C994 BE00010900             <1>       mov    esi, Logical_DOSDisks
0000C999 29C9                   <1>       sub    ecx, ecx
0000C99B 8A2D[E6520100]         <1>       mov    ch, [Current_Drv]
0000C9A1 01CE                   <1>       add    esi, ecx
                                <1>       ; ESI = Logical dos drive description table address
0000C9A3 E8C9F7FFFF             <1>       call   truncate_cluster_chain
0000C9A8 59                     <1>       pop    ecx
0000C9A9 5F                     <1>       pop    edi
0000C9AA 7230                   <1>       jc     short syscreate_truncate_err
                                <1>
                                <1>       ; 26/10/2016
                                <1>       ; EDI = Directory entry address in directory buffer
                                <1>       ; Update directory entry
0000C9AC E846DCFFFF             <1>       call   convert_current_date_time
                                <1>       ; OUTPUT -> DX = Date in dos dir entry format
                                <1>         ;       AX = Time in dos dir entry format
0000C9B1 66894716               <1>       mov    [edi+DirEntry_WrtTime], ax
0000C9B5 66895718               <1>       mov    [edi+DirEntry_WrtDate], dx
0000C9B9 66895712               <1>       mov    [edi+DirEntry_LastAccDate], dx
0000C9BD 31C0                   <1>       xor    eax, eax ; file size = 0
0000C9BF 89471C                 <1>       mov    [edi+DirEntry_FileSize], eax ; 0
0000C9C2 C605[105B0100]02       <1>       mov    byte [DirBuff_ValidData], 2 ; data changed sign
0000C9C9 BE[DC5C0100]           <1>       mov    esi, FindFile_DirEntry
0000C9CE B201                   <1>       mov    dl, 1 ; open file for writing
0000C9D0 E9AA000000             <1>       jmp    sysopen_2
                                <1>
                                <1> sysmkdir_err:
                                <1>       ; 1 = write, 2 = read & write, >2 = invalid
0000C9D5 B80B000000             <1>         mov  eax, ERR_FILE_ACCESS  ; 11 ; 'permission denied !'
0000C9DA EB73                   <1>         jmp  short sysopen_err
                                <1>
                                <1> syscreate_truncate_err:
0000C9DC B812000000             <1>       mov    eax, ERR_DRV_WRITE ; 18 ; 'disk write error !'
0000C9E1 EB6C                   <1>         jmp  short sysopen_err
                                <1>
                                <1> syscreat_inv_fname:  ; invalid file name chars
                                <1>       ; 16/10/2016
0000C9E3 B81A000000             <1>       mov    eax, ERR_INV_FILE_NAME  ; 26 ; invalid file name chars
0000C9E8 59                     <1>       pop    ecx
0000C9E9 EB64                   <1>       jmp    sysopen_err
                                <1>
                                <1> syscreat_1:
                                <1>       ; Error code in EAX
0000C9EB 3C02                   <1>         cmp  al, 02h ; 'File not found' error
0000C9ED 7560                   <1>         jne  sysopen_err
                                <1>
0000C9EF F6C110                 <1>       test   cl, 10h ; Directory
0000C9F2 0F852C020000           <1>       jnz    sysmkdir_2
                                <1>
                                <1> syscreat_2:
0000C9F8 BE[CC5C0100]           <1>       mov    esi, FindFile_Name
                                <1>         ;xor edx, edx
0000C9FD 31C0                   <1>         xor  eax, eax ; File Size  = 0
0000C9FF 31DB                   <1>       xor    ebx, ebx
0000CA01 4B                     <1>       dec    ebx ; FFFFFFFFh -> create empty file
                                <1>                  ;             (only for FAT fs)
                                <1>       ; CL = File Attributes
0000CA02 E8F8EBFFFF             <1>       call   create_file
0000CA07 7246                   <1>       jc     sysopen_err
                                <1>              ; EAX = New file's first cluster
                                <1>              ; ESI = Logical Dos Drv Descr. Table Addr.
                                <1>              ; EBX = offset CreateFile_Size
                                <1>              ; ECX = Sectors per cluster (<256)
                                <1>              ; EDX = Directory entry index/number (<65536)
                                <1>       ; 26/10/2016
                                <1>       ;mov   esi, Directory_Buffer
                                <1>       ;shl   dx, 5 ; *32
                                <1>       ;add   esi, edx
```

```
37075                                  <1>       ;; esi = directory entry address in directory buffer
37076                                  <1>
37077                                  <1>       ; Here, directory entry has been created but last
37078                                  <1>       ; modification date & time of the parent dir has not
37079                                  <1>       ; been updated, yet!
37080                                  <1>       ; (Note: Directory and FAT buffers have been updated...)
37081                                  <1>
37082 0000CA09 E822DDFFFF             <1>       call   update_parent_dir_lmdt ; now, it is OK too!
37083                                  <1>
37084                                  <1>       ; 25/10/2016
37085 0000CA0E 66B80018                <1>       mov    ax, 1800h
37086 0000CA12 BE[CC5C0100]            <1>       mov    esi, FindFile_Name
37087 0000CA17 E8EFB5FFFF              <1>       call   find_first_file
37088 0000CA1C 7231                    <1>       jc     short sysopen_err
37089                                  <1>
37090                                  <1>       ; Only possible error after here is
37091                                  <1>       ; "too many open files !" error.
37092                                  <1>       ;
37093                                  <1>       ; If "syscreat" will return with that error,
37094                                  <1>       ; (the file has been created but it could not be opened)
37095                                  <1>       ; the user must retry to open this file again
37096                                  <1>       ; or must close another file before using
37097                                  <1>       ; "sysopen" system call.
37098                                  <1>
37099 0000CA1E B201                    <1>       mov    dl, 1 ; open file for writing
37100                                  <1>       ; ESI = Directory Entry (FindFile_DirEntry) Location
37101                                  <1>       ; EAX = File Size (= 0)
37102 0000CA20 EB5D                    <1>       jmp    short sysopen_2
37103                                  <1>
37104                                  <1> sysopen: ;<open file>
37105                                  <1>       ; 26/10/2016
37106                                  <1>       ; 24/10/2016
37107                                  <1>       ; 17/10/2016
37108                                  <1>       ; 15/10/2016
37109                                  <1>       ; 06/10/2016, 07/10/2016, 08/10/2016
37110                                  <1>       ; 05/10/2016 (TRDOS 386 = TRDOS v2.0)
37111                                  <1>       ;          -derived from INT_21H.ASM-
37112                                  <1>       ;           ("loc_INT21h_open_file")
37113                                  <1>         ;    26/02/2011
37114                                  <1>         ;    INT 21h Function AH = 3Dh
37115                                  <1>         ;    Open File
37116                                  <1>         ;    INPUT
37117                                  <1>         ;       AL= File Access Value
37118                                  <1>       ;           0- Open for reading
37119                                  <1>       ;            1- Open for writing
37120                                  <1>         ;           2- Open for reading and writing
37121                                  <1>         ;          DS:DX= Pointer to filename (ASCIIZ)
37122                                  <1>         ;
37123                                  <1>       ; 14/05/2015 (Retro UNIX 386 v1 - Beginning)
37124                                  <1>       ; 22/05/2013 - 27/05/2013 (Retro UNIX 8086 v1)
37125                                  <1>       ;
37126                                  <1>       ; 'sysopen' opens a file in following manner:
37127                                  <1>       ;    1) The second argument in a sysopen says whether to
37128                                  <1>       ;     open the file ro read (0) or write (>0).
37129                                  <1>       ;    2) I-node of the particular file is obtained via 'namei'.
37130                                  <1>       ;    3) The file is opened by 'iopen'.
37131                                  <1>       ;    4) Next housekeeping is performed on the fsp table
37132                                  <1>       ;     and the user's open file list - u.fp.
37133                                  <1>       ;     a) u.fp and fsp are scanned for the next available slot.
37134                                  <1>       ;     b) An entry for the file is created in the fsp table.
37135                                  <1>       ;     c) The number of this entry is put on u.fp list.
37136                                  <1>       ;     d) The file descriptor index to u.fp list is pointed
37137                                  <1>       ;        to by u.r0.
37138                                  <1>       ;
37139                                  <1>       ; Calling sequence:
37140                                  <1>       ;     sysopen; name; mode
37141                                  <1>       ; Arguments:
37142                                  <1>       ;     name - file name or path name
37143                                  <1>       ;     mode - 0 to open for reading
37144                                  <1>       ;            1 to open for writing
37145                                  <1>       ; Inputs: (arguments)
37146                                  <1>       ; Outputs: *u.r0 - index to u.fp list (the file descriptor)
37147                                  <1>       ;             is put into r0's location on the stack.
37148                                  <1>       ; ..........................................................
37149                                  <1>       ;
37150                                  <1>       ; Retro UNIX 8086 v1 modification:
37151                                  <1>       ;      'sysopen' system call has two arguments; so,
37152                                  <1>       ;     * 1st argument, name is pointed to by BX register
37153                                  <1>       ;     * 2nd argument, mode is in CX register
37154                                  <1>       ;
37155                                  <1>       ;     AX register (will be restored via 'u.r0') will return
37156                                  <1>       ;     to the user with the file descriptor/number
37157                                  <1>       ;     (index to u.fp list).
37158                                  <1>       ;
37159                                  <1>       ;call  arg2
37160                                  <1>       ; * name - 'u.namep' points to address of file/path name
37161                                  <1>       ;          in the user's program segment ('u.segmnt')
37162                                  <1>       ;          with offset in BX register (as sysopen argument 1).
37163                                  <1>       ; * mode - sysopen argument 2 is in CX register
37164                                  <1>       ;          which is on top of stack.
37165                                  <1>       ;
37166                                  <1>       ; jsr r0,arg2 / get sys args into u.namep and on stack
37167                                  <1>       ;
37168                                  <1>            ; system call registers: ebx, ecx (through 'sysenter')
37169                                  <1>       ;
37170                                  <1>       ; TRDOS 386 (05/10/2016)
37171                                  <1>       ;
37172                                  <1>         ; INPUT ->
37173                                  <1>         ;      CL = File Access Value (Open Mode)
37174                                  <1>       ;           0 - Open file for reading
37175                                  <1>       ;            1 - Open file for writing
37176                                  <1>         ;           2 - Open device for reading
37177                                  <1>       ;           3 - Open device for writing
```

```
37178                                  <1>         ;          EBX = Pointer to filename/devicename (ASCIIZ)
37179                                  <1>       ; OUTPUT ->
37180                                  <1>       ;          eax = File/Device Handle/Number (index) (AL)
37181                                  <1>       ;          cf = 1 -> Error code in AL
37182                                  <1>       ;
37183                                  <1>       ; Modified Registers: EAX (at the return of system call)
37184                                  <1>       ;
37185                                  <1>
37186 0000CA22 80F901                  <1>       cmp    cl, 1 ; read file (0), write file (1)
37187 0000CA25 7614                    <1>       jna    short sysopen_0
37188                                  <1>
37189 0000CA27 80F903                  <1>       cmp    cl, 3
37190 0000CA2A 0F8640010000            <1>       jna    sysopen_device
37191                                  <1>
37192                                  <1>       ; Invalid access code
37193 0000CA30 B817000000              <1>       mov    eax, ERR_INV_PARAMETER
37194 0000CA35 0F874B010000            <1>       ja     sysopen_dev_err
37195                                  <1>
37196                                  <1> sysopen_0:
37197                                  <1>       ;mov   [u.namep], ebx
37198 0000CA3B 51                      <1>       push   ecx
37199 0000CA3C 89DE                    <1>       mov    esi, ebx
37200                                  <1>       ; file name is forced, change directory as temporary
37201                                  <1>       ;mov   ax, 1
37202                                  <1>       ;mov   [FFF_Valid], ah ; 0 ; reset ; 17/10/2016
37203                                  <1>       ;call  set_working_path
37204 0000CA3E E8B82B0000              <1>       call   set_working_path_x ; 17/10/2016
37205 0000CA43 731E                    <1>       jnc    short sysopen_1
37206                                  <1>
37207                                  <1> syscreat_err: ; ecx = file attributes (for 'syscreat')
37208 0000CA45 59                      <1>       pop    ecx ; open mode
37209 0000CA46 21C0                    <1>       and    eax, eax  ; 0 -> Bad Path!
37210 0000CA48 7505                    <1>       jnz    short sysopen_err
37211                                  <1>       ; eax = 0
37212 0000CA4A B80C000000              <1>       mov    eax, ERR_DIR_NOT_FOUND ; Directory not found !
37213                                  <1> sysopen_err:
37214 0000CA4F A3[64030300]            <1>       mov    [u.r0], eax
37215 0000CA54 A3[C8030300]            <1>       mov    [u.error], eax
37216 0000CA59 E8722C0000              <1>       call   reset_working_path
37217 0000CA5E E95AFAFFFF              <1>       jmp    error
37218                                  <1>
37219                                  <1> sysopen_1:
37220                                  <1>       ;mov   esi, FindFile_Name
37221 0000CA63 66B80018                <1>         mov  ax, 1800h ; Only files
37222 0000CA67 E89FB5FFFF              <1>       call   find_first_file
37223 0000CA6C 5A                      <1>       pop    edx
37224 0000CA6D 72E0                    <1>       jc     short sysopen_err ; eax = 2 (File not found !)
37225                                  <1>
37226                                  <1>       ; check_open_file_attr_access_code
37227                                  <1>
37228 0000CA6F F6C307                  <1>         test bl, 7  ; system, hidden, readonly
37229 0000CA72 740B                    <1>         jz   short sysopen_2
37230                                  <1>
37231 0000CA74 20D2                    <1>       and    dl, dl ; 0 = read mode
37232 0000CA76 7407                    <1>       jz     short sysopen_2
37233                                  <1>
37234                                  <1>       ; 1 = write, 2 = read & write, >2 = invalid
37235 0000CA78 B80B000000              <1>         mov  eax, ERR_FILE_ACCESS  ; 11  = 'permission denied !'
37236 0000CA7D EBD0                    <1>         jmp  short sysopen_err
37237                                  <1>
37238                                  <1> sysopen_2:
37239                                  <1>       ; esi = Directory Entry (FindFile_DirEntry) Location
37240 0000CA7F 89F3                    <1>       mov    ebx, esi
37241 0000CA81 31F6                    <1>         xor     esi, esi ; 0
37242 0000CA83 31FF                    <1>         xor     edi, edi ; 0
37243                                  <1> sysopen_3: ; scan the list of entries in fsp table
37244 0000CA85 80BE[6A030300]00        <1>         cmp     byte [esi+u.fp], 0
37245 0000CA8C 760F                    <1>         jna     short sysopen_4 ; empty slot
37246 0000CA8E 6646                    <1>         inc     si
37247 0000CA90 6683FE0A                <1>         cmp     si, 10
37248 0000CA94 72EF                    <1>       jb     short sysopen_3
37249                                  <1> toomanyf:
37250 0000CA96 B80D000000              <1>       mov    eax, ERR_TOO_MANY_FILES ; too many open files !
37251 0000CA9B EBB2                    <1>       jmp    short sysopen_err
37252                                  <1>
37253                                  <1> sysopen_4:
37254 0000CA9D 80BF[4A630100]00        <1>         cmp     byte [edi+OF_MODE], 0 ; Scan open files table
37255 0000CAA4 760A                    <1>       jna     short sysopen_5
37256 0000CAA6 6647                    <1>       inc    di
37257 0000CAA8 6683FF0A                <1>       cmp     di, OPENFILES ; max. number of open files (=10)
37258 0000CAAC 72EF                    <1>       jb     short sysopen_4
37259 0000CAAE EBE6                    <1>       jmp    short toomanyf
37260                                  <1>
37261                                  <1> sysopen_5:
37262 0000CAB0 FEC2                    <1>       inc    dl
37263 0000CAB2 8897[4A630100]          <1>         mov     [edi+OF_MODE], dl
37264 0000CAB8 8A15[8A5C0100]          <1>       mov    dl, [FindFile_Drv]
37265 0000CABE 8897[40630100]          <1>         mov     [edi+OF_DRIVE], dl ; Logical DOS drive number
37266 0000CAC4 66C1E702                <1>       shl    di, 2 ; *4 (dword offset)
37267                                  <1>
37268 0000CAC8 8987[90630100]          <1>       mov    [edi+OF_SIZE], eax ; File size in bytes
37269                                  <1>
37270 0000CACE 668B4314                <1>         mov  ax, [ebx+DirEntry_FstClusHI]
37271 0000CAD2 C1E010                  <1>       shl    eax, 16
37272 0000CAD5 668B431A                <1>       mov    ax, [ebx+DirEntry_FstClusLO]
37273 0000CAD9 8987[18630100]          <1>       mov     [edi+OF_FCLUSTER], eax ; First cluster
37274 0000CADF 8987[30640100]          <1>       mov     [edi+OF_CCLUSTER], eax ; Current cluster
37275                                  <1>
37276 0000CAE5 31DB                    <1>         xor  ebx, ebx
37277 0000CAE7 899F[68630100]          <1>         mov     [edi+OF_POINTER], ebx ; offset pointer (0)
37278 0000CAED 899F[58640100]          <1>         mov     [edi+OF_CCINDEX], ebx ; cluster index (0)
37279                                  <1>
37280 0000CAF3 A1[FC5C0100]            <1>       mov    eax, [FindFile_DirFirstCluster]
```

```
37281 0000CAF8 8987[B8630100]      <1>       mov   [edi+OF_DIRFCLUSTER], eax
37282                              <1>
37283 0000CAFE A1[005D0100]        <1>       mov   eax, [FindFile_DirCluster]
37284 0000CB03 8987[E0630100]      <1>       mov   [edi+OF_DIRCLUSTER], eax
37285                              <1>
37286                              <1>       ; Get (& Save) Volume ID
37287                              <1>       ; Important for files of removable drives
37288                              <1>       ; (In order to check the drive has same volume/disk)
37289 0000CB09 88D7               <1>       mov   bh, dl
37290 0000CB0B 81C300010900       <1>         add  ebx, Logical_DOSDisks
37291 0000CB11 8A4303             <1>         mov  al, [ebx+LD_FATType]
37292 0000CB14 3C01               <1>         cmp  al, 1
37293 0000CB16 7209               <1>         jb   short sysopen_6_fs
37294 0000CB18 3C02               <1>         cmp  al, 2
37295 0000CB1A 770A               <1>         ja   short sysopen_6_fat32
37296                             <1> sysopen_6_fat:
37297 0000CB1C 8B432D             <1>         mov  eax, [ebx+LD_BPB+VolumeID]
37298 0000CB1F EB08               <1>         jmp  short sysopen_7
37299                             <1> sysopen_6_fs:
37300 0000CB21 8B4328             <1>         mov  eax, [ebx+LD_FS_VolumeSerial]
37301 0000CB24 EB03               <1>         jmp  short sysopen_7
37302                             <1> sysopen_6_fat32:
37303 0000CB26 8B4349             <1>         mov  eax, [ebx+LD_BPB+FAT32_VolID]
37304                             <1> sysopen_7:
37305 0000CB29 A3[DC520100]       <1>         mov  [Current_VolSerial], eax
37306                             <1>
37307 0000CB2E 8987[08640100]     <1>       mov   [edi+OF_VOLUMEID], eax
37308                             <1>
37309                             <1>       ; 24/10/2016
37310 0000CB34 66D1EF             <1>       shr   di, 1 ; 4/2, word offset
37311 0000CB37 668B1D[045D0100]   <1>       mov   bx, [FindFile_DirEntryNumber]
37312 0000CB3E 66899F[80640100]   <1>       mov   [edi+OF_DIRENTRY], bx
37313                             <1>
37314 0000CB45 31D2               <1>       xor   edx, edx
37315                             <1>       ;shr  di, 2 ; /4 (byte offset)
37316 0000CB47 66D1EF             <1>       shr   di, 1 ; 2/2, byte offset
37317 0000CB4A 8897[5E630100]     <1>       mov   byte [edi+OF_OPENCOUNT], dl ; 0
37318 0000CB50 8897[54630100]     <1>       mov   byte [edi+OF_STATUS], dl ; 0
37319                             <1>
37320 0000CB56 89FB               <1>       mov   ebx, edi
37321 0000CB58 FEC3               <1>       inc   bl
37322                             <1>
37323 0000CB5A 889E[6A030300]     <1>         mov    [esi+u.fp], bl ; Open File Entry Number
37324 0000CB60 8935[64030300]     <1>       mov    [u.r0], esi ; move index to u.fp list
37325                             <1>                    ; into eax on stack
37326                             <1>
37327 0000CB66 E8652B0000         <1>         call      reset_working_path
37328                             <1>
37329 0000CB6B E96DF9FFFF         <1>       jmp   sysret
37330                             <1>
37331                             <1>       ; (Retro UNIX 386 v1.0)
37332                             <1>       ; 'fsp' table (10 bytes/entry)
37333                             <1>       ; bit 15                              bit 0
37334                             <1>       ; ---|----------------------------------------
37335                             <1>       ; r/w|       i-number of open file
37336                             <1>       ; ---|----------------------------------------
37337                             <1>       ;               device number
37338                             <1>       ; ----------------------------------------------
37339                             <1>       ; offset pointer, r/w pointer to file (bit 0-15)
37340                             <1>       ; ----------------------------------------------
37341                             <1>       ; offset pointer, r/w pointer to file (bit 16-31)
37342                             <1>       ; ---------------------|------------------------
37343                             <1>       ;  flag that says file    | number of processes
37344                             <1>       ;   has been deleted      | that have file open
37345                             <1>       ; ---------------------|------------------------
37346                             <1>
37347                             <1> sysopen_device:
37348                             <1>       ; 15/10/2016
37349                             <1>       ; 08/10/2016
37350                             <1>       ; 07/10/2016 (TRDOS 386 = TRDOS v2.0)
37351 0000CB70 51                 <1>       push  ecx ; open mode
37352 0000CB71 89E5               <1>       mov   ebp, esp
37353 0000CB73 B910000000         <1>       mov   ecx, 16 ; transfer length = 16 bytes
37354 0000CB78 29CC               <1>       sub   esp, ecx
37355 0000CB7A 89E7               <1>       mov   edi, esp ; destination address
37356 0000CB7C 89DE               <1>       mov   esi, ebx ; dev name in user's memory space
37357 0000CB7E E8741D0000         <1>       call  transfer_from_user_buffer
37358 0000CB83 7310               <1>       jnc   short sysopen_dev_0
37359                             <1>       ; eax = ERR_OUT_OF_MEMORY = 4 = ERR_MINOR_IM
37360 0000CB85 59                 <1>       pop   ecx
37361                             <1> sysopen_dev_err:
37362 0000CB86 A3[64030300]       <1>       mov   [u.r0], eax
37363 0000CB8B A3[C8030300]       <1>       mov   [u.error], eax
37364 0000CB90 E928F9FFFF         <1>       jmp   error
37365                             <1> sysopen_dev_0:
37366 0000CB95 89FE               <1>       mov   esi, edi ; Device name addr (max. 16 bytes, ASCIIZ)
37367                             <1>                      ; for example: "tty, TTY, /dev/tty"
37368 0000CB97 E8DC2D0000         <1>       call  get_device_number
37369 0000CB9C 89EC               <1>       mov   esp, ebp
37370 0000CB9E 59                 <1>       pop   ecx
37371 0000CB9F 7307               <1>       jnc   short sysopen_dev_1
37372 0000CBA1 B818000000         <1>       mov   eax, ERR_INV_DEV_NAME ; 24 ; 'invalid device name !'
37373 0000CBA6 EBDE               <1>       jmp   short sysopen_dev_err
37374                             <1> sysopen_dev_1:
37375                             <1>       ; eax = Device Number (AL)
37376                             <1>       ;  cl = Open mode (2 = device read, 3 = device write)
37377 0000CBA8 31DB               <1>         xor    ebx, ebx ; 0
37378                             <1> sysopen_dev_2: ; scan the list of entries
37379 0000CBAA 389B[6A030300]     <1>         cmp    [ebx+u.fp], bl ; 0
37380 0000CBB0 760E               <1>         jna    short sysopen_dev_3 ; empty slot
37381 0000CBB2 FEC3               <1>         inc    bl
37382 0000CBB4 80FB0A             <1>         cmp    bl, 10
37383 0000CBB7 72F1               <1>       jb    short sysopen_dev_2
```

```
37384                                  <1>       ;
37385 0000CBB9 B80D000000              <1>       mov    eax, ERR_TOO_MANY_FILES ; too many open files !
37386 0000CBBE EBC6                    <1>       jmp    short sysopen_dev_err
37387                                  <1> sysopen_dev_3:
37388 0000CBC0 891D[64030300]          <1>       mov    [u.r0], ebx ; File/Device index/handle/descriptor
37389                                  <1>       ; eax = device number (entry offset)
37390 0000CBC6 8AA8[DC600100]          <1>       mov    ch, [eax+DEV_ACCESS] ; bit 0 = accessable by users
37391                                  <1>                                   ; bit 1 = read access perm
37392                                  <1>                                   ; bit 2 = write access perm
37393                                  <1>                                   ; bit 3 = IOCTL permit to users
37394                                  <1>                                   ; bit 4 = block device if set
37395                                  <1>                                   ; bit 5 = 16 bit or 1024 byte
37396                                  <1>                                   ; bit 6 = 32 bit or 2048 byte
37397                                  <1>                                   ; bit 7 = installable device drv
37398 0000CBCC F6C501                  <1>       test   ch, 1 ; accessable by normal users (except root)
37399 0000CBCF 7510                    <1>       jnz    short sysopen_dev_4 ; yes, permission has been given
37400 0000CBD1 803D[B0030300]00        <1>       cmp    byte [u.uid], 0 ; root?
37401 0000CBD8 7607                    <1>       jna    short sysopen_dev_4 ; superuser can open all devices
37402                                  <1> sysopen_dev_perm_err:
37403 0000CBDA B80B000000              <1>       mov    eax, ERR_DEV_ACCESS  ; 11  = 'permission denied !'
37404 0000CBDF EBA5                    <1>       jmp    short sysopen_dev_err
37405                                  <1> sysopen_dev_4:
37406 0000CBE1 D0ED                    <1>       shr    ch, 1 ; result: 1 = read, 2 = write, 3 = r & w
37407 0000CBE3 FEC9                    <1>       dec    cl  ; result: 1 = read, 2 = write
37408 0000CBE5 84E9                    <1>       test   cl, ch
37409 0000CBE7 74F1                    <1>       jz     short sysopen_dev_perm_err
37410                                  <1>
37411 0000CBE9 D0E5                    <1>       shl    ch, 1 ; bit 0 = 0
37412                                  <1>       ; eax = device number (entry offset)
37413 0000CBEB E8A42E0000              <1>       call   device_open
37414 0000CBF0 72E8                    <1>       jc     short sysopen_dev_perm_err
37415                                  <1>
37416                                  <1>       ; eax = device number (entry offset)
37417 0000CBF2 0C80                    <1>       or     al, 80h ; set device bit (set bit 7 to 1)
37418 0000CBF4 8B1D[64030300]          <1>       mov    ebx, [u.r0]
37419 0000CBFA 8883[6A030300]          <1>       mov    [ebx+u.fp], al      ; bit 7 (=1) points to device
37420                                  <1>
37421 0000CC00 E9D8F8FFFF              <1>       jmp    sysret
37422                                  <1>
37423                                  <1> sysmkdir: ; < make directory >
37424                                  <1>       ; 15/10/2016
37425                                  <1>       ; 10/10/2016 (TRDOS 386 = TRDOS v2.0)
37426                                  <1>       ;          -derived from INT_21H.ASM-
37427                                  <1>       ;           ("loc_INT21h_create_file")
37428                                  <1>         ;    10/07/2011 (12/03/2011)
37429                                  <1>         ;    INT 21h Function AH = 3Ch
37430                                  <1>         ;    Create File
37431                                  <1>         ;    INPUT
37432                                  <1>         ;       CX = Attributes
37433                                  <1>         ;           DS:DX= Address of zero terminaned path name
37434                                  <1>         ;
37435                                  <1>         ;
37436                                  <1>       ; 14/05/2015 (Retro UNIX 386 v1 - Beginning)
37437                                  <1>       ; 27/05/2013 - 02/08/2013 (Retro UNIX 8086 v1)
37438                                  <1>       ;
37439                                  <1>       ; 'sysmkdir' creates an empty directory whose name is
37440                                  <1>       ; pointed to by arg 1. The mode of the directory is arg 2.
37441                                  <1>       ; The special entries '.' and '..' are not present.
37442                                  <1>       ; Errors are indicated if the directory already exists or
37443                                  <1>       ; user is not the super user.
37444                                  <1>       ;
37445                                  <1>       ; Calling sequence:
37446                                  <1>       ;     sysmkdir; name; mode
37447                                  <1>       ; Arguments:
37448                                  <1>       ;     name - points to the name of the directory
37449                                  <1>       ;     mode - mode of the directory
37450                                  <1>       ; Inputs: (arguments)
37451                                  <1>       ; Outputs: -
37452                                  <1>       ;    (sets 'directory' flag to 1;
37453                                  <1>       ;    'set user id on execution' and 'executable' flags to 0)
37454                                  <1>       ; ..........................................................
37455                                  <1>       ;
37456                                  <1>       ; Retro UNIX 8086 v1 modification:
37457                                  <1>       ;       'sysmkdir' system call has two arguments; so,
37458                                  <1>       ;     * 1st argument, name is pointed to by BX register
37459                                  <1>       ;     * 2nd argument, mode is in CX register
37460                                  <1>       ;
37461                                  <1>       ; TRDOS 386 (10/10/2016)
37462                                  <1>       ;
37463                                  <1>         ; INPUT ->
37464                                  <1>         ;       CL = Directory Attributes
37465                                  <1>       ;            bit 0 (1) - Read only file/dir (R)
37466                                  <1>       ;             bit 1 (1) - Hidden file/dir (H)
37467                                  <1>         ;             bit 2 (1) - System file/dir (R)
37468                                  <1>       ;             bit 3 (1) - Volume label/name (V)
37469                                  <1>         ;             bit 4 (1) - Subdirectory (D)
37470                                  <1>       ;            bit 5 (1) - File/Dir has been archived (A)
37471                                  <1>       ;         CX = 0 -> create normal directory
37472                                  <1>         ;          EBX = Pointer to directory name (ASCIIZ) -path-
37473                                  <1>       ;
37474                                  <1>       ; OUTPUT ->
37475                                  <1>       ;          eax = First cluster of the new directory
37476                                  <1>       ;          cf = 1 -> Error code in AL
37477                                  <1>       ;
37478                                  <1>       ; Modified Registers: EAX (at the return of system call)
37479                                  <1>       ;
37480                                  <1>       ; Note: If the file or directory is existing
37481                                  <1>       ;     an access error will be returned.
37482                                  <1>
37483 0000CC05 6621C9                  <1>       and    cx, cx ; if cx = 0 -> create a normal subdir
37484 0000CC08 7413                    <1>       jz     short sysmkdir_1
37485                                  <1>
37486 0000CC0A F6C110                  <1>       test   cl, 10h ; if dir flags set, also use other flags
```

```
37487 0000CC0D 0F853EFDFFFF        <1>       jnz    sysmkdir_0 ; jump to head of 'syscreat'
37488                              <1>
37489                              <1>       ; CX has wrong flags
37490 0000CC13 B817000000          <1>       mov    eax, ERR_INV_FLAGS
37491 0000CC18 E969FFFFFF          <1>       jmp    sysopen_dev_err
37492                              <1>
37493                              <1> sysmkdir_1:
37494 0000CC1D B110                <1>       mov    cl, 10h ; set subdir flag and reset other flags
37495 0000CC1F E92DFDFFFF          <1>       jmp    sysmkdir_0 ; jump to head of 'syscreat'
37496                              <1> sysmkdir_2:
37497                              <1>       ; jump from 'syscreat' ; from 'syscreat_1'
37498                              <1>       ;  CL = Directory attributes/flags
37499 0000CC24 BE[CC5C0100]        <1>       mov    esi, FindFile_Name
37500 0000CC29 E802D7FFFF          <1>       call   make_sub_directory
37501 0000CC2E 0F821BFEFFFF        <1>       jc     sysopen_err       ; NOTE: Old type (TRDOS 8086)
37502                              <1>                                ; error codes must be modified
37503                              <1>                                ; for next TRDOS 386 versions
37504                              <1>                                ; (10/10/2016)
37505                              <1>                                ; Old (MSDOS type)
37506                              <1>                                ; error codes (2011):
37507                              <1>                                ;  2 = file not found
37508                              <1>                                ;  3 = directory not found
37509                              <1>                                ;  5 = access denied
37510                              <1>                                ; 12 = no more files
37511                              <1>                                  ; 19 = disk write protected
37512                              <1>                                ; 39 = insufficient disk space
37513                              <1>                                ; 'sysdefs.s' ; 10/10/2016
37514                              <1>
37515 0000CC34 A3[64030300]        <1>       mov    [u.r0], eax ; New sub dir's first cluster
37516                              <1>
37517 0000CC39 E8922A0000          <1>         call       reset_working_path
37518                              <1>
37519 0000CC3E E99AF8FFFF          <1>       jmp    sysret
37520                              <1>
37521                              <1> sysclose: ;<close file>
37522                              <1>       ; 06/10/2016 (TRDOS 386 = TRDOS v2.0)
37523                              <1>       ;
37524                              <1>       ; 14/05/2015 (Retro UNIX 386 v1 - Beginning)
37525                              <1>       ; 22/05/2013 - 26/05/2013 (Retro UNIX 8086 v1)
37526                              <1>       ;
37527                              <1>       ; 'sysclose', given a file descriptor in 'u.r0', closes the
37528                              <1>       ; associated file. The file descriptor (index to 'u.fp' list)
37529                              <1>       ; is put in r1 and 'fclose' is called.
37530                              <1>       ;
37531                              <1>       ; Calling sequence:
37532                              <1>       ;     sysclose
37533                              <1>       ; Arguments:
37534                              <1>       ;     -
37535                              <1>       ; Inputs: *u.r0 - file descriptor
37536                              <1>       ; Outputs: -
37537                              <1>       ; ..........................................................
37538                              <1>       ;
37539                              <1>       ; Retro UNIX 8086 v1 modification:
37540                              <1>       ;      The user/application program puts file descriptor
37541                              <1>       ;       in BX register as 'sysclose' system call argument.
37542                              <1>       ;      (argument transfer method 1)
37543                              <1>
37544                              <1>       ; TRDOS 386 (06/10/2016)
37545                              <1>       ;
37546                              <1>         ; INPUT ->
37547                              <1>         ;      EBX = File Handle/Number (file index) (AL)
37548                              <1>       ; OUTPUT ->
37549                              <1>       ;         cf = 0 -> EAX = 0
37550                              <1>       ;         cf = 1 -> Error code in EAX (ERR_FILE_NOT_OPEN)
37551                              <1>       ;
37552                              <1>       ; Modified Registers: EAX (at the return of system call)
37553                              <1>       ;
37554                              <1>
37555 0000CC43 89D8                <1>       mov    eax, ebx
37556 0000CC45 31DB                <1>       xor    ebx, ebx
37557 0000CC47 891D[64030300]      <1>       mov    [u.r0], ebx ; 0  ; return value of EAX
37558 0000CC4D E8AF0E0000          <1>       call   fclose
37559 0000CC52 0F8385F8FFFF        <1>       jnc    sysret
37560 0000CC58 B80A000000          <1>       mov    eax, ERR_FILE_NOT_OPEN ; file not open !
37561 0000CC5D A3[C8030300]        <1>       mov    [u.error], eax ;
37562 0000CC62 A3[64030300]        <1>       mov    [u.r0], eax ; ! invalid handle !
37563 0000CC67 E951F8FFFF          <1>       jmp    error
37564                              <1>
37565                              <1> sysread: ; < read from file >
37566                              <1>       ; 11/10/2016 (TRDOS 386 = TRDOS v2.0)
37567                              <1>       ;          -derived from INT_21H.ASM-
37568                              <1>       ;           ("loc_INT21h_read_file")
37569                              <1>         ;   13/03/2011 (05/03/2011)
37570                              <1>         ;   INT 21h Function AH = 3Fh
37571                              <1>         ;   Read from a File
37572                              <1>         ;   INPUT
37573                              <1>       ;        BX = File Handle
37574                              <1>         ;      CX = Number of bytes to read
37575                              <1>         ;          DS:DX= Buffer address
37576                              <1>         ;
37577                              <1>       ; Note: TRDOS 386 'sysread' has been derived from
37578                              <1>       ;     Retro UNIX 386 v1 'sysread', except a few
37579                              <1>       ;     code modifications.
37580                              <1>       ;
37581                              <1>       ; 13/05/2015 (Retro UNIX 386 v1)
37582                              <1>       ; 11/05/2015 (Retro UNIX 386 v1 - Beginning)
37583                              <1>       ; 23/05/2013 (Retro UNIX 8086 v1)
37584                              <1>       ;
37585                              <1>       ; 'sysread' is given a buffer to read into and the number of
37586                              <1>       ; characters to be read. If finds the file from the file
37587                              <1>       ; descriptor located in *u.r0 (r0). This file descriptor
37588                              <1>       ; is returned from a successful open call (sysopen).
37589                              <1>       ; The i-number of file is obtained via 'rw1' and the data
```

```
37590                                  <1>       ; is read into core via 'readi'.
37591                                  <1>       ;
37592                                  <1>       ; Calling sequence:
37593                                  <1>       ;     sysread; buffer; nchars
37594                                  <1>       ; Arguments:
37595                                  <1>       ;     buffer - location of contiguous bytes where
37596                                  <1>       ;              input will be placed.
37597                                  <1>       ;     nchars - number of bytes or characters to be read.
37598                                  <1>       ; Inputs: *u.r0 - file descriptor (& arguments)
37599                                  <1>       ; Outputs: *u.r0 - number of bytes read.
37600                                  <1>       ; ..........................................................
37601                                  <1>       ;
37602                                  <1>       ; Retro UNIX 8086 v1 modification:
37603                                  <1>       ;       'sysread' system call has three arguments; so,
37604                                  <1>       ;     * 1st argument, file descriptor is in BX register
37605                                  <1>       ;     * 2nd argument, buffer address/offset in CX register
37606                                  <1>       ;     * 3rd argument, number of bytes is in DX register
37607                                  <1>       ;
37608                                  <1>       ;     AX register (will be restored via 'u.r0') will return
37609                                  <1>       ;     to the user with number of bytes read.
37610                                  <1>       ;
37611                                  <1>       ; TRDOS 386 (05/10/2016)
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 read
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 0000CC6C E8DE0E0000               <1>       call  getf1
37626 0000CC71 7277                     <1>       jc    short device_read ; read data from device
37627                                  <1>       ; EAX = First cluster of the file
37628                                  <1>
37629 0000CC73 E83F000000               <1>       call  rw1
37630 0000CC78 730A                     <1>       jnc   short sysread_0
37631                                  <1>
37632 0000CC7A A3[64030300]             <1>       mov   [u.r0], eax ; error code
37633 0000CC7F E939F8FFFF               <1>       jmp   error
37634                                  <1>
37635                                  <1> sysread_0:
37636 0000CC84 E84B170000               <1>       call  readi
37637 0000CC89 EB1D                     <1>       jmp   short rw0
37638                                  <1>
37639                                  <1> syswrite: ; < write to file >
37640                                  <1>       ; 23/10/2016
37641                                  <1>       ; 11/10/2016 (TRDOS 386 = TRDOS v2.0)
37642                                  <1>       ;          -derived from INT_21H.ASM-
37643                                  <1>       ;           ("loc_INT21h_write_file")
37644                                  <1>         ;   13/03/2011 (05/03/2011)
37645                                  <1>         ;   INT 21h Function AH = 40h
37646                                  <1>         ;   Write to a File
37647                                  <1>         ;   INPUT
37648                                  <1>       ;        BX = File Handle
37649                                  <1>         ;      CX = Number of bytes to write
37650                                  <1>         ;         DS:DX= Buffer address
37651                                  <1>         ;
37652                                  <1>       ; Note: TRDOS 386 'sysrwrite' has been derived from
37653                                  <1>       ;     Retro UNIX 386 v1 'syswrite', except a few
37654                                  <1>       ;     code modifications.
37655                                  <1>       ;
37656                                  <1>
37657                                  <1>       ; 13/05/2015 (Retro UNIX 386 v1)
37658                                  <1>       ; 11/05/2015 (Retro UNIX 386 v1 - Beginning)
37659                                  <1>       ; 23/05/2013 (Retro UNIX 8086 v1)
37660                                  <1>       ;
37661                                  <1>       ; 'syswrite' is given a buffer to write onto an output file
37662                                  <1>       ; and the number of characters to write. If finds the file
37663                                  <1>       ; from the file descriptor located in *u.r0 (r0). This file
37664                                  <1>       ; descriptor is returned from a successful open or create call
37665                                  <1>       ; (sysopen or syscreat). The i-number of file is obtained via
37666                                  <1>       ; 'rw1' and buffer is written on the output file via 'write'.
37667                                  <1>       ;
37668                                  <1>       ; Calling sequence:
37669                                  <1>       ;     syswrite; buffer; nchars
37670                                  <1>       ; Arguments:
37671                                  <1>       ;     buffer - location of contiguous bytes to be writtten.
37672                                  <1>       ;     nchars - number of characters to be written.
37673                                  <1>       ; Inputs: *u.r0 - file descriptor (& arguments)
37674                                  <1>       ; Outputs: *u.r0 - number of bytes written.
37675                                  <1>       ; ..........................................................
37676                                  <1>       ;
37677                                  <1>       ; Retro UNIX 8086 v1 modification:
37678                                  <1>       ;       'syswrite' system call has three arguments; so,
37679                                  <1>       ;     * 1st argument, file descriptor is in BX register
37680                                  <1>       ;     * 2nd argument, buffer address/offset in CX register
37681                                  <1>       ;     * 3rd argument, number of bytes is in DX register
37682                                  <1>       ;
37683                                  <1>       ;     AX register (will be restored via 'u.r0') will return
37684                                  <1>       ;     to the user with number of bytes written.
37685                                  <1>       ;
37686                                  <1>       ; INPUT ->
37687                                  <1>         ;       EBX = File handle (descriptor/index)
37688                                  <1>       ;         ECX = Buffer address
37689                                  <1>         ;           EDX = Number of bytes
37690                                  <1>       ; OUTPUT ->
37691                                  <1>       ;          EAX = Number of bytes have been written
37692                                  <1>       ;          cf = 1 -> Error code in AL
```

```
37693                                  <1>       ;
37694                                  <1>       ; Modified Registers: EAX (at the return of system call)
37695                                  <1>       ;
37696                                  <1>
37697                                  <1>       ; EBX = File descriptor
37698 0000CC8B E8BF0E0000              <1>       call  getf1
37699 0000CC90 7274                    <1>       jc    short device_write ; write data to device
37700                                  <1>       ; EAX = First cluster of the file
37701                                  <1>       ; EBX = File number  (Open file number) ; 23/10/2016
37702                                  <1>
37703 0000CC92 E820000000              <1>       call  rw1
37704 0000CC97 730A                    <1>       jnc   short syswrite_0
37705 0000CC99 A3[64030300]            <1>       mov   [u.r0], eax ; error code
37706 0000CC9E E91AF8FFFF              <1>       jmp   error
37707                                  <1>
37708                                  <1> syswrite_0:
37709 0000CCA3 E8661E0000              <1>       call  writei
37710                                  <1> rw0: ; 1:
37711 0000CCA8 A1[8C030300]            <1>         mov  eax, [u.nread]
37712 0000CCAD A3[64030300]            <1>       mov   [u.r0], eax
37713 0000CCB2 E926F8FFFF              <1>       jmp   sysret
37714                                  <1>
37715                                  <1> rw1:
37716                                  <1>       ; 11/10/2016 (TRDOS 386 = TRDOS v2.0)
37717                                  <1>       ; 14/05/2015 (Retro UNIX 386 v1)
37718                                  <1>       ; 11/05/2015 (Retro UNIX 386 v1 - Beginning)
37719                                  <1>       ; 23/05/2013 - 24/05/2013 (Retro UNIX 8086 v1)
37720                                  <1>       ; System call registers: ebx, ecx, edx (through 'sysenter')
37721                                  <1>       ;
37722                                  <1>       ; EBX = File descriptor
37723                                  <1>       ;call  getf1 ; calling point in 'getf' from 'rw1'
37724                                  <1>       ;jc    short device_rw ; read/write data from/to device
37725                                  <1>       ; EAX = First cluster of the file
37726                                  <1>
37727 0000CCB7 83F802                  <1>       cmp   eax, 2
37728 0000CCBA 7217                    <1>       jb    short rw2
37729                                  <1>       ;
37730 0000CCBC 890D[84030300]          <1>       mov   [u.base], ecx       ; buffer address/offset
37731                                  <1>                                 ;(in the user's virtual memory space)
37732 0000CCC2 8915[88030300]          <1>       mov   [u.count], edx
37733                                  <1>
37734 0000CCC8 C705[C8030300]0000-     <1>         mov     dword [u.error], 0 ; reset the last error code
37735 0000CCD0 0000                    <1>
37736 0000CCD2 C3                      <1>       retn
37737                                  <1>
37738                                  <1> rw2:
37739 0000CCD3 B80A000000              <1>       mov   eax, ERR_FILE_NOT_OPEN ; file not open !
37740 0000CCD8 A3[C8030300]            <1>       mov   dword [u.error], eax
37741 0000CCDD C3                      <1>       retn
37742                                  <1> rw3:
37743 0000CCDE B80B000000              <1>       mov   eax, ERR_FILE_ACCESS ; permission denied !
37744 0000CCE3 A3[C8030300]            <1>       mov   dword [u.error], eax
37745 0000CCE8 F9                      <1>       stc
37746 0000CCE9 C3                      <1>       retn
37747                                  <1>
37748                                  <1> device_read:
37749                                  <1>       ; 11/10/2016 (TRDOS 386 = TRDOS v2.0)
37750                                  <1>       ; cl = DEV_OPENMODE ; open mode
37751                                  <1>       ; ch = DEV_ACCESS   ; access flags
37752                                  <1>       ; al = DEV_DRIVER   ; device number (eax)
37753                                  <1>
37754 0000CCEA F6C101                  <1>       test  cl, 1 ; 1 = read, 2 = write, 3 = read&write
37755 0000CCED 74EF                    <1>       jz    short rw3
37756                                  <1>
37757 0000CCEF 89C3                    <1>       mov   ebx, eax
37758 0000CCF1 66C1E302                <1>       shl   bx, 2 ; *4
37759                                  <1>
37760 0000CCF5 F6C580                  <1>       test  ch, 80h ; bit 7, installable device driver flag
37761 0000CCF8 7406                    <1>       jz    short d_read_2 ; Kernel device
37762                                  <1>       ; installable device
37763                                  <1> d_read_1:
37764 0000CCFA FFA3[98600100]          <1>         jmp  dword [ebx+IDEV_RADDR-4]
37765                                  <1> d_read_2:
37766 0000CD00 FFA3[500F0100]          <1>       jmp   dword [ebx+KDEV_RADDR-4]
37767                                  <1>
37768                                  <1> device_write:
37769                                  <1>       ; 11/10/2016 (TRDOS 386 = TRDOS v2.0)
37770                                  <1>       ; cl = DEV_OPENMODE ; open mode
37771                                  <1>       ; ch = DEV_ACCESS   ; access flags
37772                                  <1>       ; al = DEV_DRIVER   ; device number (eax)
37773                                  <1>
37774 0000CD06 F6C102                  <1>       test  cl, 2 ; 1 = read, 2 = write, 3 = read&write
37775 0000CD09 74D3                    <1>       jz    short rw3
37776                                  <1>
37777 0000CD0B 89C3                    <1>       mov   ebx, eax
37778 0000CD0D 66C1E302                <1>       shl   bx, 2 ; *4
37779                                  <1>
37780 0000CD11 F6C580                  <1>       test  ch, 80h ; bit 7, installable device driver flag
37781 0000CD14 7406                    <1>       jz    short d_write_2 ; Kernel device
37782                                  <1>       ; installable device
37783                                  <1> d_write_1:
37784 0000CD16 FFA3[B8600100]          <1>         jmp  dword [ebx+IDEV_WADDR-4]
37785                                  <1> d_write_2:
37786 0000CD1C FFA3[A00F0100]          <1>       jmp   dword [ebx+KDEV_WADDR-4]
37787                                  <1>
37788                                  <1>
37789                                  <1> sysemt: ; enable (or disable) multi tasking -time sharing-
37790                                  <1>       ;
37791                                  <1>       ; 23/05/2016 - TRDOS 386 (TRDOS v2.0)
37792                                  <1>       ; 14/05/2015 (Retro UNIX 386 v1)
37793                                  <1>       ; 10/12/2013 - 20/04/2014 (Retro UNIX 8086 v1)
37794                                  <1>       ;
37795                                  <1>       ; Retro UNIX 8086 v1 modification:
```

```
37796                                  <1>       ;     'Enable Multi Tasking'  system call instead
37797                                  <1>       ;     of 'Emulator Trap' in original UNIX v1 for PDP-11.
37798                                  <1>       ;
37799                                  <1>       ; Retro UNIX 8086 v1 feature only!
37800                                  <1>       ;     Using purpose: Kernel will start without time-out
37801                                  <1>       ;     (internal clock/timer) functionality.
37802                                  <1>       ;     Then etc/init will enable clock/timer for
37803                                  <1>       ;     multi tasking.
37804                                  <1>       ;
37805                                  <1>       ; INPUT ->
37806                                  <1>       ;     BL = 0 -> disable multi tasking
37807                                  <1>       ;     BL > 1 -> enable multi tasking (time sharing)
37808                                  <1>       ; OUTPUT ->
37809                                  <1>       ;     none
37810                                  <1>       ;
37811                                  <1>       ;  Note: Multi tasking is disabled during system
37812                                  <1>       ;       initialization, it must be enabled by using
37813                                  <1>       ;       this system call. (Otherwise, running proces
37814                                  <1>       ;       will not be changed by another process within
37815                                  <1>       ;       run time sequence/schedule, if running process
37816                                  <1>       ;       will not 'release' itself. Only 'wakeup' procedure
37817                                  <1>       ;       for waiting processes and programmed timer events
37818                                  <1>       ;       for other processes can change running process
37819                                  <1>       ;       while multi tasking is disabled.) ** 23/05/2016 **
37820                                  <1>
37821 0000CD22 803D[B0030300]00       <1>       cmp    byte [u.uid], 0 ; root ?
37822                                  <1>       ;ja    error
37823 0000CD29 0F87D3F8FFFF           <1>       ja     badsys ; 14/05/2015
37824                                  <1>       ;
37825 0000CD2F FA                     <1>       cli
37826 0000CD30 881D[B65F0100]         <1>       mov    [multi_tasking], bl ; 0 to disable, >0 to enable
37827 0000CD36 E9A2F7FFFF             <1>       jmp    sysret
37828                                  <1>
37829                                  <1> systimer:
37830                                  <1>       ; 02/01/2017
37831                                  <1>       ; 21/12/2016
37832                                  <1>       ; 19/12/2016
37833                                  <1>       ; 10/12/2016 (callback)
37834                                  <1>       ; 10/06/2016
37835                                  <1>       ; 07/06/2016
37836                                  <1>       ; 06/06/2016
37837                                  <1>       ; 21/05/2016
37838                                  <1>       ; 19/05/2016
37839                                  <1>       ; 18/05/2016 - TRDOS 386 (TRDOS v2.0)
37840                                  <1>       ; (TRDOS 386 feature only!)
37841                                  <1>       ;
37842                                  <1>       ; (start or stop timer event(s))
37843                                  <1>       ;
37844                                  <1>       ; INPUT ->
37845                                  <1>       ;     BL = Signal return byte (response byte)
37846                                  <1>       ;          (Any requested value between 0 and 255)
37847                                  <1>       ;          (Kernel will put it at the requested address)
37848                                  <1>       ;     BH = Time count unit
37849                                  <1>       ;          0 = Stop timer event
37850                                  <1>       ;          1 = 18.2 ticks per second
37851                                  <1>       ;          2 = 10 milliseconds
37852                                  <1>       ;          3 = 1 second (for real time clock interrupt)
37853                                  <1>       ;          4 = time/tick count in current time count unit
37854                                  <1>       ;         // 10/12/2016
37855                                  <1>       ;         80h = Stop timer event (callback method)
37856                                  <1>       ;         81h = 18.2 ticks per second, callback method
37857                                  <1>       ;         82h = 10 milliseconds, callback method
37858                                  <1>       ;         83h = 1 second (for RTC int), callback method
37859                                  <1>       ;         84h = current time count unit, callback method
37860                                  <1>       ;
37861                                  <1>       ;         Note: Only 03h or 83h will set real time clock
37862                                  <1>       ;             (RTC) events (Others are for PIT events)!
37863                                  <1>       ;
37864                                  <1>       ;     NOTE: If callback (user service) method is used,
37865                                  <1>       ;         EDX will point to the return address (of service
37866                                  <1>       ;         procedure) in user's space instead of signal
37867                                  <1>       ;         response byte address. (TRDOS 386 kernel will
37868                                  <1>       ;         direct the cpu to that address -in user's space-
37869                                  <1>       ;         at the return of system call or interrupt
37870                                  <1>       ;         just after the adjusted count/time is elapsed.)
37871                                  <1>       ;         User's sevice routine must be ended with a
37872                                  <1>       ;         'iret'. Normal return addresses from system
37873                                  <1>       ;         calls or and interrupts will be kept same except
37874                                  <1>       ;         the timer returns.
37875                                  <1>       ;
37876                                  <1>       ;     BH = 0 -> Stop timer event
37877                                  <1>       ;     BL = Timer event number (1 to 255) if BH = 0
37878                                  <1>       ;          If BL = 0, all timer events (which are belongs
37879                                  <1>       ;           to running process) will be stopped
37880                                  <1>       ;     ECX = Time/Tick count (depending on time count unit)
37881                                  <1>       ;     EDX = Signal return (Response) byte address
37882                                  <1>       ;          (virtual address in user's memory space)
37883                                  <1>       ; OUTPUT ->
37884                                  <1>       ;     AL = Timer event number   (1 to 255) (max. value = 16)
37885                                  <1>       ;     IF BH Input = 0 & CF = 0 & AL = 0 ->
37886                                  <1>       ;          timer event(s) has/have been stopped/finished
37887                                  <1>       ;     CF = 1 & AL = 0 -> no timer setting space to set
37888                                  <1>       ;     CF = 1 & AL > 0 -> timer count unit is not usable
37889                                  <1>       ;
37890                                  <1>       ;     NOTE: To modify a time count for a user function,
37891                                  <1>       ;           at first, current timer event must be stopped
37892                                  <1>       ;           then a new timer event (which is related with
37893                                  <1>       ;           same user function) must be started.
37894                                  <1>       ;
37895                                  <1>       ;           Signal return (response) byte may be used for
37896                                  <1>       ;           several purposes. Kernel will put this value
37897                                  <1>       ;           to requested address during timer interrupt,
37898                                  <1>       ;           program/user can check this value to understand
```

```
37899                                  <1>       ;           which event has been occurred and what is changed.
37900                                  <1>       ;           (Multi timer events can share same signal address)
37901                                  <1>       ;
37902                                  <1>       ;     NOTE: If the process is running while the time count
37903                                  <1>       ;           is reached, kernel will put signal return (response)
37904                                  <1>       ;           byte value at requested address during timer
37905                                  <1>       ;           interrupt and the process will continue to run.
37906                                  <1>       ;           Program/process must call (jump to) it's timer event
37907                                  <1>       ;           function as required, for checking the timer event
37908                                  <1>       ;           status via signal return (response) byte address.
37909                                  <1>       ;
37910                                  <1>       ;           If the process is not running (waiting or sleeping
37911                                  <1>       ;           or released) while the time count is reached,
37912                                  <1>       ;           it is restarted from where it left, to ensure
37913                                  <1>       ;           proper multi media (video, audio, clock, timer)
37914                                  <1>       ;           functionality.
37915                                  <1>       ;
37916                                  <1>       ;           (It is better to use 'syswait' or 'syssleep',
37917                                  <1>       ;           or 'sysrele' system call just after the timer
37918                                  <1>       ;           function. Otherwise, timer events may block other
37919                                  <1>       ;           processes which are not using timer events.)

37920                                  <1>       ;
37921                                  <1>       ; Timer Event Structure: (max. 16 timer events, 16*16 bytes)
37922                                  <1>       ;      Owner:             resb 1 ; 0 = free
37923                                  <1>       ;                         ;>0 = process number (u.uno)
37924                                  <1>       ;     Calback:     resb 1 ; 1 = callback, 0 = response byte
37925                                  <1>       ;     Interrupt:      resb 1 ; 0 = Timer interrupt (or none)
37926                                  <1>       ;                         ; 1 = Real Time Clock interrupt
37927                                  <1>       ;     Response:       resb 1 ; 0 to 255, signal return value
37928                                  <1>       ;     Count Limit: resd 1 ; count of ticks (total/set)
37929                                  <1>       ;     Current Count:     resd 1 ; count of ticks (current)
37930                                  <1>       ;     Response Addr:  resd 1 ; response byte (pointer) address
37931                                  <1>       ;
37932                                  <1>
37933                                  <1>       ; 19/12/2016 (timer callback)
37934 0000CD3B C605[F4640100]00        <1>       mov   byte [tcallback], 0
37935 0000CD42 C605[F5640100]00        <1>       mov   byte [trtc], 0
37936 0000CD49 C705[D0030300]0000-     <1>       mov   dword [u.tcb], 0 ; this is not necessary...
37937 0000CD51 0000                    <1>
37938                                  <1>
37939 0000CD53 80FF80                  <1>       cmp   bh, 80h
37940 0000CD56 7225                    <1>       jb    short systimer_cb2
37941 0000CD58 7704                    <1>       ja    short systimer_cb0
37942                                  <1>
37943 0000CD5A 31D2                    <1>       xor   edx, edx ; 0, reset callback address
37944 0000CD5C EB0B                    <1>       jmp   short systimer_cb1
37945                                  <1>
37946                                  <1> systimer_cb0:
37947 0000CD5E 80FF84                  <1>       cmp   bh, 84h
37948 0000CD61 7764                    <1>       ja    short systimer_5 ;  undefined, error
37949                                  <1>
37950                                  <1>       ;mov  byte [tcallback], 1 ; 19/12/2016
37951 0000CD63 FE05[F4640100]          <1>       inc   byte [tcallback]
37952                                  <1>
37953                                  <1> systimer_cb1:
37954 0000CD69 0FB635[B3030300]        <1>       movzx esi, byte [u.uno] ; process number
37955 0000CD70 66C1E602                <1>       shl   si, 2
37956 0000CD74 8996[0C010300]          <1>       mov   [esi+p.tcb-4], edx ; set process timer callback address
37957                                  <1>                                ; (overwrite prev value if it is set!)
37958 0000CD7A 80E77F                  <1>       and   bh, 7Fh
37959                                  <1>
37960                                  <1> systimer_cb2:
37961 0000CD7D 80FF02                  <1>       cmp   bh, 2
37962 0000CD80 7445                    <1>         je      short systimer_5  ; only 18.2 ticks per second is usable
37963                                  <1>                                 ; 10 milliseconds (100 Hertz) timer
37964                                  <1>                                 ; will be set later (18/05/2016)
37965 0000CD82 774B                    <1>         ja      short systimer_6
37966                                  <1>
37967 0000CD84 20FF                    <1>       and   bh, bh
37968 0000CD86 0F84BA000000            <1>         jz      systimer_9        ; stop timer event(s)
37969                                  <1>
37970                                  <1>       ; bh = 1 (timer interrupt, 18.2 Hz, IBM PC/AT ROMBIOS default)
37971                                  <1>
37972                                  <1> systimer_19:
37973 0000CD8C B00A                    <1>       mov   al, 10 ; (*)
37974                                  <1>
37975                                  <1> systimer_0:
37976 0000CD8E B710                    <1>       mov   bh, 16
37977                                  <1>       ;
37978 0000CD90 383D[B75F0100]          <1>       cmp   [timer_events], bh ; 16 ; 07/06/2016
37979 0000CD96 7319                    <1>       jnb   short systimer_3  ; max. 16 timer events
37980                                  <1>       ;
37981 0000CD98 50                      <1>       push  eax ; (*)
37982                                  <1>
37983 0000CD99 BF[60040300]            <1>       mov   edi, timer_set  ; beginning address of timer events
37984                                  <1>                             ; setting space
37985 0000CD9E 30C0                    <1>       xor   al, al ; 0
37986                                  <1> systimer_1:
37987 0000CDA0 FEC0                    <1>       inc   al
37988 0000CDA2 803F00                  <1>       cmp   byte [edi], 0       ; is it free space ?
37989 0000CDA5 7639                    <1>       jna   short systimer_7 ; yes
37990 0000CDA7 FECF                    <1>       dec   bh
37991 0000CDA9 7405                    <1>       jz    short systimer_2
37992 0000CDAB 83C710                  <1>       add   edi, 16
37993 0000CDAE EBF0                    <1>       jmp   short  systimer_1 ; next event space
37994                                  <1>
37995                                  <1> systimer_2:
37996 0000CDB0 58                      <1>       pop   eax ; (*) discard
37997                                  <1> systimer_3:
37998 0000CDB1 C605[64030300]00        <1>       mov   byte [u.r0], 0
37999                                  <1> systimer_4:
38000 0000CDB8 C705[C8030300]1B00-     <1>         mov     dword [u.error], ERR_MISC
```

```
38001 0000CDC0 0000                <1>
38002                              <1>                                     ; one of miscellaneous/other errors
38003 0000CDC2 E9F6F6FFFF          <1>       jmp    error ; cf -> 1
38004                              <1>
38005                              <1> systimer_5:
38006 0000CDC7 883D[64030300]      <1>       mov    [u.r0], bh ; Time count unit (=2 or >3)
38007 0000CDCD EBE9                <1>       jmp    short systimer_4 ; 07/06/2016
38008                              <1>
38009                              <1> systimer_6:
38010 0000CDCF 80FF04              <1>       cmp    bh, 4
38011 0000CDD2 77F3                <1>         ja     short systimer_5  ; undefined time count unit
38012                              <1>       ;jb    short systimer_16
38013                              <1>
38014                              <1>       ;mov   al, 1  ; default (use current timer unit)
38015                              <1>                     ; countdown value is in ECX !
38016                              <1>                     ; max. value of ecx = 4294967296/10
38017                              <1>         ;jmp    short systimer_0
38018                              <1>       ;jmp   short systimer_19
38019 0000CDD4 74B6                <1>       je     short systimer_19
38020                              <1>
38021                              <1> systimer_16:
38022                              <1>       ; bh = 3
38023                              <1>       ; timer event via real time clock interrupt
38024                              <1>       ; interrupt/update frequency: 1 Hz (1 tick per second)
38025                              <1>
38026 0000CDD6 B0B6                <1>       mov    al, 182 ; (*) ; 18.2 * 10
38027 0000CDD8 FE05[F5640100]      <1>       inc    byte [trtc] ; timer event via real time clock
38028 0000CDDE EBAE                <1>         jmp     short systimer_0
38029                              <1>
38030                              <1> systimer_7:
38031 0000CDE0 A2[64030300]        <1>       mov    [u.r0], al ; timer event number
38032                              <1>       ;
38033                              <1>       ; edi = address of empty timer event area
38034 0000CDE5 A0[B3030300]        <1>       mov    al, [u.uno]
38035 0000CDEA FA                  <1>       cli    ; disable interrupts
38036 0000CDEB AA                  <1>       stosb  ; process number
38037 0000CDEC A0[F4640100]        <1>       mov    al, [tcallback] ; timer callback flag
38038 0000CDF1 AA                  <1>       stosb  ; 1= callback method, 0= signal response byte method
38039 0000CDF2 A0[F5640100]        <1>       mov    al, [trtc] ; timer interrupt type
38040 0000CDF7 AA                  <1>       stosb  ; 1= real time clock, 0= programmable interval timer
38041 0000CDF8 88D8                <1>       mov    al, bl ; Signal return (Response) value
38042 0000CDFA AA                  <1>       stosb   ; response byte
38043 0000CDFB 58                  <1>       pop    eax ; (*) ; 10 or 182
38044 0000CDFC 89D3                <1>       mov    ebx, edx ; virtual address for response/signal byte
38045 0000CDFE F7E1                <1>       mul    ecx
38046                              <1>       ; (eax = 10 * count of 18.2 Hz timer ticks)
38047                              <1>       ; (count down step = 10)
38048 0000CE00 AB                  <1>       stosd  ; count limit (reset value)
38049 0000CE01 AB                  <1>       stosd  ; current count value
38050                              <1>
38051                              <1>       ; 19/12/2016
38052 0000CE02 803D[F4640100]00    <1>       cmp    byte [tcallback], 0 ; timer callback method ?
38053 0000CE09 7604                <1>       jna    short systimer_17 ; no
38054 0000CE0B 89D8                <1>       mov    eax, ebx ; virtual address for callback routine
38055 0000CE0D EB0D                <1>       jmp    short systimer_18
38056                              <1>
38057                              <1> systimer_17: ; signal response byte method
38058                              <1>       ; ebx = virtual address
38059                              <1>       ; [u.pgdir] = page directory's physical address
38060                              <1>       ; 20/02/2017
38061 0000CE0F FE05[F6640100]      <1>       inc     byte [no_page_swap] ; 1
38062                              <1>                     ; Do not add this page to swap queue
38063                              <1>                     ; and remove it from swap queue if it is
38064                              <1>                     ; on the queue.
38065 0000CE15 E87584FFFF          <1>       call   get_physical_addr
38066 0000CE1A 721A                <1>       jc     short systimer_8 ; 07/06/2016
38067                              <1>       ; eax = physical address of the virtual address in user's space
38068                              <1> systimer_18:
38069 0000CE1C AB                  <1>       stosd  ; response addr (physical) or callback addr (virtual)
38070 0000CE1D FE05[B75F0100]      <1>       inc    byte [timer_events] ; 07/06/201
38071                              <1>       ; 02/01/2017
38072 0000CE23 0FB605[B3030300]    <1>       movzx  eax, byte [u.uno]
38073 0000CE2A FE80[FF000300]      <1>       inc    byte [eax+p.timer-1]
38074                              <1>       ;
38075 0000CE30 FB                  <1>       sti    ; enable interrupts
38076 0000CE31 E9A7F6FFFF          <1>       jmp    sysret
38077                              <1>
38078                              <1> systimer_8:
38079                              <1>       ; 10/06/2016
38080                              <1>       ; 07/06/2016
38081 0000CE36 28C0                <1>       sub    al, al ; 0
38082 0000CE38 8847F4              <1>       mov    [edi-12], al ; clear process number (free timer event)
38083                              <1>       ;mov   dword [edi], eax ; 0
38084 0000CE3B FB                  <1>       sti
38085 0000CE3C A2[64030300]        <1>       mov    [u.r0], al ; 0
38086 0000CE41 E977F6FFFF          <1>       jmp    error
38087                              <1>
38088                              <1> systimer_9:
38089                              <1>       ; 10/06/2016
38090                              <1>       ; 07/06/2016
38091 0000CE46 28C0                <1>       sub    al, al
38092 0000CE48 A2[64030300]        <1>       mov    byte [u.r0], al ; 0
38093 0000CE4D 3805[B75F0100]      <1>       cmp     byte [timer_events], al ;  0
38094 0000CE53 7631                <1>       jna    short systimer_12
38095                              <1>
38096                              <1>       ; Note: ecx and edx are undefined here
38097                              <1>       ;      (for stop timer function)
38098                              <1>
38099 0000CE55 BE[60040300]        <1>       mov    esi, timer_set  ; beginning address of timer events
38100                              <1>                               ; setting space
38101 0000CE5A A0[B3030300]        <1>       mov    al, [u.uno]
38102                              <1>
38103 0000CE5F B710                <1>       mov    bh, 16
```

```
38104                                  <1>
38105 0000CE61 08DB                    <1>       or    bl, bl
38106 0000CE63 7544                    <1>       jnz   short systimer_15
38107                                  <1>
38108                                  <1>       ; clear timer event areas belong to current process
38109                                  <1>       ; (for stopping all timer events belong to current process)
38110 0000CE65 FA                      <1>       cli   ; disable interrupts
38111                                  <1> systimer_10:
38112                                  <1>       ; 10/06/2016
38113                                  <1>       ; 07/06/2016
38114 0000CE66 8A26                    <1>       mov   ah, [esi]
38115 0000CE68 08E4                    <1>       or    ah, ah ; 0 ?
38116 0000CE6A 7411                    <1>       jz    short systimer_11
38117 0000CE6C 38C4                    <1>       cmp   ah, al ; is the process number (owner) same ?
38118 0000CE6E 750D                    <1>         jne     short systimer_11 ; no
38119                                  <1>
38120                                  <1>       ;mov   byte [esi], 0
38121 0000CE70 66C7060000              <1>       mov   word [esi], 0 ; clear
38122                                  <1>       ;mov   dword [esi+12], 0 ; clear
38123                                  <1>
38124 0000CE75 FE0D[B75F0100]          <1>       dec   byte [timer_events]
38125 0000CE7B 7409                    <1>       jz    short systimer_12
38126                                  <1>
38127                                  <1> systimer_11:
38128 0000CE7D FECF                    <1>       dec   bh
38129 0000CE7F 7405                    <1>       jz    short systimer_12
38130 0000CE81 83C610                  <1>       add   esi, 16
38131 0000CE84 EBE0                    <1>       jmp   short systimer_10
38132                                  <1>
38133                                  <1> systimer_12:
38134 0000CE86 0FB635[B3030300]        <1>       movzx esi, byte [u.uno]
38135 0000CE8D 08DB                    <1>       or    bl, bl ; all timer events or one timer event ?
38136 0000CE8F 740C                    <1>       jz    short systimer_13
38137 0000CE91 8A9E[FF000300]          <1>       mov   bl, [esi+p.timer-1]
38138 0000CE97 20DB                    <1>       and   bl, bl ; previous number of timer events for the process
38139 0000CE99 7408                    <1>       jz    short systimer_14
38140 0000CE9B FECB                    <1>       dec   bl  ; previous number of timer events for the process - 1
38141                                  <1> systimer_13:
38142 0000CE9D 889E[FF000300]          <1>       mov   [esi+p.timer-1], bl ; 0 ; no timer events for process
38143                                  <1> systimer_14:
38144 0000CEA3 FB                      <1>       sti   ; enable interrupts
38145 0000CEA4 E934F6FFFF              <1>       jmp   sysret
38146                                  <1>
38147                                  <1> systimer_15:
38148 0000CEA9 38FB                    <1>       cmp   bl, bh ; 16
38149 0000CEAB 0F8707FFFFFF            <1>         ja      systimer_4      ; max. 16 timer events !
38150                                  <1>       ;
38151 0000CEB1 88DA                    <1>       mov   dl, bl
38152 0000CEB3 FECA                    <1>       dec   dl  ; 16 -> 15 ... 1 -> 0
38153 0000CEB5 C0E204                  <1>       shl   dl, 4 ; * 16
38154 0000CEB8 0FB6FA                  <1>       movzx edi, dl
38155 0000CEBB 01F7                    <1>       add   edi, esi ; timer_set
38156                                  <1>
38157 0000CEBD 3A07                    <1>       cmp   al, [edi] ; process number
38158 0000CEBF 0F85F3FEFFFF            <1>         jne     systimer_4
38159                                  <1>
38160                                  <1>       ; same process ID
38161 0000CEC5 FA                      <1>       cli   ; disable interrupts
38162                                  <1>       ; 10/06/2016 ; 02/01/2017
38163                                  <1>       ;mov   byte [edi], 0
38164 0000CEC6 66C7070000              <1>       mov   word [edi], 0 ; clear
38165                                  <1>       ;mov   dword [edi+12], 0 ; clear
38166 0000CECB FE0D[B75F0100]          <1>       dec   byte [timer_events]
38167 0000CED1 EBB3                    <1>       jmp   short systimer_12
38168                                  <1>
38169                                  <1> sysmdate: ; < change the modification time of a file >
38170                                  <1>       ; 13/01/2017 - TRDOS 386 (TRDOS v2.0)
38171                                  <1>       ; temporary !
38172 0000CED3 B801000000              <1>       mov   eax, ERR_INV_FNUMBER ; 'invalid function number !'
38173 0000CED8 A3[C8030300]            <1>         mov     [u.error], eax
38174 0000CEDD A3[64030300]            <1>         mov     [u.r0], eax
38175 0000CEE2 E9D6F5FFFF              <1>       jmp   error
38176                                  <1>
38177                                  <1> sysvideo: ; VIDEO DATA TRANSFER FUNCTIONS
38178                                  <1>       ; 12/05/2017
38179                                  <1>       ; 11/07/2016
38180                                  <1>       ; 13/06/2016
38181                                  <1>       ; 16/05/2016 - TRDOS 386 (TRDOS v2.0)
38182                                  <1>       ;
38183                                  <1>       ;
38184                                  <1>       ; VIDEO DATA TRANSFER FUNCTIONS:
38185                                  <1>       ;
38186                                  <1>       ; Inputs:
38187                                  <1>       ;     BH = 0 = VIDEO BIOS Mode 3, tty/text mode data transfers
38188                                  <1>       ;          BL =
38189                                  <1>       ;            Bits 0&1, Transfer direction
38190                                  <1>       ;                  0 - System to system
38191                                  <1>       ;                  1 - User to system
38192                                  <1>       ;                  2 - System to user
38193                                  <1>       ;                  3 - User to user
38194                                  <1>       ;            Bits 2&3, Transfer Type
38195                                  <1>       ;                  0 - Display page transfer
38196                                  <1>       ;                  1 - Display page window transfer
38197                                  <1>       ;                  2 - Frame/Viewport/Window address transfer
38198                                  <1>       ;                  3 - Window handle transfer
38199                                  <1>       ;
38200                                  <1>       ;          /// BL = 0 -> System to system (display page) transfer
38201                                  <1>       ;             CL = Source page
38202                                  <1>       ;             DL = Destination page
38203                                  <1>       ;          /// BL = 1&2 -> user to system & system to user transfer
38204                                  <1>       ;             ECX = User buffer
38205                                  <1>       ;             DL = Video page
38206                                  <1>       ;          /// BL = 5&6 -> user to system, system to user transfer
```

```
38207 <1>       ;            (window in current display page and in current mode)
38208 <1>       ;             ESI = User's buffer address
38209 <1>       ;             ECX Low 16 bits = Top left column (X1 position)
38210 <1>       ;             ECX High 16 bits = Top row (Y1 position)
38211 <1>       ;             EDX Low 16 bits = Bottom right column (X2 position)
38212 <1>       ;             EDX High 16 bits = Bottom row (Y2 position)
38213 <1>         ;                If BL = 5 ->
38214 <1>       ;             EDI = Swap address (in user's memory space)
38215 <1>       ;             (If swap address > 0, previous content of the window
38216 <1>       ;             will be saved into swap area in user's memory space)
38217 <1>       ;           /// BL = 4 -> system to system transfer
38218 <1>       ;             ESI = System's source buffer (video page) address
38219 <1>       ;             ECX Low 16 bits = Top left column (X1 position)
38220 <1>       ;             ECX High 16 bits = Top row (Y1 position)
38221 <1>       ;             EDX Low 16 bits = Bottom right column (X2 position)
38222 <1>       ;             EDX High 16 bits = Bottom row (Y2 position)
38223 <1>       ;             EDI = System's destination buffer (video page) address
38224 <1>       ;
38225 <1>       ;     BH = 1 = CGA Graphics (0B8000h) data transfers
38226 <1>       ;          BL =
38227 <1>       ;            0 = Fill color (color in CL] (32K)
38228 <1>       ;            1 = User to system display page transfer
38229 <1>       ;            2 = System to user display page transfer
38230 <1>       ;            3 = NOT bits in window (ECX, EDX)
38231 <1>       ;            4 = Window copy (system to system)
38232 <1>       ;            5 = User to system window transfer
38233 <1>       ;            6 = System to user window transfer
38234 <1>       ;            7 = AND display page bytes with CL
38235 <1>       ;            8 = OR display page bytes with CL
38236 <1>       ;            9 = XOR display page bytes with CL
38237 <1>       ;
38238 <1>       ;           /// BL = 0 -> Fill color  (all screen pixels)
38239 <1>       ;             CL = Color value
38240 <1>       ;           /// BL = 1&2 -> user to system & system to user transfer
38241 <1>       ;             ECX = User buffer
38242 <1>       ;           /// BL = 5&6 -> user to system, system to user transfer
38243 <1>       ;            (window in current display page and in current mode)
38244 <1>       ;             ESI = User's buffer address
38245 <1>       ;             ECX Low 16 bits = Top left column (X1 position)
38246 <1>       ;             ECX High 16 bits = Top row (Y1 position)
38247 <1>       ;             EDX Low 16 bits = Bottom right column (X2 position)
38248 <1>       ;             EDX High 16 bits = Bottom row (Y2 position)
38249 <1>         ;         /// BL = 4 -> system to system (window) transfer
38250 <1>       ;             ESI = System's source buffer (video page) address
38251 <1>       ;             ECX Low 16 bits = Top left column (X1 position)
38252 <1>       ;             ECX High 16 bits = Top row (Y1 position)
38253 <1>       ;             EDX Low 16 bits = Bottom right column (X2 position)
38254 <1>       ;             EDX High 16 bits = Bottom row (Y2 position)
38255 <1>       ;             EDI = System's destination buffer (video page) address
38256 <1>       ;           /// BL = 3 -> NOT byte in display page/memory
38257 <1>       ;             ECX Low 16 bits = Top left column (X1 position)
38258 <1>       ;             ECX High 16 bits = Top row (Y1 position)
38259 <1>       ;             EDX Low 16 bits = Bottom right column (X2 position)
38260 <1>       ;             EDX High 16 bits = Bottom row (Y2 position)
38261 <1>       ;
38262 <1>       ;     BH = 2 = VGA Graphics (0A0000h) data transfers
38263 <1>       ;          BL =
38264 <1>       ;            x0h = Fill color (color in CL] (64K)
38265 <1>       ;            x1h = User to system display page transfer
38266 <1>       ;            x2h = System to user display page transfer
38267 <1>       ;            x3h = NOT bits in window (ECX, EDX)
38268 <1>       ;            x4h = Window copy (system to system)
38269 <1>       ;            x5h = User to system window transfer
38270 <1>       ;            x6h = System to user window transfer
38271 <1>       ;            x7h = AND display page bytes with CL
38272 <1>       ;            x8h = OR display page bytes with CL
38273 <1>       ;            x9h = XOR display page bytes with CL
38274 <1>       ;            x = 0 -> screen width = 320
38275 <1>       ;            x = 1 -> screen width = 640
38276 <1>       ;            x = 2 -> screen width = 800
38277 <1>       ;
38278 <1>       ;           /// BL = 0 -> Fill color  (all screen pixels)
38279 <1>       ;             CL = Color value
38280 <1>       ;           /// BL = 1&2 -> user to system & system to user transfer
38281 <1>       ;             ECX = User buffer
38282 <1>       ;           /// BL = 5&6 -> user to system, system to user transfer
38283 <1>       ;            (window in current display page and in current mode)
38284 <1>       ;             ESI = User's buffer address
38285 <1>       ;             ECX Low 16 bits = Top left column (X1 position)
38286 <1>       ;             ECX High 16 bits = Top row (Y1 position)
38287 <1>       ;             EDX Low 16 bits = Bottom right column (X2 position)
38288 <1>       ;             EDX High 16 bits = Bottom row (Y2 position)
38289 <1>             ;           /// BL = 4 -> system to system (window) transfer
38290 <1>       ;             ESI = System's source buffer (video page) address
38291 <1>       ;             ECX Low 16 bits = Top left column (X1 position)
38292 <1>       ;             ECX High 16 bits = Top row (Y1 position)
38293 <1>       ;             EDX Low 16 bits = Bottom right column (X2 position)
38294 <1>       ;             EDX High 16 bits = Bottom row (Y2 position)
38295 <1>       ;             EDI = System's destination buffer (video page) address
38296 <1>       ;           /// BL = 3 -> NOT byte in display page/memory
38297 <1>       ;             ECX Low 16 bits = Top left column (X1 position)
38298 <1>       ;             ECX High 16 bits = Top row (Y1 position)
38299 <1>       ;             EDX Low 16 bits = Bottom right column (X2 position)
38300 <1>       ;             EDX High 16 bits = Bottom row (Y2 position)
38301 <1>       ;
38302 <1>       ;     BH = 3 = Super VGA, LINEAR FRAME BUFFER data transfers
38303 <1>       ;          BL =
38304 <1>       ;            0 = Fill color (color in ECX] (Frame buffer size)
38305 <1>       ;            1 = User to system display page transfer
38306 <1>       ;            2 = System to user display page transfer
38307 <1>       ;            3 = NOT bits in window (ECX, EDX)
38308 <1>       ;            4 = Window copy (system to system)
38309 <1>       ;            5 = User to system window transfer
```

```
38310                                  <1>       ;            6 = System to user window transfer
38311                                  <1>       ;            7 = AND display page bytes with ECX
38312                                  <1>       ;            8 = OR display page bytes with ECX
38313                                  <1>       ;            9 = XOR display page bytes with ECX
38314                                  <1>       ;
38315                                  <1>       ;          /// BL = 0 -> Fill color  (all screen pixels)
38316                                  <1>       ;             CL = Color value
38317                                  <1>       ;          /// BL = 1&2 -> user to system & system to user transfer
38318                                  <1>       ;             ECX = User buffer
38319                                  <1>       ;          /// BL = 5&6 -> user to system, system to user transfer
38320                                  <1>       ;            (window in current display page and in current mode)
38321                                  <1>       ;             ESI = User's buffer address
38322                                  <1>       ;             ECX Low 16 bits = Top left column (X1 position)
38323                                  <1>       ;             ECX High 16 bits = Top row (Y1 position)
38324                                  <1>       ;             EDX Low 16 bits = Bottom right column (X2 position)
38325                                  <1>       ;             EDX High 16 bits = Bottom row (Y2 position)
38326                                  <1>       ;          /// BL = 4 -> system to system (window) transfer
38327                                  <1>       ;             ESI = System's source buffer (video page) address
38328                                  <1>       ;             ECX Low 16 bits = Top left column (X1 position)
38329                                  <1>       ;             ECX High 16 bits = Top row (Y1 position)
38330                                  <1>       ;             EDX Low 16 bits = Bottom right column (X2 position)
38331                                  <1>       ;             EDX High 16 bits = Bottom row (Y2 position)
38332                                  <1>       ;             EDI = System's destination buffer (video page) address
38333                                  <1>       ;          /// BL = 3 -> NOT byte in display page/memory
38334                                  <1>       ;             ECX Low 16 bits = Top left column (X1 position)
38335                                  <1>       ;             ECX High 16 bits = Top row (Y1 position)
38336                                  <1>       ;             EDX Low 16 bits = Bottom right column (X2 position)
38337                                  <1>       ;             EDX High 16 bits = Bottom row (Y2 position)
38338                                  <1>       ;
38339                                  <1>       ; Outputs:
38340                                  <1>       ;     EAX = transfer/byte count
38341                                  <1>       ;
38342                                  <1>       ;     NOTE: If the source or destination address passes out of
38343                                  <1>       ;     video pages (display memory limits), data will not be transferred
38344                                  <1>       ;     and EAX will return as 0.
38345                                  <1>       ;
38346                                  <1>       ;
38347                                  <1>       ; DIRECT (STANDARD VGA/CGA) DISPLAY MEMORY ACCESS FUNCTIONS:
38348                                  <1>       ;
38349                                  <1>       ;     BH = 4 = CGA direct video memory (0B8000h, 32K) access
38350                                  <1>       ;            Page directory & page tables of the user's
38351                                  <1>       ;            program will be updated to direct access to
38352                                  <1>       ;            0B8000h (32K) video (CGA, color) memory; if
38353                                  <1>       ;            there is not a permission  conflict or lock!
38354                                  <1>       ;             (User's program/process will have permision to
38355                                  <1>       ;            access locked display memory if the owner is
38356                                  <1>       ;            it's parent.)
38357                                  <1>         ;
38358                                  <1>       ;         Screen width = 320
38359                                  <1>       ;
38360                                  <1>       ;     BH = 5 = VGA direct video memory (0A0000h, 64K) access
38361                                  <1>       ;            Page directory & page tables of the user's
38362                                  <1>       ;            program will be updated to direct access to
38363                                  <1>       ;            0A0000h (64K) video (VGA) memory; if there is not
38364                                  <1>       ;            a permission conflict or lock!
38365                                  <1>       ;             (User's program/process will have permision to
38366                                  <1>       ;            access locked display memory if the owner is
38367                                  <1>       ;            it's parent.)
38368                                  <1>       ;
38369                                  <1>       ;         BL = Screen width (320, 640, 800)
38370                                  <1>       ;
38371                                  <1>       ; Outputs:
38372                                  <1>       ;     EAX = Display mmory address for direct access
38373                                  <1>       ;           0A0000h for VGA, 0B8000h for CGA
38374                                  <1>       ;     (Display memory size: 32K for CGA, 64K for VGA)
38375                                  <1>       ;     EAX = 0 if display page access permission has been denied.
38376                                  <1>       ;           (Locked!)
38377                                  <1>       ;
38378                                  <1>       ; LINEAR FRAME BUFFER ACCESS FUNCTIONS:
38379                                  <1>       ;
38380                                  <1>       ;     BH = 6 = Linear Frame Buffer direct video memory access
38381                                  <1>       ;
38382                                  <1>       ;            Page directory & page tables of the user's
38383                                  <1>       ;            program will be updated to direct access to
38384                                  <1>       ;            the configured LFB (Linear Frame Buffer) address,
38385                                  <1>       ;            if there is not a permission conflict or lock!
38386                                  <1>       ;             (User's program/process will have permision to
38387                                  <1>       ;            access locked display memory if the owner is
38388                                  <1>       ;            it's parent.)
38389                                  <1>       ;
38390                                  <1>       ;            Return: EAX = Linear Frame Buffer address
38391                                  <1>       ;                  EDX = Frame Buffer Size in bytes
38392                                  <1>       ;
38393                                  <1>       ;     BH = 7 = Get Linear Frame Buffer info (for current mode)
38394                                  <1>       ;
38395                                  <1>       ;            Return:
38396                                  <1>       ;            EAX = Frame Buffer Address (0 = is not in use)
38397                                  <1>       ;            EDX = Frame Buffer Size in bytes
38398                                  <1>       ;            BL = Current Video Mode
38399                                  <1>       ;                 BL = 0FFh -> Super VGA (Extended VGA)
38400                                  <1>       ;                 If BL = 0FFh,
38401                                  <1>         ;                       BH = 0 = 16 colors
38402                                  <1>       ;                  BH = 1 = 256 colors
38403                                  <1>       ;                  BH = 2 = 66536 colors
38404                                  <1>       ;                  BH = 3 = 24 bits TRUE (16M) colors
38405                                  <1>       ;                  BH = 4 = 32 bits TRUE (16M) colors
38406                                  <1>       ;            ECX = Pixel resolution
38407                                  <1>       ;                  CX = Width (640, 800, 1024, 1366, 1920)
38408                                  <1>       ;                  High 16 bits of ECX = Height
38409                                  <1>       ;
38410                                  <1>       ;     NOTE: Each process will have it's own frame buffer
38411                                  <1>       ;           address and resolution parameters in 'u' area.
38412                                  <1>       ;           Then, if the current frame buffer & resolution
```

```
38413                                  <1>       ;           is different, frame buffer r/w functions
38414                                  <1>       ;           will use scale factor to convert process's
38415                                  <1>         ;             pixel coordinates to actual screen coordinates.
38416                                  <1>       ;           resolution -> dimensional scale
38417                                  <1>       ;           color size -> color scale
38418                                  <1>       ;         * RGB (TRUE) colors to 256 colors conversion:
38419                                  <1>         ;             TRUE Colors -> 8,8,8 (R,G,B; byte 0 is R)
38420                                  <1>       ;           256 colors -> 2,2,2,2 (R,G,B,L; bit 0&1 is R)
38421                                  <1>       ;             bit 6&7 -> luminosity base level (0,1,2,3)
38422                                  <1>       ;             bit 4&5 -> blue level (0,1,2,3)
38423                                  <1>       ;             bit 2%3 -> green level (0,1,2,3)
38424                                  <1>       ;             bit 0&1 -> red level (0,1,2,3)
38425                                  <1>       ;           Example: total red level : luminosity + red level
38426                                  <1>       ;           Luminosity base level: 0 -> 16
38427                                  <1>       ;                                  1 -> 32
38428                                  <1>       ;                                  2 -> 64
38429                                  <1>       ;                                  3 -> 128
38430                                  <1>       ;           Color level:
38431                                  <1>       ;                                 0 -> 0
38432                                  <1>       ;                                 1 -> luminosity level
38433                                  <1>       ;                                 2 -> luminosity level + 64
38434                                  <1>       ;                                 3 -> 255
38435                                  <1>       ;         Luminosity base level = min (R,G,B)
38436                                  <1>       ;                 if it is <16, it will be set to 16
38437                                  <1>       ;         Color levels: Color values are fixed to (nearest)
38438                                  <1>       ;             one of all possible set level (step) values
38439                                  <1>       ;             (according to luminosity base level); then
38440                                  <1>       ;             color levels are set to R-L, G-L, B-L.
38441                                  <1>       ;     For example: If luminosity base level is 32
38442                                  <1>       ;            all possible set values are 0, 32, 96, 255.
38443                                  <1>       ;
38444                                  <1>       ;       * RGB (TRUE) colors to 16 colors conversion:
38445                                  <1>       ;         16 colors: R, B,G, L bits (4 bits)
38446                                  <1>       ;               If any one of R,G,B >= 128 L = 1
38447                                  <1>       ;               If max. value of (R,G,B) >= 32, it is 1
38448                                  <1>       ;                  else all color bits (R&G&B&L) are 0
38449                                  <1>       ;               If the second value >= max. value / 2
38450                                  <1>       ;                  it is 1
38451                                  <1>       ;               If third value value >= max. value / 2
38452                                  <1>       ;                  it is 1
38453                                  <1>       ;         Example: R = 132, G = 64, B = 78
38454                                  <1>       ;                 L = 1, R = 1
38455                                  <1>       ;                 G < 66 --> G = 0
38456                                  <1>       ;                 B >= 66 --> B = 1

38457                                  <1>
38458                                  <1>       ; 16/05/2016
38459 0000CEE7 31C0                    <1>       xor    eax, eax
38460 0000CEE9 A3[64030300]            <1>       mov    [u.r0], eax
38461                                  <1>
38462 0000CEEE 20FF                    <1>       and    bh, bh
38463 0000CEF0 0F8572020000            <1>       jnz    sysvideo_13 ; 11/07/2016
38464                                  <1>
38465                                  <1>       ; Video mode 0, 80*25 text mode, CGA 16 colors  ; [CRT_MODE] = 3
38466 0000CEF6 88DF                    <1>       mov    bh, bl
38467 0000CEF8 C0EF02                  <1>       shr    bh, 2
38468 0000CEFB 20FF                    <1>       and    bh, bh
38469 0000CEFD 0F8598000000            <1>         jnz     sysvideo_4
38470 0000CF03 BF00800B00              <1>       mov    edi, 0B8000h
38471 0000CF08 20D2                    <1>       and    dl, dl
38472 0000CF0A 7413                    <1>       jz     short sysvideo_1
38473 0000CF0C 80FA07                  <1>       cmp    dl, 7
38474 0000CF0F 0F87C8F5FFFF            <1>       ja     sysret
38475                                  <1> sysvideo_0:
38476 0000CF15 81C7A00F0000            <1>       add    edi, 80*25*2
38477 0000CF1B FECA                    <1>       dec    dl
38478 0000CF1D 75F6                    <1>       jnz    short sysvideo_0
38479                                  <1> sysvideo_1:
38480 0000CF1F 80E303                  <1>       and    bl, 3
38481 0000CF22 7530                    <1>       jnz    short sysvideo_2
38482 0000CF24 80F907                  <1>       cmp    cl, 7
38483 0000CF27 0F87B0F5FFFF            <1>       ja     sysret
38484                                  <1>       ; system to system video/display page transfer (mode 0)
38485 0000CF2D BE00800B00              <1>       mov    esi, 0B8000h
38486 0000CF32 0FB6C1                  <1>       movzx  eax, cl
38487 0000CF35 BAA00F0000              <1>       mov    edx, 80*25*2
38488 0000CF3A F7E2                    <1>       mul    edx
38489 0000CF3C 01C6                    <1>       add    esi, eax
38490 0000CF3E B9A00F0000              <1>       mov    ecx, (80*25*2)
38491 0000CF43 890D[64030300]          <1>       mov    [u.r0], ecx
38492 0000CF49 66C1E902                <1>       shr    cx, 2 ; /4
38493 0000CF4D F3A5                    <1>       rep    movsd
38494 0000CF4F E989F5FFFF              <1>       jmp    sysret
38495                                  <1> sysvideo_2:
38496 0000CF54 80FB02                  <1>       cmp    bl, 2
38497 0000CF57 0F8780F5FFFF            <1>         ja      sysret
38498 0000CF5D 721F                    <1>       jb     short sysvideo_3
38499                                  <1>       ; system to user video/display page transfer (mode 0)
38500 0000CF5F 89FE                    <1>       mov    esi, edi
38501 0000CF61 89CF                    <1>       mov    edi, ecx ; user buffer
38502 0000CF63 B9A00F0000              <1>       mov    ecx, 80*25*2
38503 0000CF68 E840190000              <1>       call   transfer_to_user_buffer ; fast transfer
38504 0000CF6D 0F826AF5FFFF            <1>       jc     sysret
38505 0000CF73 890D[64030300]          <1>       mov    [u.r0], ecx
38506 0000CF79 E95FF5FFFF              <1>       jmp    sysret
38507                                  <1> sysvideo_3:
38508                                  <1>       ; user to system video/display page transfer (mode 0)
38509 0000CF7E 89CE                    <1>       mov    esi, ecx ; user buffer
38510                                  <1>       ; edi = video page address
38511 0000CF80 B9A00F0000              <1>       mov    ecx, 80*25*2
38512 0000CF85 E86D190000              <1>       call   transfer_from_user_buffer ; fast transfer
38513 0000CF8A 0F824DF5FFFF            <1>       jc     sysret
38514 0000CF90 890D[64030300]          <1>       mov    [u.r0], ecx
```

```
38515 0000CF96 E942F5FFFF          <1>       jmp    sysret
38516                              <1> sysvideo_4:
38517 0000CF9B 80E303              <1>       and    bl, 3
38518 0000CF9E 0F85F6000000        <1>         jnz     sysvideo_9
38519 0000CFA4 80F907              <1>       cmp    cl, 7
38520 0000CFA7 0F8730F5FFFF        <1>       ja     sysret
38521                              <1>       ; system to system video/display page window transfer (mode 0)
38522 0000CFAD 81FE00800B00        <1>       cmp    esi, 0B8000h
38523 0000CFB3 0F8224F5FFFF        <1>       jb     sysret
38524 0000CFB9 81FE00FD0B00        <1>       cmp    esi, 0B8000h+(80*25*2*8)
38525 0000CFBF 0F8318F5FFFF        <1>       jnb    sysret
38526 0000CFC5 81FF00800B00        <1>       cmp    edi, 0B8000h
38527 0000CFCB 0F820CF5FFFF        <1>       jb     sysret
38528 0000CFD1 81FF00FD0B00        <1>         cmp     edi, 0B8000h+(80*25*2*8)
38529 0000CFD7 0F8300F5FFFF        <1>       jnb    sysret
38530                              <1>       ;
38531 0000CFDD 51                  <1>       push   ecx
38532 0000CFDE 52                  <1>       push   edx
38533 0000CFDF 0FB7C1              <1>       movzx  eax, cx ; top left column
38534 0000CFE2 50                  <1>       push   eax
38535 0000CFE3 C1E910              <1>       shr    ecx, 16 ; top row
38536 0000CFE6 66B8A000            <1>       mov    ax, 80*2 ; 80 colums, 160 bytes per row
38537 0000CFEA F7E1                <1>       mul    ecx
38538 0000CFEC 01C6                <1>       add    esi, eax
38539 0000CFEE 01C7                <1>       add    edi, eax
38540 0000CFF0 58                  <1>       pop    eax
38541 0000CFF1 66D1E0              <1>       shl    ax, 1 ; *2
38542 0000CFF4 01C6                <1>       add    esi, eax
38543 0000CFF6 01C7                <1>       add    edi, eax
38544 0000CFF8 5A                  <1>       pop    edx
38545 0000CFF9 59                  <1>       pop    ecx
38546 0000CFFA B800FD0B00          <1>       mov    eax, 0B8000h+(80*25*2*8)
38547 0000CFFF 39C6                <1>       cmp    esi, eax
38548 0000D001 0F83D6F4FFFF        <1>       jnb    sysret
38549 0000D007 39C6                <1>       cmp    esi, eax
38550 0000D009 0F83CEF4FFFF        <1>       jnb    sysret
38551                              <1>
38552 0000D00F 56                  <1>       push   esi ; ****
38553 0000D010 57                  <1>       push   edi ; ***
38554 0000D011 52                  <1>       push   edx ; **
38555 0000D012 51                  <1>       push   ecx ; *
38556 0000D013 C1E910              <1>       shr    ecx, 16 ; top row
38557 0000D016 C1EA10              <1>       shr    edx, 16 ; bottom row
38558 0000D019 83F918              <1>       cmp    ecx, 24 ; max. 25 rows
38559 0000D01C 7773                <1>       ja     short sysvideo_6
38560 0000D01E 83FA18              <1>       cmp    edx, 24 ; max. 25 rows
38561 0000D021 776E                <1>       ja     short sysvideo_6
38562 0000D023 28CA                <1>       sub    dl, cl
38563 0000D025 726A                <1>       jc     short sysvideo_6
38564 0000D027 50                  <1>       push   eax ; *****
38565 0000D028 89D3                <1>       mov    ebx, edx ; row count - 1
38566 0000D02A B8A0000000          <1>       mov    eax, 80*2
38567 0000D02F F7E0                <1>       mul    eax
38568 0000D031 01C6                <1>       add    esi, eax
38569 0000D033 01C7                <1>       add    edi, eax
38570 0000D035 58                  <1>       pop    eax ; *****
38571 0000D036 39C6                <1>       cmp    esi, eax
38572 0000D038 7757                <1>       ja     short sysvideo_6
38573 0000D03A 39C7                <1>       cmp    edi, eax
38574 0000D03C 7753                <1>       ja     short sysvideo_6
38575 0000D03E 59                  <1>       pop    ecx ; *
38576 0000D03F 5A                  <1>       pop    edx ; **
38577 0000D040 81E1FFFF0000        <1>       and    ecx, 0FFFFh
38578 0000D046 81E2FFFF0000        <1>       and    edx, 0FFFFh
38579 0000D04C 83F94F              <1>       cmp    ecx, 79 ; max. 80 columns
38580 0000D04F 7742                <1>       ja     short sysvideo_7
38581 0000D051 83FA4F              <1>       cmp    edx, 79 ; max. 80 columns
38582 0000D054 773D                <1>       ja     short sysvideo_7
38583 0000D056 28CA                <1>       sub    dl, cl
38584 0000D058 7639                <1>       jna    short sysvideo_7
38585                              <1>       ; edx = column count (width) - 1
38586 0000D05A D0E2                <1>       shl    dl, 1
38587 0000D05C 01D6                <1>       add    esi, edx
38588 0000D05E 01D7                <1>       add    edi, edx
38589 0000D060 39C6                <1>       cmp    esi, eax
38590 0000D062 772F                <1>       ja     short sysvideo_7
38591 0000D064 39C7                <1>       cmp    edi, eax
38592 0000D066 772B                <1>       ja     short sysvideo_7
38593 0000D068 5F                  <1>       pop    edi ; ***
38594 0000D069 5E                  <1>       pop    esi ; ****
38595 0000D06A FEC3                <1>       inc    bl
38596 0000D06C FEC2                <1>       inc    dl ; column count
38597 0000D06E 88D7                <1>       mov    bh, dl
38598 0000D070 D0E2                <1>       shl    dl, 1
38599 0000D072 B8A0000000          <1>       mov    eax, 80*2
38600 0000D077 28D0                <1>       sub    al, dl ; (80 - columns) * 2
38601                              <1> sysvideo_5:
38602 0000D079 88F9                <1>       mov    cl, bh
38603 0000D07B 0115[64030300]      <1>       add    [u.r0], edx
38604 0000D081 F366A5              <1>       rep    movsw
38605 0000D084 01C6                <1>       add    esi, eax ; next row
38606 0000D086 01C7                <1>       add    edi, eax ; next row
38607 0000D088 FECB                <1>       dec    bl
38608 0000D08A 75ED                <1>       jnz    short sysvideo_5
38609 0000D08C E94CF4FFFF          <1>       jmp    sysret
38610                              <1>
38611                              <1> sysvideo_6:
38612 0000D091 59                  <1>       pop    ecx ; *
38613 0000D092 5A                  <1>       pop    edx ; **
38614                              <1> sysvideo_7:
38615 0000D093 5F                  <1>       pop    edi ; ***
38616 0000D094 5E                  <1>       pop    esi ; ****
38617 0000D095 E943F4FFFF          <1>       jmp    sysret
```

```
38618                                  <1>
38619                                  <1> sysvideo_9:
38620 0000D09A 80FB02                  <1>       cmp   bl, 2
38621 0000D09D 0F873AF4FFFF            <1>         ja      sysret
38622                                  <1>
38623 0000D0A3 56                      <1>       push  esi ; ****
38624 0000D0A4 57                      <1>       push  edi ; ***
38625 0000D0A5 52                      <1>       push  edx ; **
38626 0000D0A6 51                      <1>       push  ecx ; *
38627                                  <1>
38628 0000D0A7 C1E910                  <1>       shr   ecx, 16 ; top row
38629 0000D0AA C1EA10                  <1>       shr   edx, 16 ; bottom row
38630 0000D0AD 83F918                  <1>       cmp   ecx, 24 ; max. 25 rows
38631 0000D0B0 77DF                    <1>       ja    short sysvideo_6
38632 0000D0B2 83FA18                  <1>       cmp   edx, 24 ; max. 25 rows
38633 0000D0B5 77DA                    <1>       ja    short sysvideo_6
38634 0000D0B7 28CA                    <1>       sub   dl, cl
38635 0000D0B9 72D6                    <1>       jc    short sysvideo_6
38636                                  <1>
38637 0000D0BB 88CD                    <1>       mov   ch, cl ; top row
38638 0000D0BD 8A0D[4E520100]          <1>       mov   cl, [ACTIVE_PAGE]
38639 0000D0C3 BFA00F0000              <1>       mov   edi, 80*25*2
38640 0000D0C8 D3E7                    <1>       shl   edi, cl
38641 0000D0CA 81C760700B00            <1>       add   edi, 0B8000h - 80*25*2
38642                                  <1>
38643 0000D0D0 88D7                    <1>       mov   bh, dl ; row count - 1
38644 0000D0D2 88EA                    <1>       mov   dl, ch ; top row
38645 0000D0D4 B8A0000000              <1>       mov   eax, 80*2
38646 0000D0D9 F7E2                    <1>       mul   edx
38647 0000D0DB 01C7                    <1>       add   edi, eax
38648                                  <1>
38649 0000D0DD 59                      <1>       pop   ecx ; *
38650 0000D0DE 5A                      <1>       pop   edx ; **
38651 0000D0DF 81E1FFFF0000            <1>       and   ecx, 0FFFFh
38652 0000D0E5 81E2FFFF0000            <1>       and   edx, 0FFFFh
38653 0000D0EB 83F94F                  <1>       cmp   ecx, 79 ; max. 80 columns
38654 0000D0EE 77A3                    <1>       ja    short sysvideo_7
38655 0000D0F0 83FA4F                  <1>       cmp   edx, 79 ; max. 80 columns
38656 0000D0F3 779E                    <1>       ja    short sysvideo_7
38657                                  <1>
38658 0000D0F5 28CA                    <1>       sub   dl, cl
38659 0000D0F7 769A                    <1>       jna   short sysvideo_7
38660                                  <1>
38661 0000D0F9 0FB6C1                  <1>       movzx eax, cl ; left column
38662 0000D0FC D0E0                    <1>       shl   al, 1  ; column * 2
38663 0000D0FE 01C7                    <1>       add   edi, eax
38664                                  <1>
38665 0000D100 FEC2                    <1>       inc   dl ; column count
38666 0000D102 D0E2                    <1>       shl   dl, 1
38667 0000D104 88D1                    <1>       mov   cl, dl ; column count * 2
38668 0000D106 B2A0                    <1>       mov   dl, 80*2
38669 0000D108 58                      <1>       pop   eax ; *** (swap address)
38670 0000D109 5E                      <1>       pop   esi ; ****
38671 0000D10A FEC7                    <1>       inc   bh
38672                                  <1>
38673                                  <1>       ;mov  edx, 80*2
38674 0000D10C B2A0                    <1>       mov   dl, 80*2
38675                                  <1>       ;
38676 0000D10E 80FB01                  <1>       cmp   bl, 1
38677 0000D111 7735                    <1>       ja    short sysvideo_11
38678                                  <1>
38679                                  <1>       ; user to system video/display page window transfer (mode 0)
38680 0000D113 21C0                    <1>       and   eax, eax ; swap address
38681 0000D115 7413                    <1>       jz    short sysvideo_10 ; no window swap
38682                                  <1>       ; save previous window content in user's buffer (swap address)
38683 0000D117 56                      <1>       push  esi ; user buffer
38684 0000D118 57                      <1>       push  edi ; beginning address of the window
38685 0000D119 89FE                    <1>       mov   esi, edi
38686 0000D11B 89C7                    <1>       mov   edi, eax
38687 0000D11D E88B170000              <1>       call  transfer_to_user_buffer ; fast transfer
38688 0000D122 5F                      <1>       pop   edi
38689 0000D123 5E                      <1>       pop   esi
38690 0000D124 0F82B3F3FFFF            <1>       jc    sysret
38691                                  <1> sysvideo_10:
38692                                  <1>       ; user to system video/display page window transfer (mode 0)
38693                                  <1>       ; esi =      user buffer
38694 0000D12A E8C8170000              <1>       call  transfer_from_user_buffer ; fast transfer
38695 0000D12F 0F82A8F3FFFF            <1>       jc    sysret
38696 0000D135 010D[64030300]          <1>       add   [u.r0], ecx
38697 0000D13B 01D7                    <1>       add   edi, edx ; next row
38698 0000D13D 01CE                    <1>       add   esi, ecx
38699 0000D13F FECF                    <1>       dec   bh
38700 0000D141 75E7                    <1>       jnz   short sysvideo_10
38701 0000D143 E995F3FFFF              <1>       jmp   sysret
38702                                  <1>
38703                                  <1> sysvideo_11:
38704                                  <1>       ; system to user video/display page window transfer (mode 0)
38705 0000D148 87FE                    <1>       xchg  edi, esi
38706                                  <1> sysvideo_12:
38707                                  <1>       ; esi = beginning address of the window
38708                                  <1>       ; edi =      user buffer
38709 0000D14A E85E170000              <1>       call  transfer_to_user_buffer ; fast transfer
38710 0000D14F 0F8288F3FFFF            <1>       jc    sysret
38711 0000D155 010D[64030300]          <1>       add   [u.r0], ecx
38712 0000D15B 01D6                    <1>       add   esi, edx ; next row
38713 0000D15D 01CF                    <1>       add   edi, ecx
38714 0000D15F FECF                    <1>       dec   bh
38715 0000D161 75E7                    <1>       jnz   short sysvideo_12
38716 0000D163 E975F3FFFF              <1>       jmp   sysret
38717                                  <1>
38718                                  <1> sysvideo_13:
38719 0000D168 80FF01                  <1>       cmp   bh, 1
38720 0000D16B 0F871F030000            <1>         ja      sysvideo_38
```

```
38721                                  <1>       ; BH = 1 = CGA Graphics (0B8000h) data transfers
38722                                  <1>
38723 0000D171 20DB                    <1>       and    bl, bl
38724 0000D173 751A                    <1>       jnz    short sysvideo_14
38725                                  <1>
38726                                  <1>       ; BL = 0 = Fill color (color in CL] (32K)
38727                                  <1>
38728 0000D175 88C8                    <1>       mov    al, cl
38729 0000D177 B900800000              <1>       mov    ecx, 32768
38730 0000D17C 66890D[64030300]        <1>       mov    [u.r0], cx
38731 0000D183 BF00800B00              <1>       mov    edi, 0B8000h
38732 0000D188 F3AB                    <1>       rep    stosd
38733 0000D18A E94EF3FFFF              <1>       jmp    sysret
38734                                  <1>
38735                                  <1> sysvideo_14:
38736 0000D18F 80FB01                  <1>       cmp    bl, 1
38737 0000D192 7723                    <1>       ja     short sysvideo_16
38738                                  <1>
38739 0000D194 89CE                    <1>       mov    esi, ecx ; user buffer
38740                                  <1>       ; BL = 1 = user to system video/display page transfer
38741                                  <1> sysvideo_15:
38742 0000D196 BF00800B00              <1>       mov    edi, 0B8000h
38743                                  <1>       ; edi = video page address
38744 0000D19B B900800000              <1>       mov    ecx, 32768
38745 0000D1A0 E852170000              <1>       call   transfer_from_user_buffer ; fast transfer
38746 0000D1A5 0F8232F3FFFF            <1>       jc     sysret ; [u.r0] = 0
38747 0000D1AB 66890D[64030300]        <1>       mov    [u.r0], cx
38748 0000D1B2 E926F3FFFF              <1>       jmp    sysret
38749                                  <1>
38750                                  <1> sysvideo_16:
38751 0000D1B7 80FB02                  <1>       cmp    bl, 2
38752 0000D1BA 7723                    <1>       ja     short sysvideo_18
38753                                  <1>
38754 0000D1BC 89CF                    <1>       mov    edi, ecx ; user buffer
38755                                  <1>       ; BL = 2 = system to user video/display page transfer
38756                                  <1> sysvideo_17:
38757 0000D1BE BE00800B00              <1>       mov    esi, 0B8000h
38758 0000D1C3 B900800000              <1>       mov    ecx, 32768
38759 0000D1C8 E8E0160000              <1>       call   transfer_to_user_buffer ; fast transfer
38760 0000D1CD 0F820AF3FFFF            <1>       jc     sysret ; [u.r0] = 0
38761 0000D1D3 66890D[64030300]        <1>       mov    [u.r0], cx
38762 0000D1DA E9FEF2FFFF              <1>       jmp    sysret
38763                                  <1>
38764                                  <1> sysvideo_18:
38765 0000D1DF 80FB03                  <1>       cmp    bl, 3
38766 0000D1E2 777E                    <1>       ja     short sysvideo_23
38767                                  <1>
38768                                  <1>       ; BL = 3 = NOT bits in window (ECX, EDX)
38769                                  <1>
38770 0000D1E4 BF00800B00              <1>       mov    edi, 0B8000h
38771 0000D1E9 89FE                    <1>       mov    esi, edi
38772                                  <1>
38773 0000D1EB 39CA                    <1>       cmp    edx, ecx ; bottom-right > top-left ?
38774 0000D1ED 7716                    <1>       ja     short sysvideo_20 ; window
38775                                  <1>       ; full screen (update)
38776 0000D1EF B900800000              <1>       mov    ecx, 32768
38777 0000D1F4 66890D[64030300]        <1>       mov    [u.r0], cx
38778                                  <1> sysvideo_19:
38779 0000D1FB F616                    <1>       not    byte [esi] ; NOT operation
38780 0000D1FD 46                      <1>       inc    esi
38781 0000D1FE E2FB                    <1>       loop   sysvideo_19
38782 0000D200 E9D8F2FFFF              <1>       jmp    sysret
38783                                  <1> sysvideo_20:
38784 0000D205 0FB7C2                  <1>       movzx  eax, dx ; bottom right column
38785 0000D208 6629C8                  <1>       sub    ax, cx  ; - top left column
38786 0000D20B 0F82CCF2FFFF            <1>         jb     sysret ; invalid
38787 0000D211 6640                    <1>       inc    ax ; same column no == 1 column
38788 0000D213 50                      <1>       push   eax ; byte count per window row
38789 0000D214 52                      <1>       push   edx
38790 0000D215 BB40010000              <1>       mov    ebx, 320 ; screen width
38791 0000D21A 89C8                    <1>       mov    eax, ecx
38792 0000D21C C1E810                  <1>       shr    eax, 16  ; top row
38793 0000D21F F7E3                    <1>       mul    ebx
38794 0000D221 6689CA                  <1>       mov    dx, cx ; top left column
38795 0000D224 01D0                    <1>       add    eax, edx
38796 0000D226 01C6                    <1>       add    esi, eax ; start address
38797 0000D228 59                      <1>       pop    ecx ; edx
38798 0000D229 89C8                    <1>       mov    eax, ecx
38799 0000D22B C1E810                  <1>       shr    eax, 16 ; bottom row
38800 0000D22E F7E3                    <1>       mul    ebx
38801 0000D230 6689CA                  <1>       mov    dx, cx ; bottom right column
38802 0000D233 01D0                    <1>       add    eax, edx
38803 0000D235 01C7                    <1>       add    edi, eax ; stop address (included)
38804 0000D237 5A                      <1>       pop    edx ; byte count per window row
38805 0000D238 81FFFFFF0B00            <1>       cmp    edi, 0BFFFFh
38806 0000D23E 0F8799F2FFFF            <1>       ja     sysret
38807 0000D244 56                      <1>       push   esi
38808 0000D245 4E                      <1>       dec    esi
38809                                  <1> sysvideo_21:
38810 0000D246 89D1                    <1>       mov    ecx, edx
38811                                  <1> sysvideo_22:
38812 0000D248 46                      <1>       inc    esi
38813 0000D249 F616                    <1>       not    byte [esi]
38814 0000D24B E2FB                    <1>       loop   sysvideo_22
38815 0000D24D 01DE                    <1>       add    esi, ebx ; bytes per screen row
38816                                  <1>       ;
38817 0000D24F 39FE                    <1>       cmp    esi, edi ; stop address (included in loop)
38818 0000D251 76F3                    <1>       jna    short sysvideo_21
38819 0000D253 5E                      <1>       pop    esi
38820 0000D254 29F7                    <1>       sub    edi, esi
38821 0000D256 66893D[64030300]        <1>       mov    [u.r0], di
38822 0000D25D E97BF2FFFF              <1>       jmp    sysret
38823                                  <1>
```

```
38824                                  <1> sysvideo_23:
38825 0000D262 80FB04                  <1>       cmp   bl, 4
38826 0000D265 0F87A7000000            <1>         ja      sysvideo_26
38827                                  <1>
38828                                  <1>       ; BL = 4 = window copy (system to system)
38829                                  <1>
38830 0000D26B B800800B00              <1>       mov   eax, 0B8000h
38831 0000D270 39C6                    <1>       cmp   esi, eax
38832 0000D272 0F8265F2FFFF            <1>       jb    sysret
38833 0000D278 39C7                    <1>       cmp   edi, eax
38834 0000D27A 0F825DF2FFFF            <1>       jb    sysret
38835 0000D280 6605FF7F                <1>       add   ax, 7FFFh ; 32767
38836 0000D284 39C6                    <1>       cmp   esi, eax
38837 0000D286 0F8751F2FFFF            <1>       ja    sysret
38838 0000D28C 39C7                    <1>       cmp   edi, eax
38839 0000D28E 0F8749F2FFFF            <1>       ja    sysret
38840                                  <1>
38841 0000D294 39CA                    <1>       cmp   edx, ecx ; bottom-right > top-left ?
38842 0000D296 7714                    <1>       ja    short sysvideo_24 ; window
38843                                  <1>       ; full screen copy
38844 0000D298 89C1                    <1>       mov   ecx, eax
38845 0000D29A 29F9                    <1>       sub   ecx, edi
38846 0000D29C 6641                    <1>       inc   cx
38847 0000D29E 66890D[64030300]        <1>       mov   [u.r0], cx
38848 0000D2A5 F3A4                    <1>       rep   movsb
38849 0000D2A7 E931F2FFFF              <1>       jmp   sysret
38850                                  <1> sysvideo_24:
38851 0000D2AC 0FB7C2                  <1>       movzx  eax, dx ; bottom right column
38852 0000D2AF 6629C8                  <1>       sub   ax, cx  ; - top left column
38853 0000D2B2 0F8225F2FFFF            <1>         jb      sysret ; invalid
38854 0000D2B8 6640                    <1>       inc   ax ; same column no == 1 column
38855 0000D2BA 50                      <1>       push   eax ; byte count per window row
38856                                  <1>       ;
38857 0000D2BB 52                      <1>       push   edx
38858 0000D2BC BB40010000              <1>       mov   ebx, 320 ; screen width
38859 0000D2C1 89C8                    <1>       mov   eax, ecx
38860 0000D2C3 C1E810                  <1>       shr   eax, 16      ; top row
38861 0000D2C6 F7E3                    <1>       mul   ebx
38862 0000D2C8 6689CA                  <1>       mov   dx, cx ; top left column
38863 0000D2CB 01D0                    <1>       add   eax, edx
38864 0000D2CD 01C7                    <1>       add   edi, eax ; start address
38865 0000D2CF 01C6                    <1>       add   esi, eax
38866 0000D2D1 59                      <1>       pop   ecx ; edx
38867 0000D2D2 89C8                    <1>       mov   eax, ecx
38868 0000D2D4 C1E810                  <1>       shr   eax, 16 ; bottom row
38869 0000D2D7 F7E3                    <1>       mul   ebx
38870 0000D2D9 6689CA                  <1>       mov   dx, cx ; bottom right column
38871 0000D2DC 01D0                    <1>       add   eax, edx
38872 0000D2DE 5A                      <1>       pop   edx ; byte count per window row
38873 0000D2DF 0500800B00              <1>       add   eax, 0B8000h
38874 0000D2E4 3DFFFF0B00              <1>       cmp   eax, 0BFFFFh
38875 0000D2E9 0F87EEF1FFFF            <1>       ja    sysret
38876 0000D2EF 57                      <1>       push   edi ; start address
38877 0000D2F0 50                      <1>       push   eax ; stop address (included)
38878                                  <1> sysvideo_25:
38879 0000D2F1 89D1                    <1>       mov   ecx, edx
38880 0000D2F3 F3A4                    <1>       rep   movsb
38881 0000D2F5 4F                      <1>       dec   edi
38882 0000D2F6 4E                      <1>       dec   esi
38883 0000D2F7 01DF                    <1>       add   edi, ebx ; bytes per screen row
38884 0000D2F9 01DE                    <1>       add   esi, ebx
38885                                  <1>       ;
38886 0000D2FB 3B3C24                  <1>       cmp   edi, [esp] ; stop addr(included in loop)
38887 0000D2FE 76F1                    <1>       jna   short sysvideo_25
38888 0000D300 5B                      <1>       pop   ebx ; stop address
38889 0000D301 5F                      <1>       pop   edi ; start address
38890 0000D302 29FB                    <1>       sub   ebx, edi
38891 0000D304 6643                    <1>       inc   bx
38892 0000D306 66891D[64030300]        <1>       mov   [u.r0], bx
38893 0000D30D E9CBF1FFFF              <1>       jmp   sysret
38894                                  <1>
38895                                  <1> sysvideo_26:
38896 0000D312 80FB05                  <1>       cmp   bl, 5
38897 0000D315 0F8795000000            <1>         ja      sysvideo_29
38898                                  <1>
38899                                  <1>       ; BL = 5 = window copy (user to system)
38900                                  <1>
38901 0000D31B B800800B00              <1>       mov   eax, 0B8000h
38902 0000D320 39C7                    <1>       cmp   edi, eax
38903 0000D322 0F82B5F1FFFF            <1>       jb    sysret
38904 0000D328 6605FF7F                <1>       add   ax, 7FFFh ; 32767
38905 0000D32C 39C7                    <1>       cmp   edi, eax
38906 0000D32E 0F87A9F1FFFF            <1>       ja    sysret
38907                                  <1>
38908                                  <1>       ; esi = user buffer (in user's memory space)
38909 0000D334 39CA                    <1>       cmp   edx, ecx ; bottom-right > top-left ?
38910 0000D336 0F865AFEFFFF            <1>         jna     sysvideo_15 ; full screen copy
38911                                  <1>
38912 0000D33C 0FB7C2                  <1>       movzx  eax, dx ; bottom right column
38913 0000D33F 6629C8                  <1>       sub   ax, cx  ; - top left column
38914 0000D342 0F8295F1FFFF            <1>         jb      sysret ; invalid
38915 0000D348 6640                    <1>       inc   ax ; same column no == 1 column
38916 0000D34A 50                      <1>       push   eax ; byte count per window row
38917                                  <1>
38918 0000D34B 52                      <1>       push   edx
38919 0000D34C BB40010000              <1>       mov   ebx, 320 ; screen width
38920 0000D351 89C8                    <1>       mov   eax, ecx
38921 0000D353 C1E810                  <1>       shr   eax, 16      ; top row
38922 0000D356 F7E3                    <1>       mul   ebx
38923 0000D358 6689CA                  <1>       mov   dx, cx ; top left column
38924 0000D35B 01D0                    <1>       add   eax, edx
38925 0000D35D 01C7                    <1>       add   edi, eax ; start address
38926 0000D35F 59                      <1>       pop   ecx ; edx
```

```
38927 0000D360 89C8              <1>       mov    eax, ecx
38928 0000D362 C1E810            <1>       shr    eax, 16 ; bottom row
38929 0000D365 F7E3              <1>       mul    ebx
38930 0000D367 6689CA            <1>       mov    dx, cx ; bottom right column
38931 0000D36A 01D0              <1>       add    eax, edx
38932 0000D36C 5A                <1>       pop    edx ; byte count per window row
38933 0000D36D 0500800B00        <1>       add    eax, 0B8000h
38934 0000D372 3DFFFF0B00        <1>       cmp    eax, 0BFFFFh
38935 0000D377 0F8760F1FFFF      <1>       ja     sysret
38936 0000D37D 57                <1>       push   edi ; start address
38937 0000D37E 50                <1>       push   eax ; stop address (included)
38938                            <1> sysvideo_27:
38939 0000D37F 89D1              <1>       mov    ecx, edx ; byte count
38940                            <1>       ; user to system video/display page window transfer
38941                            <1>       ; esi =      user buffer
38942 0000D381 E871150000        <1>       call   transfer_from_user_buffer ; fast transfer
38943 0000D386 7221              <1>       jc     short sysvideo_28
38944 0000D388 010D[64030300]    <1>       add    [u.r0], ecx
38945 0000D38E 01DF              <1>       add    edi, ebx ; next row
38946 0000D390 01CE              <1>       add    esi, ecx
38947 0000D392 3B3C24            <1>       cmp    edi, [esp] ; stop addr(included in loop)
38948 0000D395 76E8              <1>       jna    short sysvideo_27
38949 0000D397 5B                <1>       pop    ebx ; stop address
38950 0000D398 5F                <1>       pop    edi ; start address
38951 0000D399 29FB              <1>       sub    ebx, edi
38952 0000D39B 6643              <1>       inc    bx
38953 0000D39D 66891D[64030300]  <1>       mov    [u.r0], bx
38954 0000D3A4 E934F1FFFF        <1>       jmp    sysret
38955                            <1> sysvideo_28:
38956 0000D3A9 58                <1>       pop    eax
38957 0000D3AA 5A                <1>       pop    edx
38958 0000D3AB E92DF1FFFF        <1>       jmp    sysret
38959                            <1>
38960                            <1> sysvideo_29:
38961 0000D3B0 80FB06            <1>       cmp    bl, 6
38962 0000D3B3 0F8797000000      <1>         ja     sysvideo_32
38963                            <1>
38964                            <1>       ; BL = 6 = window copy (system to user)
38965                            <1>
38966 0000D3B9 89F7              <1>       mov    edi, esi ; user buffer
38967                            <1>
38968 0000D3BB B800800B00        <1>       mov    eax, 0B8000h
38969 0000D3C0 39C6              <1>       cmp    esi, eax
38970 0000D3C2 0F8215F1FFFF      <1>       jb     sysret
38971 0000D3C8 6605FF7F          <1>       add    ax, 7FFFh ; 32767
38972 0000D3CC 39C6              <1>       cmp    esi, eax
38973 0000D3CE 0F8709F1FFFF      <1>       ja     sysret
38974                            <1>
38975                            <1>       ; edi = user buffer (in user's memory space)
38976 0000D3D4 39CA              <1>       cmp    edx, ecx ; bottom-right > top-left ?
38977 0000D3D6 0F86E2FDFFFF      <1>         jna    sysvideo_17 ; full screen copy
38978                            <1>
38979 0000D3DC 0FB7C2            <1>       movzx  eax, dx ; bottom right column
38980 0000D3DF 6629C8            <1>       sub    ax, cx  ; - top left column
38981 0000D3E2 0F82F5F0FFFF      <1>         jb     sysret ; invalid
38982 0000D3E8 6640              <1>       inc    ax ; same column no == 1 column
38983 0000D3EA 50                <1>       push   eax ; byte count per window row
38984                            <1>
38985 0000D3EB 52                <1>       push   edx
38986 0000D3EC BB40010000        <1>       mov    ebx, 320 ; screen width
38987 0000D3F1 89C8              <1>       mov    eax, ecx
38988 0000D3F3 C1E810            <1>       shr    eax, 16      ; top row
38989 0000D3F6 F7E3              <1>       mul    ebx
38990 0000D3F8 6689CA            <1>       mov    dx, cx ; top left column
38991 0000D3FB 01D0              <1>       add    eax, edx
38992 0000D3FD 01C6              <1>       add    esi, eax ; start address
38993 0000D3FF 59                <1>       pop    ecx ; edx
38994 0000D400 89C8              <1>       mov    eax, ecx
38995 0000D402 C1E810            <1>       shr    eax, 16 ; bottom row
38996 0000D405 F7E3              <1>       mul    ebx
38997 0000D407 6689CA            <1>       mov    dx, cx ; bottom right column
38998 0000D40A 01D0              <1>       add    eax, edx
38999 0000D40C 5A                <1>       pop    edx ; byte count per window row
39000 0000D40D 0500800B00        <1>       add    eax, 0B8000h
39001 0000D412 3DFFFF0B00        <1>       cmp    eax, 0BFFFFh
39002 0000D417 0F87C0F0FFFF      <1>       ja     sysret
39003 0000D41D 56                <1>       push   esi ; start address
39004 0000D41E 50                <1>       push   eax ; stop address (included)
39005                            <1> sysvideo_30:
39006 0000D41F 89D1              <1>       mov    ecx, edx ; byte count
39007                            <1>       ; user to system video/display page window transfer
39008                            <1>       ; esi =      user buffer
39009 0000D421 E887140000        <1>       call   transfer_to_user_buffer ; fast transfer
39010 0000D426 7221              <1>       jc     short sysvideo_31
39011 0000D428 010D[64030300]    <1>       add    [u.r0], ecx
39012 0000D42E 01DF              <1>       add    edi, ebx ; next row
39013 0000D430 01CE              <1>       add    esi, ecx
39014 0000D432 3B3C24            <1>       cmp    edi, [esp] ; stop addr(included in loop)
39015 0000D435 76E8              <1>       jna    short sysvideo_30
39016 0000D437 5B                <1>       pop    ebx ; stop address
39017 0000D438 5F                <1>       pop    edi ; start address
39018 0000D439 29FB              <1>       sub    ebx, edi
39019 0000D43B 6643              <1>       inc    bx
39020 0000D43D 66891D[64030300]  <1>       mov    [u.r0], bx
39021 0000D444 E994F0FFFF        <1>       jmp    sysret
39022                            <1> sysvideo_31:
39023 0000D449 58                <1>       pop    eax
39024 0000D44A 5A                <1>       pop    edx
39025 0000D44B E98DF0FFFF        <1>       jmp    sysret
39026                            <1>
39027                            <1> sysvideo_32:
39028 0000D450 80FB07            <1>       cmp    bl, 7
39029 0000D453 770F              <1>       ja     short sysvideo_34
```

```
39030                                   <1>
39031                                   <1>     ; BL = 7 = AND display page bytes with CL
39032                                   <1>
39033 0000D455 BE00800B00               <1>     mov   esi, 0B8000h
39034 0000D45A B900800000               <1>     mov   ecx, 32768
39035                                   <1> sysvideo_33:
39036 0000D45F 200E                     <1>     and   byte [esi], cl
39037 0000D461 46                       <1>     inc   esi
39038 0000D462 E2FB                     <1>     loop  sysvideo_33
39039                                   <1>
39040                                   <1> sysvideo_34:
39041 0000D464 80FB08                   <1>     cmp   bl, 8
39042 0000D467 770F                     <1>     ja    short sysvideo_36
39043                                   <1>
39044                                   <1>     ; BL = 8 = OR display page bytes with CL
39045                                   <1>
39046 0000D469 BE00800B00               <1>     mov   esi, 0B8000h
39047 0000D46E B900800000               <1>     mov   ecx, 32768
39048                                   <1> sysvideo_35:
39049 0000D473 080E                     <1>     or    byte [esi], cl
39050 0000D475 46                       <1>     inc   esi
39051 0000D476 E2FB                     <1>     loop  sysvideo_35
39052                                   <1>
39053                                   <1> sysvideo_36:
39054 0000D478 80FB09                   <1>     cmp   bl, 9
39055 0000D47B 0F875CF0FFFF             <1>       ja    sysret ; nothing to do
39056                                   <1>
39057                                   <1>     ; BL = 9 = XOR display page bytes with CL
39058                                   <1>
39059 0000D481 BE00800B00               <1>     mov   esi, 0B8000h
39060 0000D486 B900800000               <1>     mov   ecx, 32768
39061                                   <1> sysvideo_37:
39062 0000D48B 300E                     <1>     xor   byte [esi], cl
39063 0000D48D 46                       <1>     inc   esi
39064 0000D48E E2FB                     <1>     loop  sysvideo_37
39065                                   <1>
39066                                   <1> sysvideo_38:
39067 0000D490 80FF02                   <1>     cmp   bh, 2
39068 0000D493 0F8733030000             <1>       ja    sysvideo_64
39069                                   <1>     ; BH = 2 = VGA Graphics (0A0000h) data transfers
39070                                   <1>
39071 0000D499 88DC                     <1>     mov   ah, bl
39072 0000D49B 80E30F                   <1>     and   bl, 0Fh
39073 0000D49E C0EC04                   <1>     shr   ah, 4
39074 0000D4A1 C1E310                   <1>     shl   ebx, 16
39075 0000D4A4 66BB4001                 <1>     mov   bx, 320 ; 320*200, 320*240
39076 0000D4A8 20E4                     <1>     and   ah, ah
39077 0000D4AA 7413                     <1>     jz    short sysvideo_39
39078 0000D4AC 66D1E3                   <1>     shl   bx, 1 ; 640*200, 640 * 400, 640*480
39079 0000D4AF 80FC02                   <1>     cmp   ah, 2
39080 0000D4B2 720B                     <1>     jb    short sysvideo_39
39081 0000D4B4 0F8723F0FFFF             <1>     ja    sysret ; invalid
39082                                   <1>     ; 800*600
39083 0000D4BA 6681C3A000               <1>     add   bx, 160 ; 800
39084                                   <1> sysvideo_39:
39085 0000D4BF C1CB10                   <1>     ror   ebx, 16
39086                                   <1>
39087 0000D4C2 20DB                     <1>     and   bl, bl
39088 0000D4C4 7519                     <1>     jnz   short sysvideo_40
39089                                   <1>
39090                                   <1>     ; BL = 0 = Fill color (color in CL] (64K)
39091                                   <1>
39092 0000D4C6 88C8                     <1>     mov   al, cl
39093 0000D4C8 B900000100               <1>     mov   ecx, 65536
39094 0000D4CD 890D[64030300]           <1>     mov   [u.r0], ecx
39095 0000D4D3 BF00000A00               <1>     mov   edi, 0A0000h
39096 0000D4D8 F3AB                     <1>     rep   stosd
39097 0000D4DA E9FEEFFFFF               <1>     jmp   sysret
39098                                   <1>
39099                                   <1> sysvideo_40:
39100 0000D4DF 80FB01                   <1>     cmp   bl, 1
39101 0000D4E2 7722                     <1>     ja    short sysvideo_42
39102                                   <1>
39103 0000D4E4 89CE                     <1>     mov   esi, ecx ; user buffer
39104                                   <1>     ; BL = 1 = user to system video/display page transfer
39105                                   <1> sysvideo_41:
39106 0000D4E6 BF00000A00               <1>     mov   edi, 0A0000h
39107                                   <1>     ; edi = video page address
39108 0000D4EB B900000100               <1>     mov   ecx, 65536
39109 0000D4F0 E802140000               <1>     call  transfer_from_user_buffer ; fast transfer
39110 0000D4F5 0F82E2EFFFFF             <1>     jc    sysret ; [u.r0] = 0
39111 0000D4FB 890D[64030300]           <1>     mov   [u.r0], ecx
39112 0000D501 E9D7EFFFFF               <1>     jmp   sysret
39113                                   <1>
39114                                   <1> sysvideo_42:
39115 0000D506 80FB02                   <1>     cmp   bl, 2
39116 0000D509 7722                     <1>     ja    short sysvideo_44
39117                                   <1>
39118 0000D50B 89CF                     <1>     mov   edi, ecx ; user buffer
39119                                   <1>     ; BL = 2 = system to user video/display page transfer
39120                                   <1> sysvideo_43:
39121 0000D50D BE00000A00               <1>     mov   esi, 0A0000h
39122 0000D512 B900000100               <1>     mov   ecx, 65536
39123 0000D517 E891130000               <1>     call  transfer_to_user_buffer ; fast transfer
39124 0000D51C 0F82BBEFFFFF             <1>     jc    sysret ; [u.r0] = 0
39125 0000D522 890D[64030300]           <1>     mov   [u.r0], ecx
39126 0000D528 E9B0EFFFFF               <1>     jmp   sysret
39127                                   <1>
39128                                   <1> sysvideo_44:
39129 0000D52D 80FB03                   <1>     cmp   bl, 3
39130 0000D530 777A                     <1>     ja    short sysvideo_49
39131                                   <1>
39132                                   <1>     ; BL = 3 = NOT bits in window (ECX, EDX)
```

```
39133                                  <1>
39134 0000D532 BF00000A00              <1>       mov    edi, 0A0000h
39135 0000D537 89FE                    <1>       mov    esi, edi
39136                                  <1>
39137 0000D539 39CA                    <1>       cmp    edx, ecx ; bottom-right > top-left ?
39138 0000D53B 770B                    <1>       ja     short sysvideo_45 ; window
39139                                  <1>       ; full screen (update)
39140 0000D53D B900000100              <1>       mov    ecx, 65536
39141 0000D542 890D[64030300]          <1>       mov    [u.r0], ecx
39142                                  <1> sysvideo_45:
39143 0000D548 F616                    <1>       not    byte [esi] ; NOT operation
39144 0000D54A 46                      <1>       inc    esi
39145 0000D54B E2FB                    <1>       loop   sysvideo_45
39146 0000D54D E98BEFFFFF              <1>       jmp    sysret
39147                                  <1> sysvideo_46:
39148 0000D552 0FB7C2                  <1>       movzx  eax, dx ; bottom right column
39149 0000D555 6629C8                  <1>       sub    ax, cx  ; - top left column
39150 0000D558 0F827FEFFFFF            <1>         jb      sysret ; invalid
39151 0000D55E 6640                    <1>       inc    ax ; same column no == 1 column
39152 0000D560 50                      <1>       push   eax ; byte count per window row
39153 0000D561 52                      <1>       push   edx
39154 0000D562 C1EB10                  <1>       shr    ebx, 16 ; 320,640,800 : screen width
39155 0000D565 89C8                    <1>       mov    eax, ecx
39156 0000D567 C1E810                  <1>       shr    eax, 16  ; top row
39157 0000D56A F7E3                    <1>       mul    ebx
39158 0000D56C 6689CA                  <1>       mov    dx, cx ; top left column
39159 0000D56F 01D0                    <1>       add    eax, edx
39160 0000D571 01C6                    <1>       add    esi, eax ; start address
39161 0000D573 59                      <1>       pop    ecx ; edx
39162 0000D574 89C8                    <1>       mov    eax, ecx
39163 0000D576 C1E810                  <1>       shr    eax, 16 ; bottom row
39164 0000D579 F7E3                    <1>       mul    ebx
39165 0000D57B 6689CA                  <1>       mov    dx, cx ; bottom right column
39166 0000D57E 01D0                    <1>       add    eax, edx
39167 0000D580 01C7                    <1>       add    edi, eax ; stop address (included)
39168 0000D582 5A                      <1>       pop    edx ; byte count per window row
39169 0000D583 81FFFFFF0A00            <1>       cmp    edi, 0AFFFFh
39170 0000D589 0F874EEFFFFF            <1>       ja     sysret
39171 0000D58F 56                      <1>       push   esi
39172 0000D590 4E                      <1>       dec    esi
39173                                  <1> sysvideo_47:
39174 0000D591 89D1                    <1>       mov    ecx, edx
39175                                  <1> sysvideo_48:
39176 0000D593 46                      <1>       inc    esi
39177 0000D594 F616                    <1>       not    byte [esi]
39178 0000D596 E2FB                    <1>       loop   sysvideo_48
39179 0000D598 01DE                    <1>       add    esi, ebx ; bytes per screen row
39180                                  <1>       ;
39181 0000D59A 39FE                    <1>       cmp    esi, edi ; stop address (included in loop)
39182 0000D59C 76F3                    <1>       jna    short sysvideo_47
39183 0000D59E 5E                      <1>       pop    esi
39184 0000D59F 29F7                    <1>       sub    edi, esi
39185 0000D5A1 893D[64030300]          <1>       mov    [u.r0], edi
39186 0000D5A7 E931EFFFFF              <1>       jmp    sysret
39187                                  <1>
39188                                  <1> sysvideo_49:
39189 0000D5AC 80FB04                  <1>       cmp    bl, 4
39190 0000D5AF 0F87A1000000            <1>         ja      sysvideo_52
39191                                  <1>
39192                                  <1>       ; BL = 4 = window copy (system to system)
39193                                  <1>
39194 0000D5B5 B800000A00              <1>       mov    eax, 0A0000h
39195 0000D5BA 39C6                    <1>       cmp    esi, eax
39196 0000D5BC 0F821BEFFFFF            <1>       jb     sysret
39197 0000D5C2 39C7                    <1>       cmp    edi, eax
39198 0000D5C4 0F8213EFFFFF            <1>       jb     sysret
39199 0000D5CA 6683C0FF                <1>       add    ax, 0FFFFh ; 65535
39200 0000D5CE 39C6                    <1>       cmp    esi, eax
39201 0000D5D0 0F8707EFFFFF            <1>       ja     sysret
39202 0000D5D6 39C7                    <1>       cmp    edi, eax
39203 0000D5D8 0F87FFEEFFFF            <1>       ja     sysret
39204                                  <1>
39205 0000D5DE 39CA                    <1>       cmp    edx, ecx ; bottom-right > top-left ?
39206 0000D5E0 7712                    <1>       ja     short sysvideo_50 ; window
39207                                  <1>       ; full screen copy
39208 0000D5E2 89C1                    <1>       mov    ecx, eax
39209 0000D5E4 29F9                    <1>       sub    ecx, edi
39210 0000D5E6 41                      <1>       inc    ecx
39211 0000D5E7 890D[64030300]          <1>       mov    [u.r0], ecx
39212 0000D5ED F3A4                    <1>       rep    movsb
39213 0000D5EF E9E9EEFFFF              <1>       jmp    sysret
39214                                  <1> sysvideo_50:
39215 0000D5F4 0FB7C2                  <1>       movzx  eax, dx ; bottom right column
39216 0000D5F7 6629C8                  <1>       sub    ax, cx  ; - top left column
39217 0000D5FA 0F82DDEEFFFF            <1>         jb      sysret ; invalid
39218 0000D600 6640                    <1>       inc    ax ; same column no == 1 column
39219 0000D602 50                      <1>       push   eax ; byte count per window row
39220                                  <1>       ;
39221 0000D603 52                      <1>       push   edx
39222 0000D604 C1EB10                  <1>       shr    ebx, 16 ; 320,640,800 : screen width
39223 0000D607 89C8                    <1>       mov    eax, ecx
39224 0000D609 C1E810                  <1>       shr    eax, 16      ; top row
39225 0000D60C F7E3                    <1>       mul    ebx
39226 0000D60E 6689CA                  <1>       mov    dx, cx ; top left column
39227 0000D611 01D0                    <1>       add    eax, edx
39228 0000D613 01C7                    <1>       add    edi, eax ; start address
39229 0000D615 01C6                    <1>       add    esi, eax
39230 0000D617 59                      <1>       pop    ecx ; edx
39231 0000D618 89C8                    <1>       mov    eax, ecx
39232 0000D61A C1E810                  <1>       shr    eax, 16 ; bottom row
39233 0000D61D F7E3                    <1>       mul    ebx
39234 0000D61F 6689CA                  <1>       mov    dx, cx ; bottom right column
39235 0000D622 01D0                    <1>       add    eax, edx
```

```
39236 0000D624 5A                <1>       pop    edx ; byte count per window row
39237 0000D625 0500000A00        <1>       add    eax, 0A0000h
39238 0000D62A 3DFFFF0A00        <1>       cmp    eax, 0AFFFFh
39239 0000D62F 0F87A8EEFFFF      <1>       ja     sysret
39240 0000D635 57                <1>       push   edi ; start address
39241 0000D636 50                <1>       push   eax ; stop address (included)
39242                            <1> sysvideo_51:
39243 0000D637 89D1              <1>       mov    ecx, edx
39244 0000D639 F3A4              <1>       rep    movsb
39245 0000D63B 4F                <1>       dec    edi
39246 0000D63C 4E                <1>       dec    esi
39247 0000D63D 01DF              <1>       add    edi, ebx ; bytes per screen row
39248 0000D63F 01DE              <1>       add    esi, ebx
39249                            <1>       ;
39250 0000D641 3B3C24            <1>       cmp    edi, [esp] ; stop addr(included in loop)
39251 0000D644 76F1              <1>       jna    short sysvideo_51
39252 0000D646 5B                <1>       pop    ebx ; stop address
39253 0000D647 5F                <1>       pop    edi ; start address
39254 0000D648 29FB              <1>       sub    ebx, edi
39255 0000D64A 43                <1>       inc    ebx
39256 0000D64B 891D[64030300]    <1>       mov    [u.r0], ebx
39257 0000D651 E987EEFFFF        <1>       jmp    sysret
39258                            <1>
39259                            <1> sysvideo_52:
39260 0000D656 80FB05            <1>       cmp    bl, 5
39261 0000D659 0F8791000000      <1>         ja     sysvideo_55
39262                            <1>
39263                            <1>       ; BL = 5 = window copy (user to system)
39264                            <1>
39265 0000D65F B800000A00        <1>       mov    eax, 0A0000h
39266 0000D664 39C7              <1>       cmp    edi, eax
39267 0000D666 0F8271EEFFFF      <1>       jb     sysret
39268 0000D66C 6683C0FF          <1>       add    ax, 0FFFFh ; 65535
39269 0000D670 39C7              <1>       cmp    edi, eax
39270 0000D672 0F8765EEFFFF      <1>       ja     sysret
39271                            <1>
39272                            <1>       ; esi = user buffer (in user's memory space)
39273 0000D678 39CA              <1>       cmp    edx, ecx ; bottom-right > top-left ?
39274 0000D67A 0F8666FEFFFF      <1>         jna    sysvideo_41 ; full screen copy
39275                            <1>
39276 0000D680 0FB7C2            <1>       movzx  eax, dx ; bottom right column
39277 0000D683 6629C8            <1>       sub    ax, cx  ; - top left column
39278 0000D686 0F8251EEFFFF      <1>         jb     sysret ; invalid
39279 0000D68C 6640              <1>       inc    ax ; same column no == 1 column
39280 0000D68E 50                <1>       push   eax ; byte count per window row
39281                            <1>
39282 0000D68F 52                <1>       push   edx
39283 0000D690 C1EB10            <1>       shr    ebx, 16 ; 320,640,800 : screen width
39284 0000D693 89C8              <1>       mov    eax, ecx
39285 0000D695 C1E810            <1>       shr    eax, 16      ; top row
39286 0000D698 F7E3              <1>       mul    ebx
39287 0000D69A 6689CA            <1>       mov    dx, cx ; top left column
39288 0000D69D 01D0              <1>       add    eax, edx
39289 0000D69F 01C7              <1>       add    edi, eax ; start address
39290 0000D6A1 59                <1>       pop    ecx ; edx
39291 0000D6A2 89C8              <1>       mov    eax, ecx
39292 0000D6A4 C1E810            <1>       shr    eax, 16 ; bottom row
39293 0000D6A7 F7E3              <1>       mul    ebx
39294 0000D6A9 6689CA            <1>       mov    dx, cx ; bottom right column
39295 0000D6AC 01D0              <1>       add    eax, edx
39296 0000D6AE 5A                <1>       pop    edx ; byte count per window row
39297 0000D6AF 0500000A00        <1>       add    eax, 0A0000h
39298 0000D6B4 3DFFFF0A00        <1>       cmp    eax, 0AFFFFh
39299 0000D6B9 0F871EEEFFFF      <1>       ja     sysret
39300 0000D6BF 57                <1>       push   edi ; start address
39301 0000D6C0 50                <1>       push   eax ; stop address (included)
39302                            <1> sysvideo_53:
39303 0000D6C1 89D1              <1>       mov    ecx, edx ; byte count
39304                            <1>       ; user to system video/display page window transfer
39305                            <1>       ; esi =      user buffer
39306 0000D6C3 E82F120000        <1>       call   transfer_from_user_buffer ; fast transfer
39307 0000D6C8 721F              <1>       jc     short sysvideo_54
39308 0000D6CA 010D[64030300]    <1>       add    [u.r0], ecx
39309 0000D6D0 01DF              <1>       add    edi, ebx ; next row
39310 0000D6D2 01CE              <1>       add    esi, ecx
39311 0000D6D4 3B3C24            <1>       cmp    edi, [esp] ; stop addr(included in loop)
39312 0000D6D7 76E8              <1>       jna    short sysvideo_53
39313 0000D6D9 5B                <1>       pop    ebx ; stop address
39314 0000D6DA 5F                <1>       pop    edi ; start address
39315 0000D6DB 29FB              <1>       sub    ebx, edi
39316 0000D6DD 43                <1>       inc    ebx
39317 0000D6DE 891D[64030300]    <1>       mov    [u.r0], ebx
39318 0000D6E4 E9F4EDFFFF        <1>       jmp    sysret
39319                            <1> sysvideo_54:
39320 0000D6E9 58                <1>       pop    eax
39321 0000D6EA 5A                <1>       pop    edx
39322 0000D6EB E9EDEDFFFF        <1>       jmp    sysret
39323                            <1>
39324                            <1> sysvideo_55:
39325 0000D6F0 80FB06            <1>       cmp    bl, 6
39326 0000D6F3 0F8793000000      <1>         ja     sysvideo_58
39327                            <1>
39328                            <1>       ; BL = 6 = window copy (system to user)
39329                            <1>
39330 0000D6F9 89F7              <1>       mov    edi, esi ; user buffer
39331                            <1>
39332 0000D6FB B800000A00        <1>       mov    eax, 0A0000h
39333 0000D700 39C6              <1>       cmp    esi, eax
39334 0000D702 0F82D5EDFFFF      <1>       jb     sysret
39335 0000D708 6683C0FF          <1>       add    ax, 0FFFFh ; 65535
39336 0000D70C 39C6              <1>       cmp    esi, eax
39337 0000D70E 0F87C9EDFFFF      <1>       ja     sysret
39338                            <1>
```

```
39339                                  <1>       ; edi = user buffer (in user's memory space)
39340 0000D714 39CA                    <1>       cmp    edx, ecx ; bottom-right > top-left ?
39341 0000D716 0F86A2FAFFFF            <1>         jna     sysvideo_17 ; full screen copy
39342                                  <1>
39343 0000D71C 0FB7C2                  <1>       movzx  eax, dx ; bottom right column
39344 0000D71F 6629C8                  <1>       sub    ax, cx  ; - top left column
39345 0000D722 0F82B5EDFFFF            <1>         jb      sysret ; invalid
39346 0000D728 6640                    <1>       inc    ax ; same column no == 1 column
39347 0000D72A 50                      <1>       push   eax ; byte count per window row
39348                                  <1>
39349 0000D72B 52                      <1>       push   edx
39350 0000D72C C1EB10                  <1>       shr    ebx, 16 ; 320, 640,800 ; screen width
39351 0000D72F 89C8                    <1>       mov    eax, ecx
39352 0000D731 C1E810                  <1>       shr    eax, 16      ; top row
39353 0000D734 F7E3                    <1>       mul    ebx
39354 0000D736 6689CA                  <1>       mov    dx, cx ; top left column
39355 0000D739 01D0                    <1>       add    eax, edx
39356 0000D73B 01C6                    <1>       add    esi, eax ; start address
39357 0000D73D 59                      <1>       pop    ecx ; edx
39358 0000D73E 89C8                    <1>       mov    eax, ecx
39359 0000D740 C1E810                  <1>       shr    eax, 16 ; bottom row
39360 0000D743 F7E3                    <1>       mul    ebx
39361 0000D745 6689CA                  <1>       mov    dx, cx ; bottom right column
39362 0000D748 01D0                    <1>       add    eax, edx
39363 0000D74A 5A                      <1>       pop    edx ; byte count per window row
39364 0000D74B 0500000A00              <1>       add    eax, 0A0000h
39365 0000D750 3DFFFF0A00              <1>       cmp    eax, 0AFFFFh
39366 0000D755 0F8782EDFFFF            <1>       ja     sysret
39367 0000D75B 56                      <1>       push   esi ; start address
39368 0000D75C 50                      <1>       push   eax ; stop address (included)
39369                                  <1> sysvideo_56:
39370 0000D75D 89D1                    <1>       mov    ecx, edx ; byte count
39371                                  <1>       ; user to system video/display page window transfer
39372                                  <1>       ; esi =      user buffer
39373 0000D75F E849110000              <1>       call   transfer_to_user_buffer ; fast transfer
39374 0000D764 721F                    <1>       jc     short sysvideo_57
39375 0000D766 010D[64030300]          <1>       add    [u.r0], ecx
39376 0000D76C 01DF                    <1>       add    edi, ebx ; next row
39377 0000D76E 01CE                    <1>       add    esi, ecx
39378 0000D770 3B3C24                  <1>       cmp    edi, [esp] ; stop addr(included in loop)
39379 0000D773 76E8                    <1>       jna    short sysvideo_56
39380 0000D775 5B                      <1>       pop    ebx ; stop address
39381 0000D776 5F                      <1>       pop    edi ; start address
39382 0000D777 29FB                    <1>       sub    ebx, edi
39383 0000D779 43                      <1>       inc    ebx
39384 0000D77A 891D[64030300]          <1>       mov    [u.r0], ebx
39385 0000D780 E958EDFFFF              <1>       jmp    sysret
39386                                  <1> sysvideo_57:
39387 0000D785 58                      <1>       pop    eax
39388 0000D786 5A                      <1>       pop    edx
39389 0000D787 E951EDFFFF              <1>       jmp    sysret
39390                                  <1>
39391                                  <1> sysvideo_58:
39392 0000D78C 80FB07                  <1>       cmp    bl, 7
39393 0000D78F 770F                    <1>       ja     short sysvideo_60
39394                                  <1>
39395                                  <1>       ; BL = 7 = AND display page bytes with CL
39396                                  <1>
39397 0000D791 BE00000A00              <1>       mov    esi, 0A0000h
39398 0000D796 B900000100              <1>       mov    ecx, 65536
39399                                  <1> sysvideo_59:
39400 0000D79B 200E                    <1>       and    byte [esi], cl
39401 0000D79D 46                      <1>       inc    esi
39402 0000D79E E2FB                    <1>       loop   sysvideo_59
39403                                  <1>
39404                                  <1> sysvideo_60:
39405 0000D7A0 80FB08                  <1>       cmp    bl, 8
39406 0000D7A3 770F                    <1>       ja     short sysvideo_62
39407                                  <1>
39408                                  <1>       ; BL = 8 = OR display page bytes with CL
39409                                  <1>
39410 0000D7A5 BE00000A00              <1>       mov    esi, 0A0000h
39411 0000D7AA B900000100              <1>       mov    ecx, 65536
39412                                  <1> sysvideo_61:
39413 0000D7AF 080E                    <1>       or     byte [esi], cl
39414 0000D7B1 46                      <1>       inc    esi
39415 0000D7B2 E2FB                    <1>       loop   sysvideo_61
39416                                  <1>
39417                                  <1> sysvideo_62:
39418 0000D7B4 80FB09                  <1>       cmp    bl, 9
39419 0000D7B7 0F8720EDFFFF            <1>         ja      sysret ; nothing to do
39420                                  <1>
39421                                  <1>       ; BL = 9 = XOR display page bytes with CL
39422                                  <1>
39423 0000D7BD BE00000A00              <1>       mov    esi, 0A0000h
39424 0000D7C2 B900000100              <1>       mov    ecx, 65536
39425                                  <1> sysvideo_63:
39426 0000D7C7 300E                    <1>       xor    byte [esi], cl
39427 0000D7C9 46                      <1>       inc    esi
39428 0000D7CA E2FB                    <1>       loop   sysvideo_63
39429                                  <1>
39430                                  <1> sysvideo_64:
39431 0000D7CC 80FF03                  <1>       cmp    bh, 3
39432 0000D7CF 7464                    <1>       je     short sysvideo_68
39433 0000D7D1 80FF04                  <1>       cmp    bh, 4
39434 0000D7D4 7721                    <1>       ja     short sysvideo_65
39435                                  <1>
39436                                  <1>       ; BH = 4
39437                                  <1>       ; Direct User Access for CGA video memory.
39438                                  <1>       ; Setup user's page tables for direct access to 0B8000h.
39439                                  <1>       ;
39440                                  <1>       ; Permission checks are not implemented yet !
39441                                  <1>       ; (11/07/2016)
```

```
39442                                   <1>
39443 0000D7D6 B800800B00               <1>       mov    eax, 0B8000h
39444 0000D7DB B908000000               <1>       mov    ecx, 8 ; 8 pages (8*4K=32K)
39445 0000D7E0 89C3                     <1>       mov    ebx, eax ; 12/05/2017 ; virtual = physical
39446 0000D7E2 E8BA7EFFFF               <1>       call   direct_memory_access
39447 0000D7E7 0F82F0ECFFFF             <1>       jc     sysret
39448                                   <1>       ; eax = 0B8000h if there is not an error
39449 0000D7ED A3[64030300]             <1>       mov    [u.r0], eax
39450 0000D7F2 E9E6ECFFFF               <1>       jmp    sysret
39451                                   <1>
39452                                   <1> sysvideo_65:
39453 0000D7F7 80FF05                   <1>       cmp    bh, 5
39454 0000D7FA 7721                     <1>       ja     short sysvideo_66
39455                                   <1>
39456                                   <1>       ; BH = 5
39457                                   <1>       ; Direct User Access for VGA video memory.
39458                                   <1>       ; Setup user's page tables for direct access to 0A0000h.
39459                                   <1>       ;
39460                                   <1>       ; Permission checks are not implemented yet !
39461                                   <1>       ; (11/07/2016)
39462                                   <1>
39463 0000D7FC B800000A00               <1>       mov    eax, 0A0000h
39464 0000D801 B910000000               <1>       mov    ecx, 16 ; 16 pages (16*4K=64K)
39465 0000D806 89C3                     <1>       mov    ebx, eax ; 12/05/2017 ; virtual = physical
39466 0000D808 E8947EFFFF               <1>       call   direct_memory_access
39467 0000D80D 0F82CAECFFFF             <1>       jc     sysret
39468                                   <1>       ; eax = 0A0000h if there is not an error
39469 0000D813 A3[64030300]             <1>       mov    [u.r0], eax
39470 0000D818 E9C0ECFFFF               <1>       jmp    sysret
39471                                   <1>
39472                                   <1> sysvideo_66:
39473 0000D81D 80FF06                   <1>       cmp    bh, 6
39474 0000D820 7705                     <1>       ja     short sysvideo_67
39475                                   <1>       ; BH = 6
39476                                   <1>       ; Direct User Access for (Super VGA) Linear Frame Buffer.
39477                                   <1>       ; Setup user's page tables for direct access to LFB.
39478                                   <1>       ;
39479                                   <1>       ; Not implemented yet !
39480                                   <1>       ; (11/07/2016)
39481 0000D822 E9B6ECFFFF               <1>       jmp    sysret
39482                                   <1>
39483                                   <1> sysvideo_67:
39484 0000D827 80FF07                   <1>       cmp    bh, 7
39485 0000D82A 0F87ADECFFFF             <1>       ja     sysret ; invalid !
39486                                   <1>
39487                                   <1>       ; BH = 7
39488                                   <1>       ; Get (Super/Extended VGA) Linear Frame Buffer info.
39489                                   <1>       ;
39490                                   <1>       ; Not implemented yet !
39491                                   <1>       ; (11/07/2016)
39492 0000D830 E9A8ECFFFF               <1>       jmp    sysret
39493                                   <1>
39494                                   <1> sysvideo_68:
39495                                   <1>       ; BH = 3
39496                                   <1>       ; Super VGA, LINEAR FRAME BUFFER data transfers
39497                                   <1>       ; Not implemented for yet ! (11/07/2016)
39498 0000D835 E9A3ECFFFF               <1>       jmp    sysret
39499                                   <1>
39500                                   <1> syslink:
39501                                   <1>       ; 13/01/2017 - TRDOS 386 (TRDOS v2.0)
39502                                   <1>       ; temporary !
39503 0000D83A B801000000               <1>       mov    eax, ERR_INV_FNUMBER ; 'invalid function number !'
39504 0000D83F A3[C8030300]             <1>         mov     [u.error], eax
39505 0000D844 A3[64030300]             <1>         mov     [u.r0], eax
39506 0000D849 E96FECFFFF               <1>       jmp    error
39507                                   <1>
39508                                   <1> isdir:
39509                                   <1>       ; 22/06/2015 (Retro UNIX 386 v1 - Beginning)
39510                                   <1>       ; 04/05/2013 - 02/08/2013 (Retro UNIX 8086 v1)
39511                                   <1>       ;
39512                                   <1>       ; 'isdir' check to see if the i-node whose i-number is in r1
39513                                   <1>       ;  is a directory. If it is, an error occurs, because 'isdir'
39514                                   <1>       ;  called by syslink and sysunlink to make sure directories
39515                                   <1>       ;  are not linked. If the user is the super user (u.uid=0),
39516                                   <1>       ; 'isdir' does not bother checking. The current i-node
39517                                   <1>       ;  is not disturbed.
39518                                   <1>       ;
39519                                   <1>       ; INPUTS ->
39520                                   <1>       ;    r1 - contains the i-number whose i-node is being checked.
39521                                   <1>       ;    u.uid - user id
39522                                   <1>       ; OUTPUTS ->
39523                                   <1>       ;    r1 - contains current i-number upon exit
39524                                   <1>       ;       (current i-node back in core)
39525                                   <1>       ;
39526                                   <1>       ; ((AX = R1))
39527                                   <1>       ;
39528                                   <1>         ; ((Modified registers: eAX, eDX, eBX, eCX, eSI, eDI, eBP))
39529                                   <1>       ;
39530                                   <1>
39531                                   <1>       ; / if the i-node whose i-number is in r1 is a directory
39532                                   <1>       ; / there is an error unless super user made the call
39533                                   <1>
39534 0000D84E 803D[B0030300]00         <1>       cmp    byte [u.uid], 0
39535                                   <1>              ; tstb u.uid / super user
39536 0000D855 762D                     <1>       jna    short isdir1
39537                                   <1>              ; beq 1f / yes, don't care
39538 0000D857 66FF35[51040300]         <1>       push   word [ii]
39539                                   <1>              ; mov ii,-(sp) / put current i-number on stack
39540 0000D85E E85C190000               <1>       call   iget
39541                                   <1>              ; jsr r0,iget / get i-node into core (i-number in r1)
39542 0000D863 66F705[00000300]00-      <1>       test   word [i.flgs], 4000h ; Bit 14 : Directory flag
39543 0000D86B 40                       <1>
39544                                   <1>              ; bit $40000,i.flgs / is it a directory
```

```
39545                                  <1>       ;jnz   error
39546                                  <1>             ; bne error9 / yes, error
39547 0000D86C 740F                    <1>       jz    short isdir0
39548 0000D86E C705[C8030300]0B00-     <1>       mov   dword [u.error], ERR_NOT_FILE  ; 11 ; ERR_DIR_ACCESS
39549 0000D876 0000                    <1>
39550                                  <1>                             ; 'permission denied !' error
39551                                  <1>       ; pop  ax
39552 0000D878 E940ECFFFF              <1>       jmp   error
39553                                  <1> isdir0:
39554 0000D87D 6658                    <1>       pop   ax
39555                                  <1>             ; mov (sp)+,r1 / no, put current i-number in r1 (ii)
39556 0000D87F E83B190000              <1>       call  iget
39557                                  <1>             ; jsr r0,iget / get it back in
39558                                  <1> isdir1: ; 1:
39559 0000D884 C3                      <1>       retn
39560                                  <1>             ; rts r0
39561                                  <1>
39562                                  <1> sysunlink:
39563                                  <1>       ; 13/01/2017 - TRDOS 386 (TRDOS v2.0)
39564                                  <1>       ; temporary !
39565 0000D885 B801000000              <1>       mov   eax, ERR_INV_FNUMBER ; 'invalid function number !'
39566 0000D88A A3[C8030300]            <1>         mov     [u.error], eax
39567 0000D88F A3[64030300]            <1>         mov     [u.r0], eax
39568 0000D894 E924ECFFFF              <1>       jmp   error
39569                                  <1> mkdir:
39570                                  <1>       ; 04/12/2015 (14 byte directory names)
39571                                  <1>       ; 12/10/2015
39572                                  <1>       ; 17/06/2015 (Retro UNIX 386 v1 - Beginning)
39573                                  <1>       ; 29/04/2013 - 01/08/2013 (Retro UNIX 8086 v1)
39574                                  <1>       ;
39575                                  <1>       ; 'mkdir' makes a directory entry from the name pointed to
39576                                  <1>       ; by u.namep into the current directory.
39577                                  <1>       ;
39578                                  <1>       ; INPUTS ->
39579                                  <1>       ;    u.namep - points to a file name
39580                                  <1>       ;                that is about to be a directory entry.
39581                                  <1>       ;    ii - current directory's i-number.
39582                                  <1>       ; OUTPUTS ->
39583                                  <1>       ;    u.dirbuf+2 - u.dirbuf+10 - contains file name.
39584                                  <1>       ;    u.off - points to entry to be filled
39585                                  <1>       ;           in the current directory
39586                                  <1>       ;    u.base - points to start of u.dirbuf.
39587                                  <1>       ;    r1 - contains i-number of current directory
39588                                  <1>       ;
39589                                  <1>       ; ((AX = R1)) output
39590                                  <1>       ;
39591                                  <1>       ;    (Retro UNIX Prototype : 11/11/2012, UNIXCOPY.ASM)
39592                                  <1>         ;    ((Modified registers: eAX, eDX, eBX, eCX, eSI, eDI, eBP))
39593                                  <1>       ;
39594                                  <1>
39595                                  <1>       ; 17/06/2015 - 32 bit modifications (Retro UNIX 386 v1)
39596 0000D899 31C0                    <1>       xor   eax, eax
39597 0000D89B BF[9A030300]            <1>       mov     edi, u.dirbuf+2
39598 0000D8A0 89FE                    <1>       mov   esi, edi
39599 0000D8A2 AB                      <1>       stosd
39600 0000D8A3 AB                      <1>       stosd
39601                                  <1>       ; 04/12/2015 (14 byte directory names)
39602 0000D8A4 AB                      <1>       stosd
39603 0000D8A5 66AB                    <1>       stosw
39604                                  <1>             ; jsr r0,copyz; u.dirbuf+2; u.dirbuf+10. / clear this
39605 0000D8A7 89F7                    <1>       mov   edi, esi ; offset to u.dirbuf
39606                                  <1>       ; 12/10/2015 ([u.namep] -> ebp)
39607                                  <1>       ;mov   ebp, [u.namep]
39608 0000D8A9 E849040000              <1>       call  trans_addr_nmbp ; convert virtual address to physical
39609                                  <1>             ; esi = physical address (page start + offset)
39610                                  <1>             ; ecx = byte count in the page (1 - 4096)
39611                                  <1>       ; edi = offset to u.dirbuf (edi is not modified in trans_addr_nm)
39612                                  <1>             ; mov u.namep,r2 / r2 points to name of directory entry
39613                                  <1>             ; mov $u.dirbuf+2,r3 / r3 points to u.dirbuf+2
39614                                  <1> mkdir_1: ; 1:
39615 0000D8AE 45                      <1>       inc   ebp ; 12/10/2015
39616                                  <1>       ;
39617                                  <1>       ; / put characters in the directory name in u.dirbuf+2 - u.dirbuf+10
39618                                  <1>        ; 01/08/2013
39619 0000D8AF AC                      <1>       lodsb
39620                                  <1>             ; movb (r2)+,r1 / move character in name to r1
39621 0000D8B0 20C0                    <1>       and   al, al
39622 0000D8B2 7427                    <1>       jz    short mkdir_3
39623                                  <1>             ; beq 1f / if null, done
39624 0000D8B4 3C2F                    <1>       cmp   al, '/'
39625                                  <1>             ; cmp r1,$'/ / is it a "/"?
39626 0000D8B6 7414                    <1>       je    short mkdir_err
39627                                  <1>       ;je   error
39628                                  <1>             ; beq error9 / yes, error
39629                                  <1>       ; 12/10/2015
39630 0000D8B8 6649                    <1>       dec   cx
39631 0000D8BA 7505                    <1>       jnz   short mkdir_2
39632                                  <1>       ; 12/10/2015 ([u.namep] -> ebp)
39633 0000D8BC E83C040000              <1>       call  trans_addr_nm ; convert virtual address to physical
39634                                  <1>             ; esi = physical address (page start + offset)
39635                                  <1>             ; ecx = byte count in the page
39636                                  <1>       ; edi = offset to u.dirbuf (edi is not modified in trans_addr_nm)
39637                                  <1> mkdir_2:
39638 0000D8C1 81FF[A8030300]          <1>       cmp     edi, u.dirbuf+16 ; ; 04/12/2015 (10 -> 16)
39639                                  <1>             ; cmp r3,$u.dirbuf+10. / have we reached the last slot for
39640                                  <1>                                    ; / a char?
39641 0000D8C7 74E5                    <1>       je    short mkdir_1
39642                                  <1>             ; beq 1b / yes, go back
39643 0000D8C9 AA                      <1>       stosb
39644                                  <1>             ; movb r1,(r3)+ / no, put the char in the u.dirbuf
39645 0000D8CA EBE2                    <1>       jmp   short mkdir_1
39646                                  <1>             ; br 1b / get next char
39647                                  <1> mkdir_err:
```

```
39648                                  <1>       ; 17/06/2015
39649 0000D8CC C705[C8030300]1300-      <1>       mov    dword [u.error], ERR_NOT_DIR ; 'not a valid directory !!'
39650 0000D8D4 0000                    <1>
39651 0000D8D6 E9E2EBFFFF              <1>       jmp    error
39652                                  <1>
39653                                  <1> mkdir_3: ; 1:
39654 0000D8DB A1[78030300]            <1>       mov    eax, [u.dirp]
39655 0000D8E0 A3[80030300]            <1>       mov    [u.off], eax
39656                                  <1>              ; mov u.dirp,u.off / pointer to empty current directory
39657                                  <1>                                 ; / slot to u.off
39658                                  <1> wdir: ; 29/04/2013
39659 0000D8E5 C705[84030300]-         <1>         mov     dword [u.base], u.dirbuf
39660 0000D8EB [98030300]              <1>
39661                                  <1>              ; mov $u.dirbuf,u.base / u.base points to created file name
39662 0000D8EF C705[88030300]1000-      <1>         mov     dword [u.count], 16 ; 04/12/2015 (10 -> 16)
39663 0000D8F7 0000                    <1>
39664                                  <1>              ; mov $10.,u.count / u.count = 10
39665 0000D8F9 66A1[51040300]          <1>       mov    ax, [ii]
39666                                  <1>              ; mov ii,r1 / r1 has i-number of current directory
39667 0000D8FF B201                    <1>       mov    dl, 1 ; owner flag mask ; RETRO UNIX 8086 v1 modification !
39668 0000D901 E8BF180000              <1>       call   access
39669                                  <1>              ; jsr r0,access; 1 / get i-node and set its file up
39670                                  <1>                                 ; / for writing
39671                                  <1>       ; AX = i-number of current directory
39672                                  <1>       ; 01/08/2013
39673 0000D906 FE05[C6030300]          <1>       inc     byte [u.kcall] ; the caller is 'mkdir' sign
39674 0000D90C E8FD110000              <1>       call   writei
39675                                  <1>              ; jsr r0,writei / write into directory
39676 0000D911 C3                      <1>       retn
39677                                  <1>              ; rts r0
39678                                  <1>
39679                                  <1> sysexec:
39680                                  <1>       ; 14/11/2017
39681                                  <1>       ; 13/11/2017
39682                                  <1>       ; 04/01/2017
39683                                  <1>       ; 24/10/2016
39684                                  <1>       ; 24/04/2016 - TRDOS 386 (TRDOS v2.0)
39685                                  <1>       ; 23/06/2015 - 23/10/2015 (Retro UNIX 386 v1)
39686                                  <1>       ; 03/06/2013 - 06/12/2013 (Retro UNIX 8086 v1)
39687                                  <1>       ;
39688                                  <1>       ; 'sysexec' initiates execution of a file whose path name if
39689                                  <1>       ; pointed to by 'name' in the sysexec call.
39690                                  <1>       ; 'sysexec' performs the following operations:
39691                                  <1>       ;    1. obtains i-number of file to be executed via 'namei'.
39692                                  <1>       ;    2. obtains i-node of file to be exceuted via 'iget'.
39693                                  <1>       ;    3. sets trap vectors to system routines.
39694                                  <1>       ;    4. loads arguments to be passed to executing file into
39695                                  <1>       ;      highest locations of user's core
39696                                  <1>       ;    5. puts pointers to arguments in locations immediately
39697                                  <1>       ;      following arguments.
39698                                  <1>       ;    6.      saves number of arguments in next location.
39699                                  <1>       ;    7. intializes user's stack area so that all registers
39700                                  <1>       ;      will be zeroed and the PS is cleared and the PC set
39701                                  <1>       ;      to core when 'sysret' restores registers
39702                                  <1>       ;      and does an rti.
39703                                  <1>       ;    8. inializes u.r0 and u.sp
39704                                  <1>       ;    9. zeros user's core down to u.r0
39705                                  <1>       ;   10.      reads executable file from storage device into core
39706                                  <1>       ;      starting at location 'core'.
39707                                  <1>       ;   11.      sets u.break to point to end of user's code with
39708                                  <1>       ;      data area appended.
39709                                  <1>       ;   12.      calls 'sysret' which returns control at location
39710                                  <1>       ;      'core' via 'rti' instruction.
39711                                  <1>       ;
39712                                  <1>       ; Calling sequence:
39713                                  <1>       ;      sysexec; namep; argp
39714                                  <1>       ; Arguments:
39715                                  <1>       ;      namep - points to pathname of file to be executed
39716                                  <1>       ;      argp  - address of table of argument pointers
39717                                  <1>       ;      argp1... argpn - table of argument pointers
39718                                  <1>       ;      argp1:<...0> ... argpn:<...0> - argument strings
39719                                  <1>       ; Inputs: (arguments)
39720                                  <1>       ; Outputs: -
39721                                  <1>       ; ..........................................................
39722                                  <1>       ;
39723                                  <1>       ; Retro UNIX 386 v1 modification:
39724                                  <1>       ;      User application runs in it's own virtual space
39725                                  <1>       ;      which is izolated from kernel memory (and other
39726                                  <1>       ;      memory pages) via 80386   paging in ring 3
39727                                  <1>       ;      privilige mode. Virtual start address is always 0.
39728                                  <1>       ;      User's core memory starts at linear address 400000h
39729                                  <1>       ;      (the end of the 1st 4MB).
39730                                  <1>       ;
39731                                  <1>       ; Retro UNIX 8086 v1 modification:
39732                                  <1>       ;      user/application segment and system/kernel segment
39733                                  <1>       ;      are different and sysenter/sysret/sysrele routines
39734                                  <1>       ;      are different (user's registers are saved to
39735                                  <1>       ;      and then restored from system's stack.)
39736                                  <1>       ;
39737                                  <1>       ;      NOTE: Retro UNIX 8086 v1 'arg2' routine gets these
39738                                  <1>       ;            arguments which were in these registers;
39739                                  <1>       ;            but, it returns by putting the 1st argument
39740                                  <1>       ;            in 'u.namep' and the 2nd argument
39741                                  <1>       ;            on top of stack. (1st argument is offset of the
39742                                  <1>       ;            file/path name in the user's program segment.)
39743                                  <1>
39744                                  <1>       ;call  arg2
39745                                  <1>       ; * name - 'u.namep' points to address of file/path name
39746                                  <1>       ;          in the user's program segment ('u.segmnt')
39747                                  <1>       ;          with offset in BX register (as sysopen argument 1).
39748                                  <1>       ; * argp - sysexec argument 2 is in CX register
39749                                  <1>       ;          which is on top of stack.
39750                                  <1>       ;
```

```
39751                                  <1>             ; jsr r0,arg2 / arg0 in u.namep,arg1 on top of stack
39752                                  <1>
39753                                  <1>       ; 23/06/2015 (32 bit modifications)
39754                                  <1>
39755                                  <1>       ;; 13/11/2017
39756                                  <1>       ;;mov  [u.namep], ebx ; argument 1
39757                                  <1>         ; 18/10/2015
39758 0000D912 890D[4C040300]          <1>       mov     [argv], ecx  ; * ; argument 2
39759                                  <1>
39760                                  <1>       ; 13/11/2017
39761 0000D918 89DE                    <1>       mov    esi, ebx
39762 0000D91A E8DC1C0000              <1>       call   set_working_path_x
39763 0000D91F 7319                    <1>       jnc    short sysexec_0
39764                                  <1>
39765                                  <1>       ;; 'bad command or file name'
39766                                  <1>       ;mov   eax, ERR_BAD_CMD_ARG ; 01h ; TRDOS 8086
39767                                  <1>
39768                                  <1>       ; 'file not found !' error
39769 0000D921 B802000000              <1>       mov    eax, ERR_NOT_FOUND ; 02h ; TRDOS 8086
39770                                  <1> sysexec_not_found_err:
39771                                  <1> sysexec_access_error:
39772                                  <1> sysexec_ext_error:
39773 0000D926 A3[64030300]            <1>       mov    [u.r0], eax
39774 0000D92B A3[C8030300]            <1>       mov    [u.error], eax
39775 0000D930 E89B1D0000              <1>       call   reset_working_path
39776 0000D935 E983EBFFFF              <1>       jmp    error
39777                                  <1>
39778                                  <1> sysexec_0:
39779                                  <1>       ; 13/11/2017
39780                                  <1>       ;mov   esi, FindFile_Name
39781 0000D93A 66B80018                <1>         mov  ax, 1800h ; Only files
39782 0000D93E E8C8A6FFFF              <1>       call   find_first_file
39783 0000D943 72E1                    <1>       jc     short sysexec_not_found_err ; eax = 2
39784                                  <1>
39785                                  <1>       ; check_ file attributes
39786                                  <1>       ; (attribute bits = 00ADVSHR) ; 18h = Directory+Volume
39787                                  <1>       ; BL = Attributes byte
39788                                  <1>
39789 0000D945 F6C306                  <1>         test bl, 6  ; system file or hidden file (S+H)
39790                                  <1>       ;jz    short sysexec_0ext
39791 0000D948 7417                    <1>       jz     short sysexec_1 ; yes
39792                                  <1>
39793                                  <1>       ; 13/11/2017
39794                                  <1>       ; /// TRDOS386 permission check for multiuser mode ///
39795                                  <1>       ; SYSTEM file or HIDDEN file !!
39796                                  <1>       ; (Only super user has permission to run this file.)
39797                                  <1>
39798                                  <1>       ; ([u.uid]=0 for super user or root in multiuser mode)
39799                                  <1>       ; ([u.uid]=0 for any users in singleuser mode)
39800 0000D94A 803D[B0030300]00         <1>       cmp    byte [u.uid], 0 ; Super User ([u.uid]=0) ?
39801                                  <1>       ;jna   short sysexec_0ext
39802 0000D951 760E                    <1>       jna    short sysexec_1 ; yes
39803                                  <1>
39804                                  <1>       ; 'permission denied !' error
39805 0000D953 B80B000000              <1>         mov  eax, ERR_FILE_ACCESS  ; 11 = ERR_PERM_DENIED
39806 0000D958 EBCC                    <1>         jmp  short sysexec_access_error
39807                                  <1>
39808                                  <1> sysexec_not_exf:
39809                                  <1>       ; 'not executable file !' error
39810 0000D95A B816000000              <1>       mov    eax, ERR_NOT_EXECUTABLE
39811 0000D95F EBC5                    <1>       jmp    sysexec_ext_error
39812                                  <1>
39813                                  <1> ;sysexec_0ext:
39814                                  <1> sysexec_1:
39815                                  <1>       ; 13/11/2017
39816                                  <1>       ; check program file name extension
39817                                  <1>       ; ('.PRG' for current TRDOS version)
39818 0000D961 E89DC1FFFF              <1>       call   check_prg_filename_ext
39819 0000D966 72F2                    <1>       jc     short sysexec_not_exf
39820                                  <1>
39821                                  <1>       ; '.PRG' extension is OK.
39822                                  <1>       ; Only '.PRG' files are valid program files
39823                                  <1>       ; for current TRDOS 386 version.
39824                                  <1>
39825 0000D968 8B15[F85C0100]          <1>       mov    edx, [FindFile_DirEntry+DirEntry_FileSize]
39826 0000D96E 66A1[F05C0100]          <1>       mov    ax, [FindFile_DirEntry+DirEntry_FstClusHI]
39827 0000D974 C1E010                  <1>       shl    eax, 16
39828 0000D977 66A1[F65C0100]          <1>       mov    ax, [FindFile_DirEntry+DirEntry_FstClusLO]
39829                                  <1>       ; EAX = First Cluster number
39830                                  <1>       ; EDX = File Size
39831                                  <1>
39832 0000D97D A3[51040300]            <1>       mov    [ii], eax
39833 0000D982 8915[55040300]          <1>       mov    [i.size], edx
39834                                  <1>
39835                                  <1> ;sysexec_1:
39836                                  <1>       ; 13/11/2017 - TRDOS 386 (TRDOS v2.0)
39837                                  <1>       ; 24/06/2015 - 23/10/2015 (Retro UNIX 386 v1)
39838                                  <1>         ; Moving arguments to the end of [u.upage]
39839                                  <1>       ; (by regarding page borders in user's memory space)
39840                                  <1>       ;
39841                                  <1>       ; 10/10/2015
39842                                  <1>       ; 21/07/2015
39843 0000D988 89E5                    <1>       mov    ebp, esp ; (**)
39844                                  <1>       ; 18/10/2015
39845 0000D98A 89EF                    <1>       mov    edi, ebp
39846 0000D98C B900010000              <1>       mov    ecx, MAX_ARG_LEN ; 256
39847                                  <1>       ;sub   edi, MAX_ARG_LEN ; 256
39848 0000D991 29CF                    <1>       sub    edi, ecx
39849 0000D993 89FC                    <1>       mov    esp, edi ; *!*
39850 0000D995 31C0                    <1>       xor    eax, eax
39851 0000D997 A3[8C030300]            <1>       mov    [u.nread], eax ; 0
39852 0000D99C 66A3[4A040300]          <1>       mov    [argc], ax ; 0 ; 13/11/2017
39853 0000D9A2 49                      <1>       dec    ecx ; 256 - 1
```

```
39854 0000D9A3 890D[88030300]      <1>       mov   [u.count], ecx ; MAX_ARG_LEN - 1 ; 255
39855                              <1>       ;mov  dword [u.count], MAX_ARG_LEN - 1 ; 255
39856                              <1> sysexec_2:
39857 0000D9A9 8B35[4C040300]      <1>       mov   esi, [argv] ; 18/10/2015
39858 0000D9AF E866000000          <1>       call  get_argp
39859 0000D9B4 B904000000          <1>       mov   ecx, 4 ; mov ecx, 4
39860                              <1> sysexec_3:
39861 0000D9B9 21C0                <1>       and   eax, eax
39862 0000D9BB 0F84ED070000        <1>         jz      sysexec_6
39863                              <1>       ; 18/10/2015
39864 0000D9C1 010D[4C040300]      <1>       add   [argv], ecx ; 4
39865 0000D9C7 66FF05[4A040300]    <1>       inc   word [argc]
39866                              <1>       ;
39867 0000D9CE A3[84030300]        <1>       mov   [u.base], eax
39868                              <1>       ; 23/10/2015
39869 0000D9D3 66C705[C4030300]00- <1>       mov   word [u.pcount], 0
39870 0000D9DB 00                  <1>
39871                              <1> sysexec_4:
39872 0000D9DC E86C0E0000          <1>       call  cpass ; get a character from user's core memory
39873 0000D9E1 750E                <1>         jnz short sysexec_5
39874                              <1>             ; (max. 255 chars + null)
39875                              <1>       ; 18/10/2015
39876 0000D9E3 28C0                <1>       sub   al, al
39877 0000D9E5 AA                  <1>       stosb
39878 0000D9E6 FF05[8C030300]      <1>       inc   dword [u.nread]
39879 0000D9EC E9BD070000          <1>       jmp   sysexec_6 ; 24/04/2016
39880                              <1> sysexec_5:
39881 0000D9F1 AA                  <1>       stosb
39882 0000D9F2 20C0                <1>       and   al, al
39883 0000D9F4 75E6                <1>       jnz   short sysexec_4
39884 0000D9F6 B904000000          <1>       mov   ecx, 4
39885 0000D9FB 390D[48040300]      <1>       cmp   [ncount], ecx ; 4
39886 0000DA01 72A6                <1>       jb    short sysexec_2
39887 0000DA03 8B35[44040300]      <1>       mov   esi, [nbase]
39888 0000DA09 010D[44040300]      <1>       add   [nbase], ecx ; 4
39889 0000DA0F 66290D[48040300]    <1>       sub   [ncount], cx
39890 0000DA16 8B06                <1>       mov   eax, [esi]
39891 0000DA18 EB9F                <1>       jmp   short sysexec_3
39892                              <1>
39893                              <1> get_argp:
39894                              <1>       ; 14/11/2017 - TRDOS 386 (TRDOS v2.0)
39895                              <1>       ; 18/10/2015 (nbase, ncount)
39896                              <1>       ; 21/07/2015
39897                              <1>       ; 24/06/2015 (Retro UNIX 386 v1)
39898                              <1>       ; Get (virtual) address of argument from user's core memory
39899                              <1>       ;
39900                              <1>       ; INPUT:
39901                              <1>       ;     esi = virtual address of argument pointer
39902                              <1>       ; OUTPUT:
39903                              <1>       ;     eax = virtual address of argument
39904                              <1>       ;
39905                              <1>       ; Modified registers: EAX, EBX, ECX, EDX, ESI
39906                              <1>       ;
39907 0000DA1A 833D[BC030300]00    <1>       cmp     dword [u.ppgdir], 0 ; /etc/init ?
39908                              <1>                                   ; (the caller is kernel)
39909 0000DA21 7667                <1>         jna     short get_argpk
39910                              <1>       ;
39911 0000DA23 89F3                <1>       mov   ebx, esi
39912 0000DA25 E86578FFFF          <1>       call  get_physical_addr ; get physical address
39913 0000DA2A 0F8289000000        <1>         jc      get_argp_err
39914 0000DA30 A3[44040300]        <1>       mov   [nbase], eax ; physical address
39915 0000DA35 66890D[48040300]    <1>       mov   [ncount], cx ; remain byte count in page (1-4096)
39916 0000DA3C B804000000          <1>       mov   eax, 4 ; 21/07/2015
39917 0000DA41 6639C1              <1>       cmp   cx, ax ; 4
39918 0000DA44 735D                <1>       jnb   short get_argp2
39919 0000DA46 89F3                <1>       mov   ebx, esi
39920 0000DA48 01CB                <1>       add   ebx, ecx
39921 0000DA4A E84078FFFF          <1>       call  get_physical_addr ; get physical address
39922 0000DA4F 7268                <1>       jc    short get_argp_err
39923                              <1>       ;push  esi
39924 0000DA51 89C6                <1>       mov   esi, eax
39925 0000DA53 66870D[48040300]    <1>       xchg  cx, [ncount]
39926 0000DA5A 8735[44040300]      <1>       xchg  esi, [nbase]
39927 0000DA60 B504                <1>       mov   ch, 4
39928 0000DA62 28CD                <1>       sub   ch, cl
39929                              <1> get_argp0:
39930 0000DA64 AC                  <1>       lodsb
39931 0000DA65 6650                <1>       push  ax
39932 0000DA67 FEC9                <1>       dec   cl
39933 0000DA69 75F9                <1>         jnz     short get_argp0
39934 0000DA6B 8B35[44040300]      <1>       mov   esi, [nbase]
39935                              <1>       ; 21/07/2015
39936 0000DA71 0FB6C5              <1>       movzx eax, ch
39937 0000DA74 0105[44040300]      <1>       add   [nbase], eax
39938 0000DA7A 662905[48040300]    <1>       sub   [ncount], ax
39939                              <1> get_argp1:
39940 0000DA81 AC                  <1>       lodsb
39941 0000DA82 FECD                <1>       dec   ch
39942 0000DA84 7447                <1>         jz      short get_argp3
39943 0000DA86 6650                <1>         push ax
39944 0000DA88 EBF7                <1>       jmp     short get_argp1
39945                              <1> get_argpk:
39946                              <1>       ; Argument is in kernel's memory space
39947 0000DA8A 66C705[48040300]00- <1>       mov   word [ncount], PAGE_SIZE ; 4096
39948 0000DA92 10                  <1>
39949 0000DA93 8935[44040300]      <1>       mov   [nbase], esi
39950 0000DA99 8305[44040300]04    <1>       add   dword [nbase], 4
39951 0000DAA0 8B06                <1>       mov   eax, [esi] ; virtual addr. = physcal addr.
39952 0000DAA2 C3                  <1>       retn
39953                              <1> get_argp2:
39954                              <1>       ; 21/07/2015
39955                              <1>       ;mov  eax, 4
39956 0000DAA3 8B15[44040300]      <1>       mov   edx, [nbase] ; 18/10/2015
```

```
39957 0000DAA9 0105[44040300]      <1>       add    [nbase], eax
39958 0000DAAF 662905[48040300]    <1>       sub    [ncount], ax
39959                              <1>       ;
39960 0000DAB6 8B02                <1>       mov    eax, [edx]
39961 0000DAB8 C3                  <1>       retn
39962                              <1> get_argp_err:
39963 0000DAB9 A3[C8030300]        <1>       mov    [u.error], eax
39964                              <1>       ; 14/11/2017
39965 0000DABE B801000000          <1>       mov    eax, ERR_BAD_CMD_ARG ; 01h ; TRDOS 8086
39966 0000DAC3 A3[64030300]        <1>       mov    [u.r0], eax
39967 0000DAC8 E9F0E9FFFF          <1>       jmp    error
39968                              <1> get_argp3:
39969 0000DACD B103                <1>       mov    cl, 3
39970                              <1> get_argp4:
39971 0000DACF C1E008              <1>       shl    eax, 8
39972 0000DAD2 665A                <1>       pop    dx
39973 0000DAD4 88D0                <1>       mov    al, dl
39974 0000DAD6 E2F7                <1>         loop    get_argp4
39975                              <1>       ;pop   esi
39976 0000DAD8 C3                  <1>       retn
39977                              <1>
39978                              <1> sysstat:
39979                              <1>       ; 13/01/2017 - TRDOS 386 (TRDOS v2.0)
39980                              <1>       ; temporary !
39981 0000DAD9 B801000000          <1>       mov    eax, ERR_INV_FNUMBER ; 'invalid function number !'
39982 0000DADE A3[C8030300]        <1>         mov     [u.error], eax
39983 0000DAE3 A3[64030300]        <1>         mov     [u.r0], eax
39984 0000DAE8 E9D0E9FFFF          <1>       jmp    error
39985                              <1>
39986                              <1> sysfstat:
39987                              <1>       ; 13/01/2017 - TRDOS 386 (TRDOS v2.0)
39988                              <1>       ; temporary !
39989 0000DAED B801000000          <1>       mov    eax, ERR_INV_FNUMBER ; 'invalid function number !'
39990 0000DAF2 A3[C8030300]        <1>         mov     [u.error], eax
39991 0000DAF7 A3[64030300]        <1>         mov     [u.r0], eax
39992 0000DAFC E9BCE9FFFF          <1>       jmp    error
39993                              <1>
39994                              <1> fclose:
39995                              <1>       ; 06/10/2016 (TRDOS 386 = TRDOS v2.0)
39996                              <1>       ;
39997                              <1>       ; 18/06/2015 (Retro UNIX 386 v1 - Beginning)
39998                              <1>       ;            (32 bit offset pointer modification)
39999                              <1>       ; 19/04/2013 - 12/01/2014 (Retro UNIX 8086 v1)
40000                              <1>       ;
40001                              <1>       ; Given the file descriptor (index to the u.fp list)
40002                              <1>       ; 'fclose' first gets the i-number of the file via 'getf'.
40003                              <1>       ; If i-node is active (i-number > 0) the entry in
40004                              <1>       ; u.fp list is cleared. If all the processes that opened
40005                              <1>       ; that file close it, then fsp etry is freed and the file
40006                              <1>       ; is closed. If not a return is taken.
40007                              <1>       ; If the file has been deleted while open, 'anyi' is called
40008                              <1>       ; to see anyone else has it open, i.e., see if it is appears
40009                              <1>       ; in another entry in the fsp table. Upon return from 'anyi'
40010                              <1>       ; a check is made to see if the file is special.
40011                              <1>       ;
40012                              <1>       ; INPUTS ->
40013                              <1>       ;    r1 - contains the file descriptor (value=0,1,2...)
40014                              <1>       ;    u.fp - list of entries in the fsp table
40015                              <1>       ;    fsp - table of entries (4 words/entry) of open files.
40016                              <1>       ; OUTPUTS ->
40017                              <1>       ;    r1 - contains the same file descriptor
40018                              <1>       ;    r2 - contains i-number
40019                              <1>       ;
40020                              <1>       ; ((AX = R1))
40021                              <1>       ; ((Modified registers: eDX, eBX, eCX, eSI, eDI, eBP))
40022                              <1>       ;
40023                              <1>       ; Retro UNIX 8086 v1 modification : CF = 1
40024                              <1>       ;             if i-number of the file is 0. (error)
40025                              <1>       ;
40026                              <1>       ; TRDOS 386 (06/10/2016)
40027                              <1>       ;
40028                              <1>       ; INPUT:
40029                              <1>       ;     EAX = File Handle (File Descriptor, File Index)
40030                              <1>       ;
40031                              <1>       ; OUTPUT:
40032                              <1>       ;     CF = 1 -> File not open !
40033                              <1>       ;     CF = 0 -> OK!
40034                              <1>       ;          EBX = File Number (System)
40035                              <1>       ;          [cdev] = Logical DOS Drive Number
40036                              <1>       ;          EAX = File Handle/Number (user)
40037                              <1>       ;
40038                              <1>       ; Modified Registers: EBX
40039                              <1>
40040 0000DB01 50                  <1>       push   eax ; File handle
40041                              <1>
40042 0000DB02 E846000000          <1>       call   getf
40043 0000DB07 0F829E1F0000        <1>       jc     device_close ; eax = device number
40044                              <1>
40045 0000DB0D 80BB[4A630100]01    <1>       cmp    byte [ebx+OF_MODE], 1 ; open mode ; 0 = empty entry
40046 0000DB14 722E                <1>       jb     short fclose_1           ; 1 = read, 2 = write
40047                              <1>
40048 0000DB16 83F801              <1>       cmp    eax, 1 ; is the first cluster number > 0
40049 0000DB19 7229                <1>       jb     short fclose_1 ; no, this is empty entry
40050                              <1>
40051                              <1> fclose_0:
40052 0000DB1B FE8B[5E630100]      <1>       dec    byte [ebx+OF_OPENCOUNT] ; decrement the number of processes
40053                              <1>                                      ; that have opened the file
40054 0000DB21 7921                <1>       jns    short fclose_1 ; jump if not negative (jump if bit 7 is 0)
40055                              <1>                     ; if all processes haven't closed the file, return
40056                              <1>       ;
40057                              <1>       ; eax ; First cluster
40058 0000DB23 31C0                <1>       xor    eax, eax ; 0
40059 0000DB25 8883[4A630100]      <1>       mov    [ebx+OF_MODE], al ; 0 = empty entry
```

```
40060                                  <1>       ;mov   [ebx+OF_STATUS], al ; 0 = empty entry
40061 0000DB2B 66C1E302                <1>       shl    bx, 2
40062 0000DB2F 8983[18630100]          <1>       mov    [ebx+OF_FCLUSTER], eax ; 0
40063 0000DB35 8983[30640100]          <1>       mov    [ebx+OF_CCLUSTER], eax ; 0
40064                                  <1>       ;mov   [ebx+OF_CCINDEX], eax ; 0
40065 0000DB3B A3[74030300]            <1>       mov    [u.fofp], eax ; 0
40066 0000DB40 66C1EB02                <1>       shr    bx, 2
40067                                  <1> fclose_1: ; 1:
40068 0000DB44 58                      <1>       pop    eax ; File handle (File Descriptor, File Index)
40069 0000DB45 C680[6A030300]00        <1>       mov    byte [eax+u.fp], 0 ; clear that entry in the u.fp list
40070 0000DB4C C3                      <1>       retn
40071                                  <1>
40072                                  <1> getf:
40073                                  <1>       ; 12/10/2016
40074                                  <1>       ; 11/10/2016
40075                                  <1>       ; 08/10/2016
40076                                  <1>       ; 06/10/2016 (TRDOS 386 = TRDOS v2.0)
40077                                  <1>       ; / get the device number and the i-number of an open file
40078                                  <1>       ; 13/05/2015
40079                                  <1>       ; 11/05/2015 (Retro UNIX 386 v1 - Beginning)
40080                                  <1>       ; 19/04/2013 - 18/11/2013 (Retro UNIX 8086 v1)
40081                                  <1>       ;
40082 0000DB4D 89C3                    <1>       mov    ebx, eax
40083                                  <1> getf1:
40084 0000DB4F 83FB0A                  <1>       cmp    ebx, 10
40085 0000DB52 730A                    <1>         jnb  short getf2
40086 0000DB54 8A9B[6A030300]          <1>       mov    bl, [ebx+u.fp]
40087 0000DB5A 08DB                    <1>       or     bl, bl
40088 0000DB5C 7503                    <1>       jnz    short getf3
40089                                  <1> getf2:
40090                                  <1>       ; 'File not open !' error (ax=0)
40091 0000DB5E 29C0                    <1>       sub    eax, eax
40092 0000DB60 C3                      <1>       retn
40093                                  <1> getf3:
40094 0000DB61 F6C380                  <1>       test   bl, 80h
40095 0000DB64 7530                    <1>       jnz    short getf5 ; device
40096 0000DB66 FECB                    <1>       dec    bl ; 0 based
40097 0000DB68 8A83[40630100]          <1>       mov    al, [ebx+OF_DRIVE]
40098 0000DB6E A2[46030300]            <1>       mov    [cdev], al
40099 0000DB73 C0E302                  <1>       shl    bl, 2 ; *4 (dword offset)
40100 0000DB76 8B83[90630100]          <1>       mov    eax, [ebx+OF_SIZE]
40101 0000DB7C A3[55040300]            <1>       mov    [i.size], eax ; file size
40102 0000DB81 8D83[68630100]          <1>       lea    eax, [ebx+OF_POINTER] ;12/10/2016
40103 0000DB87 A3[74030300]            <1>       mov    [u.fofp], eax
40104 0000DB8C 8B83[18630100]          <1>       mov    eax, [ebx+OF_FCLUSTER]
40105 0000DB92 C0EB02                  <1>       shr    bl, 2 ; /4 (byte offset)
40106                                  <1> getf4:
40107 0000DB95 C3                      <1>       retn
40108                                  <1> getf5:
40109                                  <1>       ; get device number
40110 0000DB96 80E37F                  <1>       and    bl, 7Fh ; 1 to 7Fh
40111 0000DB99 FECB                    <1>       dec    bl ; 0 based (0 to 7Eh)
40112 0000DB9B 8A83[72610100]          <1>       mov    al, [ebx+DEV_DRIVER]
40113 0000DBA1 8AAB[DC600100]          <1>       mov    ch, [ebx+DEV_ACCESS]
40114 0000DBA7 8A8B[90610100]          <1>       mov    cl, [ebx+DEV_OPENMODE]
40115 0000DBAD 80E5FE                  <1>       and    ch, 0FEh ; reset bit 0 ; dev_close
40116 0000DBB0 F9                      <1>       stc ; cf = 1
40117 0000DBB1 C3                      <1>       retn
40118                                  <1>
40119                                  <1> namei:
40120                                  <1>       ; 13/11/2017 (TRDOS 386 = TRDOS v2.0)
40121                                  <1>       ; 04/12/2015 (14 byte file names)
40122                                  <1>       ; 18/10/2015 (nbase, ncount)
40123                                  <1>       ; 21/08/2015, 12/10/2015
40124                                  <1>       ; 17/06/2015, 02/07/2015, 18/07/2015
40125                                  <1>       ; 16/06/2015 (Retro UNIX 386 v1 - Beginning)
40126                                  <1>       ; 24/04/2013 - 31/07/2013 (Retro UNIX 8086 v1)
40127                                  <1>       ;
40128                                  <1>       ; 'namei' takes a file path name and returns i-number of
40129                                  <1>       ; the file in the current directory or the root directory
40130                                  <1>       ; (if the first character of the pathname is '/').
40131                                  <1>       ;
40132                                  <1>       ; INPUTS ->
40133                                  <1>       ;    u.namep - points to a file path name
40134                                  <1>       ;    u.cdir - i-number of users directory
40135                                  <1>       ;    u.cdev - device number on which user directory resides
40136                                  <1>       ; OUTPUTS ->
40137                                  <1>       ;    r1 - i-number of file
40138                                  <1>       ;    cdev
40139                                  <1>       ;    u.dirbuf - points to directory entry where a match
40140                                  <1>       ;               occurs in the search for file path name.
40141                                  <1>       ;              If no match u.dirb points to the end of
40142                                  <1>       ;               the directory and r1 = i-number of the current
40143                                  <1>       ;              directory.
40144                                  <1>       ; ((AX = R1))
40145                                  <1>       ;
40146                                  <1>       ; (Retro UNIX Prototype : 07/10/2012 - 05/01/2013, UNIXCOPY.ASM)
40147                                  <1>         ; ((Modified registers: eDX, eBX, eCX, eSI, eDI, eBP))
40148                                  <1>       ;
40149                                  <1>
40150 0000DBB2 66A1[68030300]          <1>       mov    ax, [u.cdir]
40151                                  <1>              ; mov u.cdir,r1 / put the i-number of current directory
40152                                  <1>                           ; / in r1
40153 0000DBB8 668B15[AE030300]        <1>       mov    dx, [u.cdrv]
40154 0000DBBF 668915[46030300]        <1>       mov    [cdev], dx       ; NOTE: Retro UNIX 8086 v1
40155                                  <1>                               ; device/drive number is in 1 byte,
40156                                  <1>                               ; not in 1 word!
40157                                  <1>              ; mov u.cdev,cdev / device number for users directory
40158                                  <1>                             ; / into cdev
40159                                  <1>       ; 12/10/2015
40160                                  <1>       ; 16/06/2015 - 32 bit modifications (Retro UNIX 386 v1)
40161                                  <1>              ; convert virtual (pathname) addr to physical address
40162 0000DBC6 E82C010000              <1>       call    trans_addr_nmbp ; 12/10/2015
```

```
                                <1>             ; esi = physical address of [u.namep]
                                <1>             ; ecx = byte count in the page
0000DBCB 803E2F                 <1>       cmp   byte [esi], '/'
                                <1>             ; cmpb *u.namep,$'/ / is first char in file name a /
0000DBCE 751E                   <1>       jne   short namei_1
                                <1>             ; bne 1f
0000DBD0 FF05[7C030300]         <1>       inc   dword [u.namep]
                                <1>             ; inc u.namep / go to next char
0000DBD6 6649                   <1>       dec   cx ; remain byte count in the page
0000DBD8 7506                   <1>       jnz   short namei_0
                                <1>       ; 12/10/2015
0000DBDA E818010000             <1>       call  trans_addr_nmbp ; convert virtual address to physical
                                <1>             ; esi = physical address (page start + offset)
                                <1>             ; ecx = byte count in the page
0000DBDF 4E                     <1>       dec   esi
                                <1> namei_0:
0000DBE0 46                     <1>       inc   esi  ; go to next char
0000DBE1 66A1[50030300]         <1>       mov   ax, [rootdir] ; 09/07/2013
                                <1>             ; mov rootdir,r1 / put i-number of rootdirectory in r1
0000DBE7 C605[46030300]00       <1>       mov   byte [cdev], 0
                                <1>             ; clr cdev / clear device number
                                <1> namei_1: ; 1:
0000DBEE F606FF                 <1>       test  byte [esi], 0FFh
0000DBF1 74A2                   <1>       jz    short getf4
                                <1>       ;jz     nig
                                <1>             ; tstb *u.namep / is the character in file name a nul
                                <1>             ; beq nig / yes, end of file name reached;
                                <1>                   ; / branch to "nig"
                                <1> namei_2: ; 1:
                                <1>       ; 18/10/2015
0000DBF3 8935[44040300]         <1>       mov   [nbase], esi
0000DBF9 66890D[48040300]       <1>       mov   [ncount], cx
                                <1>       ;
                                <1>       ;mov  dx, 2
0000DC00 B202                   <1>       mov   dl, 2 ; user flag (read, non-owner)
0000DC02 E8BE150000             <1>       call  access
                                <1>             ; jsr r0,access; 2 / get i-node with i-number r1
                                <1>       ; 'access' will not return here if user has not "r" permission !
0000DC07 66F705[00000300]00-    <1>       test  word [i.flgs], 4000h
0000DC0F 40                     <1>
                                <1>             ; bit $40000,i.flgs / directory i-node?
0000DC10 746A                   <1>         jz      short namei_err
                                <1>             ; beq error3 / no, got an error
                                <1>       ; 16/06/2015 - 32 bit modifications (Retro UNIX 386 v1)
0000DC12 31C0                   <1>       xor   eax, eax
0000DC14 A3[80030300]           <1>       mov   [u.off], eax ; 0
0000DC19 66A1[55040300]         <1>       mov   ax, [i.size]
0000DC1F A3[78030300]           <1>       mov   [u.dirp], eax
                                <1>             ; mov i.size,u.dirp / put size of directory in u.dirp
                                <1>             ; clr u.off / u.off is file offset used by user
0000DC24 C705[74030300]-        <1>       mov   dword [u.fofp], u.off
0000DC2A [80030300]             <1>
                                <1>             ; mov $u.off,u.fofp / u.fofp is a pointer to
                                <1>                                 ; / the offset portion of fsp entry
                                <1> namei_3: ; 2:
0000DC2E C705[84030300]-        <1>       mov   dword [u.base], u.dirbuf
0000DC34 [98030300]             <1>
                                <1>             ; mov $u.dirbuf,u.base / u.dirbuf holds a file name
                                <1>                                    ; / copied from a directory
0000DC38 C705[88030300]1000-    <1>       mov   dword [u.count], 16 ; 04/12/2015 (10 -> 16)
0000DC40 0000                   <1>
                                <1>             ; mov $10.,u.count / u.count is byte count
                                <1>                                ; / for reads and writes
0000DC42 66A1[51040300]         <1>       mov   ax, [ii]
                                <1>       ; 31/07/2013 ('namei_r') - 16/06/2015 ('u.kcall')
0000DC48 FE05[C6030300]         <1>       inc     byte [u.kcall] ; the caller is 'namei' sign
0000DC4E E881070000             <1>       call  readi
                                <1>             ; jsr r0,readi / read 10. bytes of file
                                <1>                  ; with i-number (r1); i.e. read a directory entry
0000DC53 8B0D[8C030300]         <1>       mov   ecx, [u.nread]
0000DC59 09C9                   <1>       or    ecx, ecx
                                <1>             ; tst u.nread
0000DC5B 741B                   <1>       jz    short nib
                                <1>             ; ble nib / gives error return
                                <1>       ;
0000DC5D 668B1D[98030300]       <1>       mov   bx, [u.dirbuf]
0000DC64 6621DB                 <1>       and   bx, bx
                                <1>             ; tst u.dirbuf /
0000DC67 7522                   <1>       jnz   short namei_4
                                <1>             ; bne 3f / branch when active directory entry
                                <1>                    ; / (i-node word in entry non zero)
0000DC69 A1[80030300]           <1>       mov   eax, [u.off]
0000DC6E 83E810                 <1>       sub   eax, 16 ; 04/12/2015 (10 -> 16)
0000DC71 A3[78030300]           <1>       mov   [u.dirp], eax
                                <1>             ; mov u.off,u.dirp
                                <1>             ; sub $10.,u.dirp
0000DC76 EBB6                   <1>       jmp   short namei_3
                                <1>             ; br 2b
                                <1>
                                <1>       ; 18/07/2013
                                <1> nib:
0000DC78 31C0                   <1>       xor   eax, eax  ; xor ax, ax ; ax = 0 -> file not found
0000DC7A F9                     <1>       stc
                                <1> nig:
0000DC7B C3                     <1>       retn
                                <1>
                                <1> namei_err:
                                <1>       ; 16/06/2015
0000DC7C C705[C8030300]1300-    <1>       mov   dword [u.error], ERR_NOT_DIR ; 'not a directory !' error
0000DC84 0000                   <1>
0000DC86 E932E8FFFF             <1>       jmp   error
                                <1>
                                <1> namei_4: ; 3:
```

```
40266                                  <1>       ; 18/10/2015
40267                                  <1>       ; 12/10/2015
40268                                  <1>       ; 21/08/2015
40269                                  <1>       ; 18/07/2015
40270 0000DC8B 8B2D[7C030300]          <1>       mov    ebp, [u.namep]
40271                                  <1>              ; mov u.namep,r2 / u.namep points into a file name string
40272 0000DC91 BF[9A030300]            <1>       mov    edi, u.dirbuf + 2
40273                                  <1>              ; mov $u.dirbuf+2,r3 / points to file name of directory entry
40274                                  <1>       ; 18/10/2015
40275 0000DC96 8B35[44040300]          <1>       mov    esi, [nbase]
40276 0000DC9C 668B0D[48040300]        <1>       mov    cx, [ncount]
40277                                  <1>       ;
40278 0000DCA3 6621C9                  <1>       and    cx, cx
40279 0000DCA6 7505                    <1>       jnz    short namei_5
40280                                  <1>       ;
40281 0000DCA8 E850000000              <1>       call   trans_addr_nm ; convert virtual address to physical
40282                                  <1>              ; esi = physical address (page start + offset)
40283                                  <1>              ; ecx = byte count in the page
40284                                  <1> namei_5: ; 3:
40285 0000DCAD 45                      <1>       inc    ebp ; 18/07/2015
40286 0000DCAE AC                      <1>       lodsb   ; mov al, [esi] ; inc esi (al = r4)
40287                                  <1>              ; movb (r2)+,r4 / move a character from u.namep string into r4
40288 0000DCAF 08C0                    <1>       or     al, al
40289 0000DCB1 741D                    <1>       jz     short namei_7
40290                                  <1>              ; beq 3f / if char is nul, then the last char in string
40291                                  <1>                    ; / has been moved
40292 0000DCB3 3C2F                    <1>       cmp    al, '/'
40293                                  <1>              ; cmp r4,$'/ / is char a </>
40294 0000DCB5 7419                    <1>       je     short namei_7
40295                                  <1>              ; beq 3f
40296                                  <1>       ; 12/10/2015
40297 0000DCB7 6649                    <1>       dec    cx ; remain byte count in the page
40298 0000DCB9 7505                    <1>       jnz    short namei_6
40299 0000DCBB E83D000000              <1>       call   trans_addr_nm ; convert virtual address to physical
40300                                  <1>              ; esi = physical address (page start + offset)
40301                                  <1>              ; ecx = byte count in the page
40302                                  <1> namei_6:
40303 0000DCC0 81FF[A8030300]          <1>         cmp     edi, u.dirbuf + 16 ; 04/12/2015 (10 -> 16)
40304                                  <1>              ; cmp r3,$u.dirbuf+10. / have I checked
40305                                  <1>                                   ; / all 8 bytes of file name
40306 0000DCC6 74E5                    <1>       je     short namei_5
40307                                  <1>              ; beq 3b
40308 0000DCC8 AE                      <1>       scasb
40309                                  <1>              ; cmpb (r3)+,r4 / compare char in u.namep string to file name
40310                                  <1>                             ; / char read from directory
40311 0000DCC9 74E2                    <1>       je     short namei_5
40312                                  <1>              ; beq 3b / branch if chars match
40313                                  <1>
40314 0000DCCB E95EFFFFFF              <1>         jmp    namei_3 ; 2b
40315                                  <1>              ; br 2b / file names do not match go to next directory entry
40316                                  <1> namei_7: ; 3:
40317 0000DCD0 81FF[A8030300]          <1>       cmp    edi, u.dirbuf + 16 ; 04/12/2015 (10 -> 16)
40318                                  <1>              ; cmp r3,$u.dirbuf+10. / if equal all 8 bytes were matched
40319 0000DCD6 740A                    <1>       je     short namei_8
40320                                  <1>              ; beq 3f
40321 0000DCD8 8A27                    <1>       mov    ah, [edi]
40322                                  <1>       ;inc   edi
40323 0000DCDA 20E4                    <1>       and    ah, ah
40324                                  <1>              ; tstb (r3)+ /
40325 0000DCDC 0F854CFFFFFF            <1>         jnz     namei_3
40326                                  <1>              ; bne 2b
40327                                  <1> namei_8: ; 3
40328 0000DCE2 892D[7C030300]          <1>       mov    [u.namep], ebp ; 18/07/2015
40329                                  <1>              ; mov r2,u.namep / u.namep points to char
40330                                  <1>                               ; / following a / or nul
40331                                  <1>       ;mov   bx, [u.dirbuf]
40332                                  <1>              ; mov u.dirbuf,r1 / move i-node number in directory
40333                                  <1>                                ; / entry to r1
40334 0000DCE8 20C0                    <1>       and    al, al
40335                                  <1>              ; tst r4 / if r4 = 0 the end of file name reached,
40336                                  <1>                    ;  / if r4 = </> then go to next directory
40337                                  <1>       ; mov  ax, bx
40338 0000DCEA 66A1[98030300]          <1>       mov    ax, [u.dirbuf] ; 17/06/2015
40339 0000DCF0 0F85FDFEFFFF            <1>         jnz     namei_2
40340                                  <1>              ; bne 1b
40341                                  <1>       ; AX = i-number of the file
40342                                  <1> ;;nig:
40343 0000DCF6 C3                      <1>       retn
40344                                  <1>              ; tst (r0)+ / gives non-error return
40345                                  <1> ;;nib:
40346                                  <1> ;;    xor    ax, ax ; Retro UNIX 8086 v1 modification !
40347                                  <1>                     ; ax = 0 -> file not found
40348                                  <1> ;;    stc    ; 27/05/2013
40349                                  <1> ;;    retn
40350                                  <1>              ; rts r0
40351                                  <1>
40352                                  <1> trans_addr_nmbp:
40353                                  <1>       ; 18/10/2015
40354                                  <1>       ; 12/10/2015
40355 0000DCF7 8B2D[7C030300]          <1>       mov    ebp, [u.namep]
40356                                  <1> trans_addr_nm:
40357                                  <1>       ; Convert virtual (pathname) address to physical address
40358                                  <1>       ; (Retro UNIX 386 v1 feature only !)
40359                                  <1>       ; 18/10/2015
40360                                  <1>       ; 12/10/2015 (u.pnbase & u.pncount has been removed from code)
40361                                  <1>       ; 02/07/2015
40362                                  <1>       ; 17/06/2015
40363                                  <1>       ; 16/06/2015
40364                                  <1>       ;
40365                                  <1>       ; INPUTS:
40366                                  <1>       ;     ebp = pathname address (virtual) ; [u.namep]
40367                                  <1>       ;     [u.pgdir] = user's page directory
40368                                  <1>       ; OUTPUT:
```

```
40369                                  <1>       ;       esi = physical address of the pathname
40370                                  <1>       ;     ecx = remain byte count in the page
40371                                  <1>       ;
40372                                  <1>       ; (Modified registers: EAX, EBX, ECX, EDX, ESI)
40373                                  <1>       ;
40374 0000DCFD 833D[BC030300]00        <1>         cmp     dword [u.ppgdir], 0  ; /etc/init ? (sysexec)
40375 0000DD04 7618                    <1>       jna   short trans_addr_nmk ; the caller is os kernel;
40376                                  <1>                                  ; it is already physical address
40377 0000DD06 50                      <1>       push  eax
40378 0000DD07 89EB                    <1>       mov   ebx, ebp ; [u.namep] ; pathname address (virtual)
40379 0000DD09 E88175FFFF              <1>             call  get_physical_addr ; get physical address
40380 0000DD0E 7204                    <1>       jc    short tr_addr_nm_err
40381                                  <1>       ; 18/10/2015
40382                                  <1>       ; eax = physical address
40383                                  <1>       ; cx = remain byte count in page (1-4096)
40384                                  <1>             ; 12/10/2015 (cx = [u.pncount])
40385 0000DD10 89C6                    <1>       mov   esi, eax ; 12/10/2015 (esi=[u.pnbase])
40386 0000DD12 58                      <1>       pop   eax
40387 0000DD13 C3                      <1>       retn
40388                                  <1>
40389                                  <1> tr_addr_nm_err:
40390 0000DD14 A3[C8030300]            <1>       mov   [u.error], eax
40391                                  <1>       ;pop  eax
40392 0000DD19 E99FE7FFFF              <1>       jmp   error
40393                                  <1>
40394                                  <1> trans_addr_nmk:
40395                                  <1>       ; 12/10/2015
40396                                  <1>       ; 02/07/2015
40397 0000DD1E 8B35[7C030300]          <1>       mov   esi, [u.namep]  ; [u.pnbase]
40398 0000DD24 66B90010                <1>       mov   cx, PAGE_SIZE ; 4096 ; [u.pncount]
40399 0000DD28 C3                      <1>       retn
40400                                  <1>
40401                                  <1> syschdir:
40402                                  <1>       ; / makes the directory specified in the argument
40403                                  <1>       ; / the current directory
40404                                  <1>       ;
40405                                  <1>       ; 23/06/2015 (Retro UNIX 386 v1 - Beginning)
40406                                  <1>       ; 19/06/2013 (Retro UNIX 8086 v1)
40407                                  <1>       ;
40408                                  <1>       ; 'syschdir' makes the directory specified in its argument
40409                                  <1>       ; the current working directory.
40410                                  <1>       ;
40411                                  <1>       ; Calling sequence:
40412                                  <1>       ;     syschdir; name
40413                                  <1>       ; Arguments:
40414                                  <1>       ;     name - address of the path name of a directory
40415                                  <1>       ;            terminated by nul byte.
40416                                  <1>       ; Inputs: -
40417                                  <1>       ; Outputs: -
40418                                  <1>       ; ............................................................
40419                                  <1>       ;
40420                                  <1>       ; Retro UNIX 8086 v1 modification:
40421                                  <1>       ;      The user/application program puts address of
40422                                  <1>       ;      the path name in BX register as 'syschdir'
40423                                  <1>       ;      system call argument.
40424                                  <1>
40425 0000DD29 891D[7C030300]          <1>       mov   [u.namep], ebx
40426                                  <1>             ;jsr r0,arg; u.namep / u.namep points to path name
40427 0000DD2F E87EFEFFFF              <1>       call  namei
40428                                  <1>             ; jsr r0,namei / find its i-number
40429                                  <1>       ;jc   error
40430                                  <1>             ; br error3
40431 0000DD34 730F                    <1>       jnc   short syschdir0
40432                                  <1>       ; 'directory not found !' error
40433 0000DD36 C705[C8030300]0C00-     <1>       mov   dword [u.error], ERR_DIR_NOT_FOUND ; 12
40434 0000DD3E 0000                    <1>
40435 0000DD40 E978E7FFFF              <1>       jmp   error
40436                                  <1> syschdir0:
40437 0000DD45 E87B140000              <1>       call  access
40438                                  <1>             ; jsr r0,access; 2 / get i-node into core
40439 0000DD4A 66F705[00000300]00-     <1>       test  word [i.flgs], 4000h
40440 0000DD52 40                      <1>
40441                                  <1>             ; bit $40000,i.flgs / is it a directory?
40442                                  <1>       ;jz   error
40443                                  <1>             ; beq error3 / no error
40444 0000DD53 750F                    <1>       jnz   short syschdir1
40445 0000DD55 C705[C8030300]1300-     <1>       mov   dword [u.error], ERR_NOT_DIR ; 'not a valid directory !'
40446 0000DD5D 0000                    <1>
40447 0000DD5F E959E7FFFF              <1>       jmp   error
40448                                  <1> syschdir1:
40449 0000DD64 66A3[68030300]          <1>       mov   [u.cdir], ax
40450                                  <1>             ; mov r1,u.cdir / move i-number to users
40451                                  <1>                           ; / current directory
40452 0000DD6A 66A1[46030300]          <1>       mov   ax, [cdev]
40453 0000DD70 66A3[AE030300]          <1>       mov   [u.cdrv], ax
40454                                  <1>             ; mov cdev,u.cdev / move its device to users
40455                                  <1>                             ; / current device
40456 0000DD76 E962E7FFFF              <1>       jmp   sysret
40457                                  <1>             ; br sysret3
40458                                  <1>
40459                                  <1> syschmod: ; < change mode of file >
40460                                  <1>       ; 13/01/2017 - TRDOS 386 (TRDOS v2.0)
40461                                  <1>       ; temporary !
40462 0000DD7B B801000000              <1>       mov   eax, ERR_INV_FNUMBER ; 'invalid function number !'
40463 0000DD80 A3[C8030300]            <1>         mov     [u.error], eax
40464 0000DD85 A3[64030300]            <1>         mov     [u.r0], eax
40465 0000DD8A E92EE7FFFF              <1>       jmp   error
40466                                  <1>
40467                                  <1> isown:
40468                                  <1>       ; 22/06/2015 (Retro UNIX 386 v1 - Beginning)
40469                                  <1>       ; 04/05/2013 - 07/07/2013 (Retro UNIX 8086 v1)
40470                                  <1>       ;
40471                                  <1>       ; 'isown' is given a file name (the 1st argument).
```

```
40472                                  <1>       ;  It find the i-number of that file via 'namei'
40473                                  <1>       ;  then gets the i-node into core via 'iget'.
40474                                  <1>       ;  It then tests to see if the user is super user.
40475                                  <1>       ;  If not, it cheks to see if the user is owner of
40476                                  <1>       ;  the file. If he is not an error occurs.
40477                                  <1>       ;  If user is the owner 'setimod' is called to indicate
40478                                  <1>       ;  the inode has been modificed and the 2nd argument of
40479                                  <1>       ;  the call is put in r2.
40480                                  <1>       ;
40481                                  <1>       ; INPUTS ->
40482                                  <1>       ;    arguments of syschmod and syschown calls
40483                                  <1>       ; OUTPUTS ->
40484                                  <1>       ;    u.uid - id of user
40485                                  <1>       ;    imod - set to a 1
40486                                  <1>       ;    r2 - contains second argument of the system call
40487                                  <1>       ;
40488                                  <1>       ;   ((AX=R2) output as 2nd argument)
40489                                  <1>       ;
40490                                  <1>         ; ((Modified registers: eAX, eDX, eBX, eCX, eSI, eDI, eBP))
40491                                  <1>       ;
40492                                  <1>             ; jsr r0,arg2 / u.namep points to file name
40493                                  <1>       ;; ! 2nd argument on top of stack !
40494                                  <1>       ;; 22/06/2015 - 32 bit modifications
40495                                  <1>       ;; 07/07/2013
40496 0000DD8F 891D[7C030300]         <1>       mov    [u.namep], ebx ;; 1st argument
40497 0000DD95 51                     <1>       push   ecx ;; 2nd argument
40498                                  <1>       ;;
40499 0000DD96 E817FEFFFF             <1>       call   namei
40500                                  <1>             ; jsr r0,namei / get its i-number
40501                                  <1>        ; Retro UNIX 8086 v1 modification !
40502                                  <1>        ; ax = 0 -> file not found
40503                                  <1>       ;and   ax, ax
40504                                  <1>       ;jz    error
40505                                  <1>       ;jc    error ; 27/05/2013
40506                                  <1>             ; br error3
40507 0000DD9B 730F                   <1>       jnc    short isown0
40508                                  <1>       ; 'file not found !' error
40509 0000DD9D C705[C8030300]0C00-     <1>       mov    dword [u.error], ERR_FILE_NOT_FOUND ; 12
40510 0000DDA5 0000                   <1>
40511 0000DDA7 E911E7FFFF             <1>       jmp    error
40512                                  <1> isown0:
40513 0000DDAC E80E140000             <1>       call   iget
40514                                  <1>             ; jsr r0,iget / get i-node into core
40515 0000DDB1 A0[B0030300]           <1>       mov    al, [u.uid] ; 02/08/2013
40516 0000DDB6 08C0                   <1>       or     al, al
40517                                  <1>             ; tstb u.uid / super user?
40518 0000DDB8 7417                   <1>       jz     short isown1
40519                                  <1>             ; beq 1f / yes, branch
40520 0000DDBA 3A05[03000300]         <1>       cmp    al, [i.uid]
40521                                  <1>             ; cmpb i.uid,u.uid / no, is this the owner of
40522                                  <1>                           ; / the file
40523                                  <1>       ;jne   error
40524                                  <1>             ; beq 1f / yes
40525                                  <1>             ; jmp error3 / no, error
40526 0000DDC0 740F                   <1>       je     short isown1
40527                                  <1>
40528 0000DDC2 C705[C8030300]0B00-     <1>       mov    dword [u.error], ERR_NOT_OWNER  ; 11
40529 0000DDCA 0000                   <1>
40530                                  <1>                     ;  'permission denied !' error
40531 0000DDCC E9ECE6FFFF             <1>       jmp    error
40532                                  <1> isown1: ; 1:
40533 0000DDD1 E8ED130000             <1>       call   setimod
40534                                  <1>             ; jsr r0,setimod / indicates
40535                                  <1>             ;            ; / i-node has been modified
40536 0000DDD6 58                     <1>       pop    eax ; 2nd argument
40537                                  <1>             ; mov (sp)+,r2 / mode is put in r2
40538                                  <1>                    ; / (u.off put on stack with 2nd arg)
40539 0000DDD7 C3                     <1>       retn
40540                                  <1>             ; rts r0
40541                                  <1>
40542                                  <1> ;;arg:  ; < get system call arguments >
40543                                  <1>       ; 'arg' extracts an argument for a routine whose call is
40544                                  <1>       ; of form:
40545                                  <1>       ;     sys 'routine' ; arg1
40546                                  <1>       ;           or
40547                                  <1>       ;     sys 'routine' ; arg1 ; arg2
40548                                  <1>       ;           or
40549                                  <1>       ;     sys 'routine' ; arg1;...;arg10 (sys exec)
40550                                  <1>       ;
40551                                  <1>       ; INPUTS ->
40552                                  <1>       ;    u.sp+18 - contains a pointer to one of arg1..argn
40553                                  <1>       ;     This pointers's value is actually the value of
40554                                  <1>       ;     update pc at the the trap to sysent (unkni) is
40555                                  <1>       ;     made to process the sys instruction
40556                                  <1>       ;    r0 - contains the return address for the routine
40557                                  <1>       ;     that called arg. The data in the word pointer
40558                                  <1>       ;     to by the return address is used as address
40559                                  <1>       ;     in which the extracted argument is stored
40560                                  <1>       ;
40561                                  <1>       ; OUTPUTS ->
40562                                  <1>       ;    'address' - contains the extracted argument
40563                                  <1>       ;    u.sp+18 - is incremented by 2
40564                                  <1>       ;    r1 - contains the extracted argument
40565                                  <1>       ;    r0 - points to the next instruction to be
40566                                  <1>       ;       executed in the calling routine.
40567                                  <1>       ;
40568                                  <1>
40569                                  <1>       ; mov u.sp,r1
40570                                  <1>       ; mov *18.(r1),*(r0)+ / put argument of system call
40571                                  <1>                    ; / into argument of arg2
40572                                  <1>       ; add $2,18.(r1) / point pc on stack
40573                                  <1>                          ; / to next system argument
40574                                  <1>       ; rts r0
```

```
                                        <1>
                                        <1> ;;arg2: ; < get system calls arguments - with file name pointer>
                                        <1>       ; 'arg2' takes first argument in system call
                                        <1>       ;  (pointer to name of the file) and puts it in location
                                        <1>       ;  u.namep; takes second argument and puts it in u.off
                                        <1>       ;  and on top of the stack
                                        <1>       ;
                                        <1>       ; INPUTS ->
                                        <1>       ;    u.sp, r0
                                        <1>       ;
                                        <1>       ; OUTPUTS ->
                                        <1>       ;    u.namep
                                        <1>       ;    u.off
                                        <1>       ;    u.off pushed on stack
                                        <1>       ;    r1
                                        <1>       ;
                                        <1>
                                        <1>       ; jsr  r0,arg; u.namep / u.namep contains value of
                                        <1>                            ; / first arg in sys call
                                        <1>       ; jsr r0,arg; u.off / u.off contains value of
                                        <1>                            ; / second arg in sys call
                                        <1>       ; mov r0,r1 / r0 points to calling routine
                                        <1>       ; mov (sp),r0 / put operation code back in r0
                                        <1>       ; mov u.off,(sp) / put pointer to second argument
                                        <1>                        ; / on stack
                                        <1>       ; jmp (r1) / return to calling routine
                                        <1>
                                        <1> syschown: ; < change owner of file >
                                        <1>       ; 23/06/2015 (Retro UNIX 386 v1 - Beginning)
                                        <1>       ; 20/06/2013 - 02/08/2013 (Retro UNIX 8086 v1)
                                        <1>       ;
                                        <1>       ; 'syschown' changes the owner of the file whose name is given
                                        <1>       ; as null terminated string pointed to by 'name' has it's owner
                                        <1>       ; changed to 'owner'
                                        <1>       ;
                                        <1>       ; Calling sequence:
                                        <1>       ;     syschown; name; owner
                                        <1>       ; Arguments:
                                        <1>       ;     name - address of the file name
                                        <1>       ;            terminated by null byte.
                                        <1>       ;     owner - (new) owner (number/ID)
                                        <1>       ;
                                        <1>       ; Inputs: -
                                        <1>       ; Outputs: -
                                        <1>       ; ..........................................................
                                        <1>       ;
                                        <1>       ; Retro UNIX 8086 v1 modification:
                                        <1>       ;      'syschown' system call has two arguments; so,
                                        <1>       ;     * 1st argument, name is pointed to by BX register
                                        <1>       ;     * 2nd argument, owner number is in CX register
                                        <1>       ;
                                        <1>       ; / name; owner
 0000DDD8 E8B2FFFFFF                    <1>       call  isown
                                        <1>             ; jsr r0,isown / get the i-node and check user status
 0000DDDD 803D[B0030300]00              <1>       cmp   byte [u.uid], 0 ; 02/08/2013
                                        <1>             ; tstb u.uid / super user
 0000DDE4 7418                          <1>       jz    short syschown1
                                        <1>             ; beq 2f / yes, 2f
 0000DDE6 F605[00000300]20              <1>         test    byte [i.flgs], 20h ; 32
                                        <1>             ; bit $40,i.flgs / no, set userid on execution?
                                        <1>       ;jnz  error
                                        <1>             ; bne 3f / yes error, could create Trojan Horses
 0000DDED 740F                          <1>       jz    short syschown1
                                        <1>       ; 'permission denied !'
 0000DDEF C705[C8030300]0B00-           <1>       mov   dword [u.error], ERR_FILE_ACCESS  ; 11
 0000DDF7 0000                          <1>
 0000DDF9 E9BFE6FFFF                    <1>       jmp   error
                                        <1> syschown1: ; 2:
                                        <1>       ; AL = owner (number/ID)
 0000DDFE A2[03000300]                  <1>       mov   [i.uid], al ; 23/06/2015
                                        <1>             ;  movb      r2,i.uid / no, put the new owners id
                                        <1>                           ; / in the i-node
 0000DE03 E9D5E6FFFF                    <1>       jmp   sysret
                                        <1>       ; 1:
                                        <1>             ; jmp sysret4
                                        <1>       ; 3:
                                        <1>             ; jmp  error
                                        <1>
                                        <1> systime: ; / get time of year
                                        <1>       ; 23/06/2015 (Retro UNIX 386 v1 - Beginning)
                                        <1>       ; 20/06/2013 (Retro UNIX 8086 v1)
                                        <1>       ;
                                        <1>       ; 20/06/2013
                                        <1>       ; 'systime' gets the time of the year.
                                        <1>       ; The present time is put on the stack.
                                        <1>       ;
                                        <1>       ; Calling sequence:
                                        <1>       ;     systime
                                        <1>       ; Arguments: -
                                        <1>       ;
                                        <1>       ; Inputs: -
                                        <1>       ; Outputs: sp+2, sp+4 - present time
                                        <1>       ; ..........................................................
                                        <1>       ;
                                        <1>       ; Retro UNIX 8086 v1 modification:
                                        <1>       ;      'systime' system call will return to the user
                                        <1>       ;     with unix time (epoch) in DX:AX register pair
                                        <1>       ;
                                        <1>       ;     !! Major modification on original Unix v1 'systime'
                                        <1>       ;     system call for PC compatibility !!
                                        <1>
 0000DE08 E8B9130000                    <1>       call  epoch
 0000DE0D A3[64030300]                  <1>       mov   [u.r0], eax
```

```
40678                                  <1>             ; mov s.time,4(sp)
40679                                  <1>             ; mov s.time+2,2(sp) / put the present time
40680                                  <1>                                  ; / on the stack
40681                                  <1>             ; br sysret4
40682 0000DE12 E9C6E6FFFF             <1>       jmp   sysret
40683                                  <1>
40684                                  <1> sysstime: ; / set time
40685                                  <1>       ; 23/06/2015 (Retro UNIX 386 v1 - Beginning)
40686                                  <1>       ; 20/06/2013 - 02/08/2013 (Retro UNIX 8086 v1)
40687                                  <1>       ;
40688                                  <1>       ; 'sysstime' sets the time. Only super user can use this call.
40689                                  <1>       ;
40690                                  <1>       ; Calling sequence:
40691                                  <1>       ;     sysstime
40692                                  <1>       ; Arguments: -
40693                                  <1>       ;
40694                                  <1>       ; Inputs: sp+2, sp+4 - time system is to be set to.
40695                                  <1>       ; Outputs: -
40696                                  <1>       ; ..........................................................
40697                                  <1>       ;
40698                                  <1>       ; Retro UNIX 8086 v1 modification:
40699                                  <1>       ;     the user calls 'sysstime' with unix (epoch) time
40700                                  <1>       ;     (to be set) is in CX:BX register pair as two arguments.
40701                                  <1>       ;
40702                                  <1>       ;     Retro UNIX 8086 v1 argument transfer method 2 is used
40703                                  <1>       ;     to get sysstime system call arguments from the user;
40704                                  <1>       ;     * 1st argument, lowword of unix time is in BX register
40705                                  <1>       ;     * 2nd argument, highword of unix time is in CX register
40706                                  <1>       ;
40707                                  <1>       ;     !! Major modification on original Unix v1 'sysstime'
40708                                  <1>       ;     system call for PC compatibility !!
40709                                  <1>
40710 0000DE17 803D[B0030300]00       <1>       cmp   byte [u.uid], 0
40711                                  <1>             ; tstb u.uid / is user the super user
40712                                  <1>       ;ja   error
40713                                  <1>             ; bne error4 / no, error
40714 0000DE1E 760F                   <1>       jna   short systime1
40715                                  <1>       ; 'permission denied !'
40716 0000DE20 C705[C8030300]0B00-    <1>       mov   dword [u.error], ERR_NOT_SUPERUSER  ; 11
40717 0000DE28 0000                   <1>
40718 0000DE2A E98EE6FFFF             <1>       jmp   error
40719                                  <1> systime1:
40720                                  <1>       ; 23/06/2015 (Retro UNIX 386 v1 - 32 bit version)
40721                                  <1>       ; EBX = unix (epoch) time (from user)
40722 0000DE2F 89D8                   <1>       mov   eax, ebx
40723 0000DE31 E892130000             <1>       call  set_date_time
40724                                  <1>             ; mov 4(sp),s.time
40725                                  <1>             ; mov 2(sp),s.time+2 / set the system time
40726 0000DE36 E9A2E6FFFF             <1>       jmp   sysret
40727                                  <1>             ; br sysret4
40728                                  <1>
40729                                  <1> sysbreak:
40730                                  <1>       ; 18/10/2015
40731                                  <1>       ; 07/10/2015
40732                                  <1>       ; 23/06/2015 (Retro UNIX 386 v1 - Beginning)
40733                                  <1>       ; 20/06/2013 - 24/03/2014 (Retro UNIX 8086 v1)
40734                                  <1>       ;
40735                                  <1>       ; 'sysbreak' sets the programs break points.
40736                                  <1>       ; It checks the current break point (u.break) to see if it is
40737                                  <1>       ; between "core" and the stack (sp). If it is, it is made an
40738                                  <1>       ; even address (if it was odd) and the area between u.break
40739                                  <1>       ; and the stack is cleared. The new breakpoint is then put
40740                                  <1>       ; in u.break and control is passed to 'sysret'.
40741                                  <1>       ;
40742                                  <1>       ; Calling sequence:
40743                                  <1>       ;     sysbreak; addr
40744                                  <1>       ; Arguments: -
40745                                  <1>       ;
40746                                  <1>       ; Inputs: u.break - current breakpoint
40747                                  <1>       ; Outputs: u.break - new breakpoint
40748                                  <1>       ;     area between old u.break and the stack (sp) is cleared.
40749                                  <1>       ; ..........................................................
40750                                  <1>       ;
40751                                  <1>       ; Retro UNIX 8086 v1 modification:
40752                                  <1>       ;     The user/application program puts breakpoint address
40753                                  <1>       ;      in BX register as 'sysbreak' system call argument.
40754                                  <1>       ;     (argument transfer method 1)
40755                                  <1>       ;
40756                                  <1>       ;  NOTE: Beginning of core is 0 in Retro UNIX 8086 v1 !
40757                                  <1>       ;     ((!'sysbreak' is not needed in Retro UNIX 8086 v1!))
40758                                  <1>       ;  NOTE:
40759                                  <1>       ;     'sysbreak' clears extended part (beyond of previous
40760                                  <1>       ;     'u.break' address) of user's memory for original unix's
40761                                  <1>       ;     'bss' compatibility with Retro UNIX 8086 v1 (19/11/2013)
40762                                  <1>
40763                                  <1>             ; mov u.break,r1 / move users break point to r1
40764                                  <1>             ; cmp r1,$core / is it the same or lower than core?
40765                                  <1>             ; blos 1f / yes, 1f
40766                                  <1>       ; 23/06/2015
40767 0000DE3B 8B2D[90030300]         <1>       mov   ebp, [u.break] ; virtual address (offset)
40768                                  <1>       ;and  ebp, ebp
40769                                  <1>       ;jz   short sysbreak_3
40770                                  <1>       ; Retro UNIX 386 v1 NOTE: u.break points to virtual address !!!
40771                                  <1>       ; (Even break point address is not needed for Retro UNIX 386 v1)
40772 0000DE41 8B15[5C030300]         <1>       mov   edx, [u.sp] ; kernel stack at the beginning of sys call
40773 0000DE47 83C20C                 <1>       add   edx, 12 ; EIP -4-> CS -4-> EFLAGS -4-> ESP (user)
40774                                  <1>       ; 07/10/2015
40775 0000DE4A 891D[90030300]         <1>       mov   [u.break], ebx ; virtual address !!!
40776                                  <1>       ;
40777 0000DE50 3B1A                   <1>       cmp   ebx, [edx] ; compare new break point with
40778                                  <1>                        ; with top of user's stack (virtual!)
40779 0000DE52 7327                   <1>       jnb   short sysbreak_3
40780                                  <1>             ; cmp r1,sp / is it the same or higher
```

```
40781                                  <1>                     ; / than the stack?
40782                                  <1>             ; bhis 1f / yes, 1f
40783 0000DE54 89DE                    <1>       mov   esi, ebx
40784 0000DE56 29EE                    <1>       sub   esi, ebp ; new break point - old break point
40785 0000DE58 7621                    <1>       jna   short sysbreak_3
40786                                  <1>       ;push ebx
40787                                  <1> sysbreak_1:
40788 0000DE5A 89EB                    <1>       mov   ebx, ebp
40789 0000DE5C E82E74FFFF              <1>       call  get_physical_addr ; get physical address
40790 0000DE61 0F82ADFEFFFF            <1>       jc    tr_addr_nm_err
40791                                  <1>       ; 18/10/2015
40792 0000DE67 89C7                    <1>       mov   edi, eax
40793 0000DE69 29C0                    <1>       sub   eax, eax ; 0
40794                                  <1>             ; ECX = remain byte count in page (1-4096)
40795 0000DE6B 39CE                    <1>       cmp   esi, ecx
40796 0000DE6D 7302                    <1>       jnb   short sysbreak_2
40797 0000DE6F 89F1                    <1>       mov   ecx, esi
40798                                  <1> sysbreak_2:
40799 0000DE71 29CE                    <1>       sub   esi, ecx
40800 0000DE73 01CD                    <1>       add   ebp, ecx
40801 0000DE75 F3AA                    <1>       rep   stosb
40802 0000DE77 09F6                    <1>       or    esi, esi
40803 0000DE79 75DF                    <1>       jnz   short sysbreak_1
40804                                  <1>       ;
40805                                  <1>             ; bit $1,r1 / is it an odd address
40806                                  <1>             ; beq 2f / no, its even
40807                                  <1>             ; clrb (r1)+ / yes, make it even
40808                                  <1>       ; 2: / clear area between the break point and the stack
40809                                  <1>             ; cmp r1,sp / is it higher or same than the stack
40810                                  <1>             ; bhis 1f / yes, quit
40811                                  <1>             ; clr (r1)+ / clear word
40812                                  <1>             ; br 2b / go back
40813                                  <1>       ;pop  ebx
40814                                  <1> sysbreak_3: ; 1:
40815                                  <1>       ;mov  [u.break], ebx ; virtual address !!!
40816                                  <1>             ; jsr r0,arg; u.break / put the "address"
40817                                  <1>                   ; / in u.break (set new break point)
40818                                  <1>             ; br sysret4 / br sysret
40819 0000DE7B E95DE6FFFF              <1>       jmp   sysret
40820                                  <1>
40821                                  <1> sysseek: ; / moves read write pointer in an fsp entry
40822                                  <1>       ; 06/11/2016 - TRDOS 386 (TRDOS v2.0)
40823                                  <1>       ; 22/06/2015 (Retro UNIX 386 v1 - Beginning)
40824                                  <1>       ; 07/07/2013 - 05/08/2013 (Retro UNIX 8086 v1)
40825                                  <1>       ;
40826                                  <1>       ; 'sysseek' changes the r/w pointer of (3rd word of in an
40827                                  <1>       ; fsp entry) of an open file whose file descriptor is in u.r0.
40828                                  <1>       ; The file descriptor refers to a file open for reading or
40829                                  <1>       ; writing. The read (or write) pointer is set as follows:
40830                                  <1>       ;     * if 'ptrname' is 0, the pointer is set to offset.
40831                                  <1>       ;     * if 'ptrname' is 1, the pointer is set to its
40832                                  <1>       ;       current location plus offset.
40833                                  <1>       ;     * if 'ptrname' is 2, the pointer is set to the
40834                                  <1>       ;       size of file plus offset.
40835                                  <1>       ; The error bit (e-bit) is set for an undefined descriptor.
40836                                  <1>       ;
40837                                  <1>       ; Calling sequence:
40838                                  <1>       ;     sysseek; offset; ptrname
40839                                  <1>       ; Arguments:
40840                                  <1>       ;     offset - number of bytes desired to move
40841                                  <1>       ;              the r/w pointer
40842                                  <1>       ;     ptrname - a switch indicated above
40843                                  <1>       ;
40844                                  <1>       ; Inputs: r0 - file descriptor
40845                                  <1>       ; Outputs: -
40846                                  <1>       ; ............................................................
40847                                  <1>       ;
40848                                  <1>       ; Retro UNIX 8086 v1 modification:
40849                                  <1>       ;       'sysseek' system call has three arguments; so,
40850                                  <1>       ;     * 1st argument, file descriptor is in BX (BL) register
40851                                  <1>       ;     * 2nd argument, offset is in CX register
40852                                  <1>       ;     * 3rd argument, ptrname/switch is in DX (DL) register
40853                                  <1>
40854 0000DE80 E821000000              <1>       call  seektell
40855                                  <1>       ; EAX = Current R/W pointer of the file
40856                                  <1>       ; EBX = [u.fofp]
40857                                  <1>       ; [u.base] = offset (ECX input)
40858                                  <1>
40859 0000DE85 0305[84030300]          <1>       add   eax, [u.base]
40860 0000DE8B 8903                    <1>       mov   [ebx], eax
40861 0000DE8D E94BE6FFFF              <1>       jmp   sysret
40862                                  <1>
40863                                  <1> systell: ; / get the r/w pointer
40864                                  <1>       ; 06/11/2016 - TRDOS 386 (TRDOS v2.0) - temporary !-
40865                                  <1>       ; 22/06/2015 (Retro UNIX 386 v1 - Beginning)
40866                                  <1>       ; 07/07/2013 - 05/08/2013 (Retro UNIX 8086 v1)
40867                                  <1>       ;
40868                                  <1>       ; Retro UNIX 8086 v1 modification:
40869                                  <1>       ; ! 'systell' does not work in original UNIX v1,
40870                                  <1>       ;         it returns with error !
40871                                  <1>       ; Inputs: r0 - file descriptor
40872                                  <1>       ; Outputs: r0 - file r/w pointer
40873                                  <1>
40874                                  <1>       ;xor  ecx, ecx ; 0
40875 0000DE92 BA01000000              <1>       mov   edx, 1 ; 05/08/2013
40876                                  <1>       ;call seektell
40877 0000DE97 E810000000              <1>       call  seektell0 ; 05/08/2013
40878                                  <1>       ;; 06/11/2016
40879                                  <1>       ;; mov eax, [ebx]
40880 0000DE9C A3[64030300]            <1>       mov   [u.r0], eax
40881 0000DEA1 E937E6FFFF              <1>       jmp   sysret
40882                                  <1>
40883                                  <1> ; Original unix v1 'systell' system call:
```

```
40884                                  <1>             ; jsr r0,seektell
40885                                  <1>             ; br error4
40886                                  <1>
40887                                  <1> seektell:
40888                                  <1>       ; 06/11/2016 - TRDOS 386 (TRDOS v2.0)
40889                                  <1>       ; 03/01/2016
40890                                  <1>       ; 22/06/2015 (Retro UNIX 386 v1 - Beginning)
40891                                  <1>       ; 07/07/2013 - 05/08/2013 (Retro UNIX 8086 v1)
40892                                  <1>       ;
40893                                  <1>       ; 'seektell' puts the arguments from sysseek and systell
40894                                  <1>       ; call in u.base and u.count. It then gets the i-number of
40895                                  <1>       ; the file from the file descriptor in u.r0 and by calling
40896                                  <1>       ; getf. The i-node is brought into core and then u.count
40897                                  <1>       ; is checked to see it is a 0, 1, or 2.
40898                                  <1>       ; If it is 0 - u.count stays the same
40899                                  <1>       ;          1 - u.count = offset (u.fofp)
40900                                  <1>       ;         2 - u.count = i.size (size of file)
40901                                  <1>       ;
40902                                  <1>       ; !! Retro UNIX 8086 v1 modification:
40903                                  <1>       ;     Argument 1, file descriptor is in BX;
40904                                  <1>       ;     Argument 2, offset is in CX;
40905                                  <1>       ;     Argument 3, ptrname/switch is in DX register.
40906                                  <1>       ;
40907                                  <1>       ; ((Return -> eax = base for offset (position= base+offset))
40908                                  <1>       ;
40909 0000DEA6 890D[84030300]          <1>       mov    [u.base], ecx ; offset
40910                                  <1> seektell0:
40911 0000DEAC 8915[88030300]          <1>       mov    [u.count], edx
40912                                  <1>       ; EBX = file descriptor (file number)
40913 0000DEB2 E898FCFFFF              <1>       call   getf1
40914                                  <1>       ; EAX = First cluster of the file
40915                                  <1>       ; EBX = File number (Open file number)
40916                                  <1>       ; [u.fofp] = Pointer to File pointer
40917                                  <1>       ; [i.size] = File size
40918                                  <1>
40919 0000DEB7 09C0                    <1>       or     eax, eax
40920 0000DEB9 7514                    <1>       jnz    short seektell1
40921                                  <1>
40922 0000DEBB B80A000000              <1>       mov    eax, ERR_FILE_NOT_OPEN
40923 0000DEC0 A3[64030300]            <1>       mov    [u.r0], eax
40924 0000DEC5 A3[C8030300]            <1>       mov    dword [u.error], eax ; 'file not open !'
40925 0000DECA E9EEE5FFFF              <1>       jmp    error
40926                                  <1>
40927                                  <1> seektell1:
40928 0000DECF 8B1D[74030300]          <1>         mov     ebx, [u.fofp]
40929 0000DED5 803D[88030300]01        <1>       cmp    byte [u.count], 1
40930 0000DEDC 7705                    <1>       ja     short seektell2
40931 0000DEDE 7409                    <1>       je     short seektell3
40932 0000DEE0 31C0                    <1>       xor    eax, eax
40933 0000DEE2 C3                      <1>       retn
40934                                  <1>
40935                                  <1> seektell2:
40936 0000DEE3 A1[55040300]            <1>         mov        eax, [i.size]
40937 0000DEE8 C3                      <1>       retn
40938                                  <1>
40939                                  <1> seektell3:
40940 0000DEE9 8B03                    <1>       mov    eax, [ebx]
40941 0000DEEB C3                      <1>       retn
40942                                  <1>
40943                                  <1> sysintr: ; / set interrupt handling
40944                                  <1>       ; 22/06/2015 (Retro UNIX 386 v1 - Beginning)
40945                                  <1>       ; 07/07/2013 (Retro UNIX 8086 v1)
40946                                  <1>       ;
40947                                  <1>       ; 'sysintr' sets the interrupt handling value. It puts
40948                                  <1>       ; argument of its call in u.intr then branches into 'sysquit'
40949                                  <1>       ; routine. u.tty is checked if to see if a control tty exists.
40950                                  <1>       ; If one does the interrupt character in the tty buffer is
40951                                  <1>       ; cleared and 'sysret'is called. If one does not exits
40952                                  <1>       ; 'sysret' is just called.
40953                                  <1>       ;
40954                                  <1>       ; Calling sequence:
40955                                  <1>       ;     sysintr; arg
40956                                  <1>       ; Argument:
40957                                  <1>       ;     arg - if 0, interrupts (ASCII DELETE) are ignored.
40958                                  <1>       ;         - if 1, intterupts cause their normal result
40959                                  <1>       ;           i.e force an exit.
40960                                  <1>       ;         - if arg is a location within the program,
40961                                  <1>       ;           control is passed to that location when
40962                                  <1>       ;           an interrupt occurs.
40963                                  <1>       ; Inputs: -
40964                                  <1>       ; Outputs: -
40965                                  <1>       ; ..........................................................
40966                                  <1>       ;
40967                                  <1>       ; Retro UNIX 8086 v1 modification:
40968                                  <1>       ;      'sysintr' system call sets u.intr to value of BX
40969                                  <1>       ;     then branches into sysquit.
40970                                  <1>       ;
40971 0000DEEC 66891D[AA030300]        <1>       mov    [u.intr], bx
40972                                  <1>             ; jsr r0,arg; u.intr / put the argument in u.intr
40973                                  <1>             ; br 1f / go into quit routine
40974 0000DEF3 E9E5E5FFFF              <1>       jmp    sysret
40975                                  <1>
40976                                  <1> sysquit:
40977                                  <1>       ; 22/06/2015 (Retro UNIX 386 v1 - Beginning)
40978                                  <1>       ; 07/07/2013 (Retro UNIX 8086 v1)
40979                                  <1>       ;
40980                                  <1>       ; 'sysquit' turns off the quit signal. it puts the argument of
40981                                  <1>       ; the call in u.quit. u.tty is checked if to see if a control
40982                                  <1>       ; tty exists. If one does the interrupt character in the tty
40983                                  <1>       ; buffer is cleared and 'sysret'is called. If one does not exits
40984                                  <1>       ; 'sysret' is just called.
40985                                  <1>       ;
40986                                  <1>       ; Calling sequence:
```

```
                                    <1>       ;     sysquit; arg
                                    <1>       ; Argument:
                                    <1>       ;     arg - if 0, this call diables quit signals from the
                                    <1>       ;           typewriter (ASCII FS)
                                    <1>       ;         - if 1, quits are re-enabled and cause execution to
                                    <1>       ;           cease and a core image to be produced.
                                    <1>       ;            i.e force an exit.
                                    <1>       ;         - if arg is an addres in the program,
                                    <1>       ;           a quit causes control to sent to that
                                    <1>       ;           location.
                                    <1>       ; Inputs: -
                                    <1>       ; Outputs: -
                                    <1>       ; ..........................................................
                                    <1>       ;
                                    <1>       ; Retro UNIX 8086 v1 modification:
                                    <1>       ;       'sysquit' system call sets u.quit to value of BX
                                    <1>       ;     then branches into 'sysret'.
                                    <1>       ;
0000DEF8 66891D[AC030300]           <1>       mov   [u.quit], bx
0000DEFF E9D9E5FFFF                 <1>       jmp   sysret
                                    <1>             ; jsr r0,arg; u.quit / put argument in u.quit
                                    <1>       ;1:
                                    <1>             ; mov u.ttyp,r1 / move pointer to control tty buffer
                                    <1>                         ; / to r1
                                    <1>             ; beq sysret4 / return to user
                                    <1>             ; clrb 6(r1) / clear the interrupt character
                                    <1>                       ; / in the tty buffer
                                    <1>             ; br sysret4 / return to user
                                    <1>
                                    <1> syssetuid: ; / set process id
                                    <1>       ; 22/06/2015 (Retro UNIX 386 v1 - Beginning)
                                    <1>       ; 07/07/2013 - 02/08/2013 (Retro UNIX 8086 v1)
                                    <1>       ;
                                    <1>       ; 'syssetuid' sets the user id (u.uid) of the current process
                                    <1>       ; to the process id in (u.r0). Both the effective user and
                                    <1>       ; u.uid and the real user u.ruid are set to this.
                                    <1>       ; Only the super user can make this call.
                                    <1>       ;
                                    <1>       ; Calling sequence:
                                    <1>       ;     syssetuid
                                    <1>       ; Arguments: -
                                    <1>       ;
                                    <1>       ; Inputs: (u.r0) - contains the process id.
                                    <1>       ; Outputs: -
                                    <1>       ; ..........................................................
                                    <1>       ;
                                    <1>       ; Retro UNIX 8086 v1 modification:
                                    <1>       ;       BL contains the (new) user ID of the current process
                                    <1>
                                    <1>             ; movb *u.r0,r1 / move process id (number) to r1
0000DF04 3A1D[B1030300]             <1>       cmp   bl, [u.ruid]
                                    <1>             ; cmpb r1,u.ruid / is it equal to the real user
                                    <1>                           ; / id number
0000DF0A 741E                       <1>       je    short setuid1
                                    <1>             ; beq 1f / yes
0000DF0C 803D[B0030300]00           <1>       cmp   byte [u.uid], 0 ; 02/08/2013
                                    <1>             ; tstb u.uid / no, is current user the super user?
                                    <1>       ;ja   error
                                    <1>             ; bne error4 / no, error
0000DF13 760F                       <1>       jna   short setuid0
0000DF15 C705[C8030300]0B00-        <1>       mov   dword [u.error], ERR_NOT_SUPERUSER  ; 11
0000DF1D 0000                       <1>
                                    <1>                           ;  'permission denied !' error
0000DF1F E999E5FFFF                 <1>       jmp   error
                                    <1> setuid0:
0000DF24 881D[B1030300]             <1>       mov   [u.ruid], bl
                                    <1> setuid1: ; 1:
0000DF2A 881D[B0030300]             <1>       mov   [u.uid], bl ; 02/08/2013
                                    <1>             ; movb r1,u.uid / put process id in u.uid
                                    <1>             ; movb r1,u.ruid / put process id in u.ruid
0000DF30 E9A8E5FFFF                 <1>       jmp   sysret
                                    <1>             ; br sysret4 / system return
                                    <1>
                                    <1> sysgetuid: ; < get user id >
                                    <1>       ; 22/06/2015 (Retro UNIX 386 v1 - Beginning)
                                    <1>       ; 07/07/2013 (Retro UNIX 8086 v1)
                                    <1>       ;
                                    <1>       ; 'sysgetuid' returns the real user ID of the current process.
                                    <1>       ; The real user ID identifies the person who is logged in,
                                    <1>       ; in contradistinction to the effective user ID, which
                                    <1>       ; determines his access permission at each moment. It is thus
                                    <1>       ; useful to programs which operate using the 'set user ID'
                                    <1>       ; mode, to find out who invoked them.
                                    <1>       ;
                                    <1>       ; Calling sequence:
                                    <1>       ;     syssetuid
                                    <1>       ; Arguments: -
                                    <1>       ;
                                    <1>       ; Inputs: -
                                    <1>       ; Outputs: (u.r0) - contains the real user's id.
                                    <1>       ; ..........................................................
                                    <1>       ;
                                    <1>       ; Retro UNIX 8086 v1 modification:
                                    <1>       ;       AL contains the real user ID at return.
                                    <1>       ;
0000DF35 0FB605[B1030300]           <1>       movzx  eax, byte [u.ruid]
0000DF3C A3[64030300]               <1>       mov   [u.r0], eax
                                    <1>             ; movb u.ruid,*u.r0 / move the real user id to (u.r0)
0000DF41 E997E5FFFF                 <1>       jmp   sysret
                                    <1>             ; br sysret4 / systerm return, sysret
                                    <1>
                                    <1> anyi:
                                    <1>       ; 06/10/2016 (TRDOS 386 = TRDOS v2.0)
```

```
41090                                  <1>       ; Major Modification!
41091                                  <1>       ; TRDOS 386 does not permit to delete a file while it is open
41092                                  <1>       ; The role of 'anyi' procedure has beeen changed to ensure that.
41093                                  <1>       ;
41094                                  <1>       ; 22/06/2015 (Retro UNIX 386 v1 - Beginning)
41095                                  <1>       ; 25/04/2013 (Retro UNIX 8086 v1)
41096                                  <1>       ;
41097                                  <1>       ; 'anyi' is called if a file deleted while open.
41098                                  <1>       ; "anyi" checks to see if someone else has opened this file.
41099                                  <1>       ;
41100                                  <1>       ; INPUTS ->
41101                                  <1>       ;    r1 - contains an i-number
41102                                  <1>       ;    fsp - start of table containing open files
41103                                  <1>       ;
41104                                  <1>       ; OUTPUTS ->
41105                                  <1>       ;    "deleted" flag set in fsp entry of another occurrence of
41106                                  <1>       ;        this file and r2 points 1st word of this fsp entry.
41107                                  <1>       ;    if file not found - bit in i-node map is cleared
41108                                  <1>       ;                    (i-node is freed)
41109                                  <1>       ;             all blocks related to i-node are freed
41110                                  <1>       ;            all flags in i-node are cleared
41111                                  <1>       ; ((AX = R1)) input
41112                                  <1>       ;
41113                                  <1>       ;    (Retro UNIX Prototype : 02/12/2012, UNIXCOPY.ASM)
41114                                  <1>         ;    ((Modified registers: eDX, eCX, eBX, eSI, eDI, eBP))
41115                                  <1>       ;
41116                                  <1>       ; / r1 contains an i-number
41117                                  <1>
41118                                  <1>       ; TRDOS 386 (06/10/2016)
41119                                  <1>       ;
41120                                  <1>       ; INPUT:
41121                                  <1>       ;     EAX = First Cluster
41122                                  <1>       ;      DL = Logical DOS Drive Number
41123                                  <1>       ;
41124                                  <1>       ; OUTPUT:
41125                                  <1>       ;     CF = 1 -> EBX = File Handle/Number/Index
41126                                  <1>       ;     CF = 0 -> EBX = 0
41127                                  <1>       ;
41128                                  <1>       ; Modified Registers: EBX
41129                                  <1>
41130 0000DF46 31DB                    <1>       xor    ebx, ebx
41131                                  <1> anyi_0:
41132 0000DF48 80BB[4A630100]00        <1>       cmp    byte [ebx+OF_MODE], 0 ; 0 = empty entry
41133 0000DF4F 770A                    <1>       ja     short anyi_2 ; 1 (r), 2 (w) or 3 (r&w)
41134                                  <1> anyi_1:
41135 0000DF51 FEC3                    <1>       inc    bl
41136 0000DF53 80FB0A                  <1>       cmp    bl, OPENFILES ; max. count of open files
41137 0000DF56 72F0                    <1>       jb     short anyi_0
41138 0000DF58 31C0                    <1>       xor    eax, eax
41139 0000DF5A C3                      <1>       retn
41140                                  <1> anyi_2:
41141 0000DF5B 3A93[40630100]          <1>       cmp    dl, [ebx+OF_DRIVE]
41142 0000DF61 75EE                    <1>       jne    short anyi_1
41143 0000DF63 66C1E302                <1>       shl    bx, 2 ; *4 (dword offset)
41144 0000DF67 3B83[18630100]          <1>       cmp    eax, [ebx+OF_FCLUSTER]
41145 0000DF6D 7406                    <1>       je     short anyi_3
41146 0000DF6F 66C1EB02                <1>       shr    bx, 2 ; /4 (byte offset)
41147 0000DF73 EBDC                    <1>       jmp    short anyi_1
41148                                  <1> anyi_3:
41149 0000DF75 66C1EB02                <1>       shr    bx, 2 ; /4 (bytes offset) (index)
41150 0000DF79 F9                      <1>       stc
41151 0000DF7A C3                      <1>       retn
41152                                  <1>
41153                                  <1> ; Retro UNIX 386 v1 Kernel (v0.2) - u7.s
41154                                  <1> ; Last Modification: 14/11/2015
41155                                  <1>
41156                                  <1> sysmount: ; / mount file system
41157                                  <1>       ; 24/10/2016 - TRDOS 386 (TRDOS v2.0)
41158                                  <1>       ; temporary !
41159 0000DF7B B80F000000              <1>       mov    eax, ERR_DRV_NOT_RDY ; drive not ready !
41160 0000DF80 A3[C8030300]            <1>         mov     [u.error], eax
41161 0000DF85 A3[64030300]            <1>         mov     [u.r0], eax
41162 0000DF8A E92EE5FFFF              <1>       jmp    error
41163                                  <1>
41164                                  <1> sysumount: ; / special dismount file system
41165                                  <1>       ; 24/10/2016 - TRDOS 386 (TRDOS v2.0)
41166                                  <1>       ; temporary !
41167 0000DF8F B80F000000              <1>       mov    eax, ERR_DRV_NOT_RDY ; drive not ready !
41168 0000DF94 A3[C8030300]            <1>         mov     [u.error], eax
41169 0000DF99 A3[64030300]            <1>         mov     [u.r0], eax
41170 0000DF9E E91AE5FFFF              <1>       jmp    error
41171                                  <1>
41172                                  <1> ; Retro UNIX 386 v1 Kernel (v0.2) - SYS9.INC
41173                                  <1> ; Last Modification: 09/12/2015
41174                                  <1>
41175                                  <1> syssleep:
41176                                  <1>       ; 29/06/2015 - (Retro UNIX 386 v1)
41177                                  <1>       ; 11/06/2014 - (Retro UNIX 8086 v1)
41178                                  <1>       ;
41179                                  <1>       ; Retro UNIX 8086 v1 feature only
41180                                  <1>       ; (INPUT -> none)
41181                                  <1>       ;
41182 0000DFA3 0FB61D[B3030300]        <1>       movzx  ebx, byte [u.uno] ; process number
41183 0000DFAA 8AA3[7F000300]          <1>       mov    ah, [ebx+p.ttyc-1] ; current/console tty
41184 0000DFB0 E812120000              <1>       call   sleep
41185 0000DFB5 E923E5FFFF              <1>       jmp    sysret
41186                                  <1>
41187                                  <1> _vp_clr:
41188                                  <1>       ; Reset/Clear Video Page
41189                                  <1>       ;
41190                                  <1>       ; 30/06/2015 - (Retro UNIX 386 v1)
41191                                  <1>       ; 21/05/2013 - 30/10/2013(Retro UNIX 8086 v1) (U0.ASM)
41192                                  <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
0000DFBA 28C0                   <1>       sub   al, al
                                <1>       ; al = 0 (clear video page)
                                <1>       ; bh = video page ; 13/05/2016
0000DFBC B407                   <1>       mov   ah, 07h
                                <1>       ; ah = 7 (attribute/color)
0000DFBE 6631C9                 <1>       xor   cx, cx ; 0, left upper column (cl) & row (cl)
0000DFC1 66BA4F18               <1>       mov   dx, 184Fh ; right lower column & row (dl=24, dh=79)
0000DFC5 E8403AFFFF             <1>       call  _scroll_up
                                <1>       ; bh = video page
0000DFCA 6631D2                 <1>       xor   dx, dx ; 0 (cursor position)
0000DFCD E9763DFFFF             <1>       jmp   _set_cpos
                                <1>
                                <1> sysmsg:
                                <1>       ; 13/05/2016
                                <1>       ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
                                <1>       ; 01/07/2015 - 11/11/2015 (Retro UNIX 386 v1)
                                <1>       ; Print user-application message on user's console tty
                                <1>       ;
                                <1>       ; Input -> EBX = Message address
                                <1>       ;         ECX = Message length (max. 255)
                                <1>       ;         DL = Color (IBM PC Rombios color attributes)
                                <1>       ;
0000DFD2 81F9FF000000           <1>       cmp   ecx, MAX_MSG_LEN ; 255
0000DFD8 0F87FFE4FFFF           <1>       ja    sysret ; nothing to do with big message size
0000DFDE 08C9                   <1>       or    cl, cl
0000DFE0 0F84F7E4FFFF           <1>       jz    sysret
0000DFE6 20D2                   <1>       and   dl, dl
0000DFE8 7502                   <1>       jnz   short sysmsg0
0000DFEA B207                   <1>       mov   dl, 07h ; default color
                                <1>             ; (black background, light gray character)
                                <1> sysmsg0:
0000DFEC 891D[84030300]         <1>       mov   [u.base], ebx
0000DFF2 8815[4F520100]         <1>       mov   [ccolor], dl ; color attributes
0000DFF8 89E5                   <1>       mov   ebp, esp
0000DFFA 31DB                   <1>       xor   ebx, ebx ; 0
0000DFFC 891D[8C030300]         <1>       mov   [u.nread], ebx ; 0
                                <1>       ;
0000E002 381D[C6030300]         <1>       cmp   [u.kcall], bl ; 0
0000E008 7769                   <1>       ja    short sysmsgk ; Temporary (01/07/2015)
                                <1>       ;
0000E00A 890D[88030300]         <1>       mov   [u.count], ecx
0000E010 41                     <1>       inc   ecx ; + 00h ; ASCIIZ
0000E011 29CC                   <1>       sub   esp, ecx
0000E013 89E7                   <1>       mov   edi, esp
0000E015 89E6                   <1>       mov   esi, esp
0000E017 66891D[C4030300]       <1>       mov   [u.pcount], bx ; reset page (phy. addr.) counter
                                <1>       ; 11/11/2015
0000E01E 8A25[94030300]         <1>       mov   ah, [u.ttyp] ; recent open tty
                                <1>       ; 0 = none
0000E024 FECC                   <1>       dec   ah
0000E026 790C                   <1>       jns   short sysmsg1
0000E028 8A1D[B3030300]         <1>       mov   bl, [u.uno] ; process number
0000E02E 8AA3[7F000300]         <1>       mov   ah, [ebx+p.ttyc-1] ; user's (process's) console tty
                                <1> sysmsg1:
0000E034 8825[96030300]         <1>       mov   [u.ttyn], ah
                                <1> sysmsg2:
0000E03A E80E080000             <1>       call  cpass
0000E03F 7416                   <1>       jz    short sysmsg5
0000E041 AA                     <1>       stosb
0000E042 20C0                   <1>       and   al, al
0000E044 75F4                   <1>       jnz   short sysmsg2
                                <1> sysmsg3:
0000E046 80FC07                 <1>       cmp   ah, 7 ; tty number
0000E049 7711                   <1>       ja    short sysmsg6 ; serial port
0000E04B E83E000000             <1>       call  print_cmsg
                                <1> sysmsg4:
0000E050 89EC                   <1>       mov   esp, ebp
0000E052 E986E4FFFF             <1>       jmp   sysret
                                <1> sysmsg5:
0000E057 C60700                 <1>       mov   byte [edi], 0
0000E05A EBEA                   <1>       jmp   short sysmsg3
                                <1> sysmsg6:
0000E05C 8A06                   <1>       mov   al, [esi]
0000E05E E861110000             <1>       call  sndc
0000E063 72EB                   <1>       jc    short sysmsg4
0000E065 803E00                 <1>       cmp   byte [esi], 0  ; 0 is stop character
0000E068 76E6                   <1>       jna   short sysmsg4
0000E06A 46                     <1>       inc   esi
0000E06B 8A25[96030300]         <1>       mov   ah, [u.ttyn]
0000E071 EBE9                   <1>       jmp   short sysmsg6
                                <1>
                                <1> sysmsgk: ; Temporary (01/07/2015)
                                <1>       ; The message has been sent by Kernel (ASCIIZ string)
                                <1>       ; (ECX -character count- will not be considered)
0000E073 8B35[84030300]         <1>       mov   esi, [u.base]
0000E079 8A25[4E520100]         <1>       mov   ah, [ptty] ; present/current screen (video page)
0000E07F 8825[96030300]         <1>       mov   [u.ttyn], ah
0000E085 C605[C6030300]00       <1>       mov   byte [u.kcall], 0
0000E08C EBB8                   <1>       jmp   short sysmsg3
                                <1>
                                <1> print_cmsg:
                                <1>       ; 13/05/2016 - TRDOS 386 (TRDOS v2.0)
                                <1>       ; 01/07/2015 (Retro UNIX 386 v1)
```

```
41296                                  <1>       ;
41297                                  <1>       ; print message (on user's console tty)
41298                                  <1>       ;     with requested color
41299                                  <1>       ;
41300                                  <1>       ; INPUTS:
41301                                  <1>       ;     esi = message address
41302                                  <1>       ;     [u.ttyn] = tty number (0 to 7)
41303                                  <1>       ;     [ccolor] = color attributes (IBM PC BIOS colors)
41304                                  <1>
41305 0000E08E 8A3D[96030300]          <1>       mov   bh, [u.ttyn]
41306                                  <1>       ;mov   bh, ah
41307                                  <1>
41308 0000E094 AC                      <1>       lodsb
41309                                  <1> pcmsg1:
41310 0000E095 56                      <1>       push  esi
41311 0000E096 8A1D[4F520100]          <1>       mov   bl, [ccolor]
41312                                  <1>       ;mov   bh, [u.ttyn]
41313 0000E09C E8113CFFFF              <1>       call  _write_tty
41314 0000E0A1 5E                      <1>       pop   esi
41315 0000E0A2 AC                      <1>       lodsb
41316 0000E0A3 20C0                    <1>       and   al, al  ; 0
41317 0000E0A5 75EE                    <1>       jnz   short pcmsg1
41318 0000E0A7 C3                      <1>       retn
41319                                  <1>
41320                                  <1> sysgeterr:
41321                                  <1>       ; 09/12/2015
41322                                  <1>       ; 21/09/2015 - (Retro UNIX 386 v1 feature only!)
41323                                  <1>       ; Get last error number or page fault count
41324                                  <1>       ; (for debugging)
41325                                  <1>       ;
41326                                  <1>       ; Input -> EBX = return type
41327                                  <1>       ;        0 = last error code (which is in 'u.error')
41328                                  <1>       ;        FFFFFFFFh = page fault count for running process
41329                                  <1>       ;        FFFFFFFEh = total page fault count
41330                                  <1>       ;        1 .. FFFFFFFDh = undefined
41331                                  <1>       ;
41332                                  <1>       ; Output -> EAX = last error number or page fault count
41333                                  <1>       ;        (depending on EBX input)
41334                                  <1>       ;
41335 0000E0A8 21DB                    <1>       and   ebx, ebx
41336 0000E0AA 750B                    <1>       jnz   short glerr_2
41337                                  <1> glerr_0:
41338 0000E0AC A1[C8030300]            <1>       mov   eax, [u.error]
41339                                  <1> glerr_1:
41340 0000E0B1 A3[64030300]            <1>       mov   [u.r0], eax
41341 0000E0B6 C3                      <1>       retn
41342                                  <1> glerr_2:
41343 0000E0B7 43                      <1>       inc   ebx ; FFFFFFFFh -> 0, FFFFFFFEh -> FFFFFFFFh
41344 0000E0B8 74FD                    <1>       jz    short glerr_2 ; page fault count for process
41345 0000E0BA 43                      <1>       inc   ebx ; FFFFFFFFh -> 0
41346 0000E0BB 75EF                    <1>       jnz   short glerr_0
41347 0000E0BD A1[80050300]            <1>       mov   eax, [PF_Count] ; total page fault count
41348 0000E0C2 EBED                    <1>         jmp     short glerr_1
41349                                  <1> glerr_3:
41350 0000E0C4 A1[CC030300]            <1>       mov   eax, [u.pfcount]
41351 0000E0C9 EBE6                    <1>       jmp   short glerr_1
41352                                  <1>
41353                                  <1> load_and_run_file:
41354                                  <1>       ; 22/01/2017
41355                                  <1>       ; 04/01/2017, 07/01/2017
41356                                  <1>       ; 24/10/2016
41357                                  <1>       ; 24/04/2016, 02/05/2016, 03/05/2016, 06/05/2016
41358                                  <1>       ; 23/04/2016 (TRDOS 386 = TRDOS v2.0)
41359                                  <1>       ; 23/10/2015 (Retro UNIX 386 v1, 'sysexec')
41360                                  <1>       ; 23/06/2015 (Retro UNIX 386 v1 - Beginning)
41361                                  <1>       ; 03/06/2013 - 06/12/2013 (Retro UNIX 8086 v1)
41362                                  <1>       ; EAX = First Cluster number
41363                                  <1>       ; EDX = File Size
41364                                  <1>       ; ESI = Argument list address
41365                                  <1>       ; [argc] = argument count
41366                                  <1>       ; [u.nread] = argument list length
41367                                  <1>       ; [esp] = return address to the caller (*)
41368                                  <1>       ;
41369 0000E0CB 8935[4C040300]          <1>       mov   [argv], esi
41370 0000E0D1 8915[55040300]          <1>       mov   [i.size], edx
41371 0000E0D7 A3[51040300]            <1>       mov   [ii], eax
41372                                  <1>
41373                                  <1>       ;sti   ; 07/01/2017
41374                                  <1>       ;mov   eax, [k_page_dir]
41375                                  <1>       ;mov   [u.pgdir], eax
41376 0000E0DC 31C0                    <1>       xor   eax, eax ; clc ; *** ; 04/01/2017
41377                                  <1>       ;mov   [u.r0], eax ; 0 ; 07/01/2017
41378                                  <1>
41379                                  <1>       ; 06/05/2016
41380                                  <1>       ; Set 'sysexit' return order to MainProg
41381                                  <1>       ;
41382 0000E0DE 58                      <1>       pop   eax ; * 'loc_load_and_run_file_8:' address
41383                                  <1>       ;; 22/01/2017
41384                                  <1>       ;;cli ; 07/01/2017
41385 0000E0DF 8B25[BC510100]          <1>       mov   esp, [tss.esp0]
41386                                  <1>       ;
41387                                  <1>       ; 'loc_load_run_file_8' address has
41388                                  <1>       ; 'jmp loc_file_rw_restore_retn' instruction
41389                                  <1>       ; 'loc_file_rw_restore_retn:' will return to
41390                                  <1>       ; [mainprog_return_addr]
41391                                  <1>       ; just after 'call command_interpreter'
41392                                  <1>       ;
41393 0000E0E5 68[3B630000]            <1>       push  _end_of_mainprog ; we must not return to here !
41394 0000E0EA FF35[A45F0100]          <1>       push  dword [mainprog_return_addr]
41395 0000E0F0 89E5                    <1>       mov   ebp, esp ; **
41396                                  <1>       ;
41397 0000E0F2 9C                      <1>       pushfd  ; EFLAGS      ; IRETD ; ***
41398 0000E0F3 6A08                    <1>       push  KCODE ; cs    ; IRETD
```

```
41399 0000E0F5 50                  <1>       push   eax ; * (eip) ; IRETD
41400 0000E0F6 8925[5C030300]      <1>       mov    [u.sp], esp
41401                              <1>       ;mov   byte [u.quant], time_count
41402 0000E0FC 1E                  <1>       push   ds
41403 0000E0FD 06                  <1>       push   es
41404 0000E0FE 0FA0                <1>       push   fs
41405 0000E100 0FA8                <1>       push   gs
41406                              <1>       ;mov   eax, [u.r0]
41407 0000E102 29C0                <1>       sub    eax, eax
41408 0000E104 60                  <1>       pushad
41409 0000E105 68[DDC40000]        <1>       push   sysret
41410                              <1>       ;push  sysrel1 ; 07/01/2017
41411 0000E10A 8925[60030300]      <1>       mov    [u.usp], esp
41412                              <1>       ;
41413 0000E110 E83D060000          <1>       call   wswap ; Save MainProg (process 1) 'u' structure
41414                              <1>                    ; and registers for return (from program)
41415 0000E115 89EC                <1>       mov    esp, ebp ; **
41416                              <1>       ;;22/01/2017
41417                              <1>       ;;sti ; 07/01/2017
41418 0000E117 50                  <1>       push   eax  ; * 'loc_load_and_run_file_8:' address
41419                              <1>       ;
41420                              <1>       ;;; 02/05/2016
41421                              <1>       ;;; Create a new process (parent: MainProg)
41422 0000E118 31F6                <1>       xor    esi, esi
41423                              <1> cnpm_1: ; search p.stat table for unused process number
41424 0000E11A 46                  <1>       inc    esi
41425 0000E11B 80BE[AF000300]00    <1>       cmp    byte [esi+p.stat-1], 0 ; SFREE
41426                              <1>                           ; is process active, unused, dead
41427 0000E122 760B                <1>       jna    short cnpm_2 ; it's unused so branch
41428 0000E124 6683FE10            <1>       cmp    si, nproc    ; all processes checked
41429 0000E128 72F0                <1>       jb     short cnpm_1    ; no, branch back
41430 0000E12A E98282FFFF          <1>       jmp    panic
41431                              <1> cnpm_2:
41432 0000E12F A1[B8030300]        <1>       mov    eax, [u.pgdir] ; page directory of MainProg
41433 0000E134 A3[BC030300]        <1>       mov    [u.ppgdir], eax ; parent's page directory
41434 0000E139 E83C6AFFFF          <1>       call   allocate_page
41435 0000E13E 0F826D82FFFF        <1>       jc     panic
41436                              <1>       ; EAX = UPAGE (user structure page) address
41437 0000E144 A3[B4030300]        <1>       mov    [u.upage], eax ; memory page for 'user' struct (child)
41438 0000E149 89F7                <1>       mov    edi, esi
41439 0000E14B 66C1E702            <1>       shl    di, 2
41440 0000E14F 8987[BC000300]      <1>       mov    [edi+p.upage-4], eax ; memory page for 'user' struct
41441 0000E155 E89A6AFFFF          <1>       call   clear_page ; 03/05/2016
41442                              <1>       ;movzx eax, byte [p.ttyc] ; console tty (for MainProg)
41443 0000E15A 6629C0              <1>       sub    ax, ax ; 0
41444 0000E15D 668986[7F000300]    <1>       mov     [esi+p.ttyc-1], ax ; al - set child's console tty
41445                              <1>                               ; ah - reset child's wait channel
41446 0000E164 89F0                <1>       mov    eax, esi
41447 0000E166 A2[B3030300]        <1>       mov    [u.uno], al ; child process number
41448 0000E16B FE86[AF000300]      <1>         inc     byte [esi+p.stat-1] ; 1, SRUN
41449 0000E171 66D1E6              <1>       shl    si, 1 ; multiply si by 2 to get index into p.pid table
41450 0000E174 66FF05[4E030300]    <1>       inc    word [mpid] ; increment m.pid; get a new process name
41451 0000E17B 66A1[4E030300]      <1>       mov    ax, [mpid]
41452 0000E181 668986[1E000300]    <1>       mov    [esi+p.pid-2], ax ; put new process name
41453                              <1>                              ; in child process' name slot
41454                              <1>       ;mov   ax, [p.pid]  ; get process name of MainProg
41455 0000E188 66B80100            <1>       mov    ax, 1
41456 0000E18C 668986[3E000300]    <1>       mov    [esi+p.ppid-2], ax ; put parent process name
41457                              <1>                                 ; in parent process slot for child
41458 0000E193 6648                <1>       dec    ax ; 0
41459 0000E195 66A3[94030300]      <1>       mov    [u.ttyp], ax ; 0
41460                              <1>       ;;;
41461 0000E19B A1[51040300]        <1>       mov    eax,  [ii]
41462                              <1>       ; Retro UNIX 386 v1, 'sysexec' (u2.s)
41463 0000E1A0 E81C100000          <1>       call   iopen
41464                              <1>       ; 06/06/2016
41465 0000E1A5 C605[A9030300]01    <1>       mov    byte [u.pri], 1 ; normal priority
41466                              <1>       ;
41467 0000E1AC EB10                <1>       jmp    short sysexec_7 ; 02/05/2016
41468                              <1>
41469                              <1> sysexec_6:
41470                              <1>       ; 14/11/2017
41471                              <1>       ; 13/11/2017 (TRDOS 386)
41472 0000E1AE 8925[4C040300]      <1>       mov    [argv], esp ; *!* ; start address of argument list
41473                              <1>
41474                              <1>       ; 18/10/2015 (Retro UNIX 386 v1)
41475                              <1>       ; argument list transfer from user's core memory to
41476                              <1>       ; kernel stack frame is OK here.
41477                              <1>       ; [u.nread] = ; argument list length
41478                              <1>
41479                              <1>       ; 04/01/2017
41480                              <1>       ; 24/10/2016
41481                              <1>       ;;02/05/2016
41482                              <1>       ; 23/04/2016
41483                              <1>       ; 18/10/2015 ('sysexec_6')
41484                              <1>       ; 23/06/2015
41485 0000E1B4 A1[B8030300]        <1>       mov    eax, [u.pgdir] ; physical address of page directory
41486                              <1>       ;;cmp  eax, [k_page_dir] ; TRDOS MainProg ?
41487                              <1>       ;;je   short sysexec_7
41488 0000E1B9 E8F56AFFFF          <1>       call   deallocate_page_dir
41489                              <1> sysexec_7:
41490 0000E1BE E8256AFFFF          <1>       call   make_page_dir
41491 0000E1C3 0F82E881FFFF        <1>       jc     panic  ; allocation error
41492                              <1>                     ; after a deallocation would be nonsence !?
41493                              <1>       ; 24/07/2015
41494                              <1>       ; map kernel pages (1st 4MB) to PDE 0
41495                              <1>       ;     of the user's page directory
41496                              <1>       ;     (It is needed for interrupts!)
41497                              <1>       ; 18/10/2015
41498 0000E1C9 8B15[20520100]      <1>       mov    edx, [k_page_dir] ; Kernel's page directory
41499 0000E1CF 8B02                <1>       mov    eax, [edx] ; physical address of
41500                              <1>                       ; kernel's first page table (1st 4 MB)
41501                              <1>                       ; (PDE 0 of kernel's page directory)
```

```
41502 0000E1D1 8B15[B8030300]      <1>       mov    edx, [u.pgdir]
41503 0000E1D7 8902                <1>       mov    [edx], eax ; PDE 0 (1st 4MB)
41504                              <1>       ;
41505                              <1>       ; 20/07/2015
41506 0000E1D9 BB00004000          <1>       mov    ebx, CORE ; start address = 0 (virtual) + CORE
41507                              <1>       ; 18/10/2015
41508 0000E1DE BE[3C040300]        <1>       mov    esi, pcore ; physical start address
41509                              <1> sysexec_8:
41510 0000E1E3 B907000000          <1>       mov    ecx, PDE_A_USER + PDE_A_WRITE + PDE_A_PRESENT
41511 0000E1E8 E8196AFFFF          <1>       call   make_page_table
41512 0000E1ED 0F82BE81FFFF        <1>       jc     panic
41513                              <1>       ;mov   ecx, PTE_A_USER + PTE_A_WRITE + PTE_A_PRESENT
41514 0000E1F3 E81C6AFFFF          <1>       call   make_page ; make new page, clear and set the pte
41515 0000E1F8 0F82B381FFFF        <1>       jc     panic
41516                              <1>       ;
41517 0000E1FE 8906                <1>       mov    [esi], eax ; 24/06/2015
41518                              <1>       ; ebx = virtual address (24/07/2015)
41519 0000E200 E8B46FFFFF          <1>       call   add_to_swap_queue
41520                              <1>       ; 18/10/2015
41521 0000E205 81FE[40040300]      <1>       cmp    esi, ecore ; user's stack (last) page ?
41522 0000E20B 740C                <1>       je     short sysexec_9 ; yes
41523 0000E20D BE[40040300]        <1>       mov    esi, ecore  ; physical address of the last page
41524                              <1>       ; 20/07/2015
41525 0000E212 BB00F0FFFF          <1>       mov    ebx, (ECORE - PAGE_SIZE) + CORE
41526                              <1>       ; ebx = virtual end address + segment base address - 4K
41527 0000E217 EBCA                <1>         jmp     short sysexec_8
41528                              <1> sysexec_9:
41529                              <1>       ; 24/04/2016
41530                              <1>       ; 18/10/2015
41531                              <1>       ; 26/08/2015
41532                              <1>       ; 25/06/2015
41533                              <1>       ; move arguments from kernel stack to [ecore]
41534                              <1>       ; (argument list/line will be copied from kernel stack
41535                              <1>       ; frame to the last (stack) page of user's core memory)
41536                              <1>       ; 18/10/2015
41537 0000E219 8B3D[40040300]      <1>       mov    edi, [ecore]
41538 0000E21F 81C700100000        <1>       add    edi, PAGE_SIZE
41539 0000E225 0FB705[4A040300]    <1>       movzx  eax, word [argc]
41540 0000E22C 09C0                <1>       or     eax, eax
41541 0000E22E 7509                <1>       jnz    short sysexec_10
41542 0000E230 89FB                <1>       mov    ebx, edi
41543 0000E232 83EB04              <1>       sub    ebx, 4
41544 0000E235 8903                <1>       mov    [ebx], eax ; 0
41545 0000E237 EB45                <1>       jmp    short sysexec_13
41546                              <1> sysexec_10:
41547 0000E239 8B0D[8C030300]      <1>       mov    ecx, [u.nread]
41548                              <1>       ; 13/11/2017
41549                              <1>       ;mov   esi, TextBuffer ; 'load_and_execute_file'
41550                              <1>       ;mov   esi, esp     ; 'sysexec'
41551 0000E23F 8B35[4C040300]      <1>       mov    esi, [argv] ; 24/04/2016 (TRDOS 386  = TRDOS v2.0)
41552 0000E245 29CF                <1>       sub    edi, ecx ; page end address - argument list length
41553 0000E247 89C2                <1>       mov    edx, eax
41554 0000E249 FEC2                <1>       inc    dl ; argument count + 1 for argc value
41555 0000E24B C0E202              <1>       shl    dl, 2  ; 4 * (argument count + 1)
41556 0000E24E 89FB                <1>       mov    ebx, edi
41557 0000E250 80E3FC              <1>       and    bl, 0FCh ; 32 bit (dword) alignment
41558 0000E253 29D3                <1>       sub    ebx, edx
41559 0000E255 89FA                <1>       mov    edx, edi
41560 0000E257 F3A4                <1>       rep    movsb
41561 0000E259 89D6                <1>       mov    esi, edx
41562 0000E25B 89DF                <1>       mov    edi, ebx
41563 0000E25D BA00F0BFFF          <1>       mov    edx, ECORE - PAGE_SIZE ; virtual addr. of the last page
41564 0000E262 2B15[40040300]      <1>       sub    edx, [ecore] ; difference (virtual - physical)
41565 0000E268 AB                  <1>       stosd  ; eax = argument count
41566                              <1> sysexec_11:
41567 0000E269 89F0                <1>       mov    eax, esi
41568 0000E26B 01D0                <1>       add    eax, edx
41569 0000E26D AB                  <1>       stosd  ; eax = virtual address
41570                              <1>       ;dec   byte [argc]
41571 0000E26E 66FF0D[4A040300]    <1>       dec    word [argc] ; 14/11/2017
41572 0000E275 7407                <1>       jz     short sysexec_13
41573                              <1> sysexec_12:
41574 0000E277 AC                  <1>       lodsb
41575 0000E278 20C0                <1>       and    al, al
41576 0000E27A 75FB                <1>       jnz    short sysexec_12
41577 0000E27C EBEB                <1>       jmp    short sysexec_11
41578                              <1> sysexec_13:
41579                              <1>       ; 24/10/2016
41580                              <1>       ; 24/04/2016 - TRDOS 386 (TRDOS v2.0)
41581                              <1>       ; 23/06/2015 - 19/10/2015 (Retro UNIX 386 v1, 'sysexec_13')
41582                              <1>       ;
41583                              <1>       ; moving arguments to [ecore] is OK here..
41584                              <1>       ;
41585                              <1>       ; ebx = beginning addres of argument list pointers
41586                              <1>       ;      in user's stack
41587 0000E27E 2B1D[40040300]      <1>       sub    ebx, [ecore]
41588 0000E284 81C300F0BFFF        <1>       add     ebx, (ECORE - PAGE_SIZE)
41589                              <1>               ; end of core - 4096 (last page)
41590                              <1>               ; (virtual address)
41591 0000E28A 891D[4C040300]      <1>       mov    [argv], ebx
41592 0000E290 891D[90030300]      <1>       mov    [u.break], ebx ; available user memory
41593                              <1>       ;
41594 0000E296 29C0                <1>       sub    eax, eax
41595 0000E298 C705[88030300]2000- <1>       mov    dword [u.count], 32 ; Executable file header size
41596 0000E2A0 0000                <1>
41597 0000E2A2 C705[74030300]-     <1>       mov    dword [u.fofp], u.off
41598 0000E2A8 [80030300]          <1>
41599 0000E2AC A3[80030300]        <1>       mov    [u.off], eax ; 0
41600 0000E2B1 A3[84030300]        <1>       mov    [u.base], eax ; 0, start of user's core (virtual)
41601                              <1>       ; 24/10/2016
41602 0000E2B6 A0[E6520100]        <1>       mov    al, [Current_Drv]
41603 0000E2BB A2[46030300]        <1>       mov    [cdev], al
41604                              <1>       ;
```

```
41605 0000E2C0 A1[51040300]        <1>       mov    eax, [ii] ; Fist Cluster of the Program (PRG) file
41606                              <1>       ; EAX = First cluster of the executable file
41607 0000E2C5 E80A010000          <1>       call   readi
41608                              <1>
41609 0000E2CA 8B0D[90030300]      <1>       mov    ecx, [u.break] ; top of user's stack (physical addr.)
41610 0000E2D0 890D[88030300]      <1>       mov    [u.count], ecx ; save for overrun check
41611                              <1>       ;
41612 0000E2D6 8B0D[8C030300]      <1>       mov    ecx, [u.nread]
41613 0000E2DC 890D[90030300]      <1>       mov    [u.break], ecx ; virtual address (offset from start)
41614 0000E2E2 80F920              <1>       cmp    cl, 32
41615 0000E2E5 7540                <1>         jne     short sysexec_15
41616                              <1>       ;:
41617                              <1>       ; Retro UNIX 386 v1 (32 bit) executable file header format
41618 0000E2E7 8B35[3C040300]      <1>       mov    esi, [pcore] ; start address of user's core memory
41619                              <1>                           ; (phys. start addr. of the exec. file)
41620 0000E2ED AD                  <1>       lodsd
41621 0000E2EE 663DEB1E            <1>       cmp    ax, 1EEBh ; EBH, 1Eh -> jump to +32
41622 0000E2F2 7533                <1>       jne    short sysexec_15
41623 0000E2F4 AD                  <1>       lodsd
41624 0000E2F5 89C1                <1>       mov    ecx, eax ; text (code) section size
41625 0000E2F7 AD                  <1>       lodsd
41626 0000E2F8 01C1                <1>       add    ecx, eax ; + data section size (initialized data)
41627 0000E2FA 89CB                <1>       mov    ebx, ecx
41628 0000E2FC AD                  <1>       lodsd
41629 0000E2FD 01C3                <1>       add    ebx, eax ; + bss section size (for overrun checking)
41630 0000E2FF 3B1D[88030300]      <1>       cmp    ebx, [u.count]
41631 0000E305 7711                <1>       ja     short sysexec_14  ; program overruns stack !
41632                              <1>       ;
41633                              <1>       ; add bss section size to [u.break]
41634 0000E307 0105[90030300]      <1>       add    [u.break], eax
41635                              <1>       ;
41636 0000E30D 83E920              <1>       sub    ecx, 32  ; header size (already loaded)
41637                              <1>       ;cmp   ecx, [u.count]
41638                              <1>       ;jnb   short sysexec_16
41639 0000E310 890D[88030300]      <1>       mov    [u.count], ecx ; required read count
41640 0000E316 EB29                <1>       jmp    short sysexec_16
41641                              <1> sysexec_14:
41642                              <1>       ; insufficient (out of) memory
41643 0000E318 C705[C8030300]0400- <1>       mov    dword [u.error], ERR_MINOR_IM ; 1
41644 0000E320 0000                <1>
41645 0000E322 E996E1FFFF          <1>       jmp    error
41646                              <1> sysexec_15:
41647 0000E327 8B15[55040300]      <1>         mov  edx, [i.size] ; file size
41648 0000E32D 29CA                <1>       sub    edx, ecx ; file size - loaded bytes
41649 0000E32F 7626                <1>       jna    short sysexec_17 ; no need to next read
41650 0000E331 01D1                <1>       add    ecx, edx ; [i.size]
41651 0000E333 3B0D[88030300]      <1>       cmp    ecx, [u.count] ; overrun check (!)
41652 0000E339 77DD                <1>       ja     short sysexec_14
41653 0000E33B 8915[88030300]      <1>       mov    [u.count], edx
41654                              <1> sysexec_16:
41655 0000E341 A1[51040300]        <1>       mov    eax, [ii] ; first cluster
41656 0000E346 E889000000          <1>       call   readi
41657 0000E34B 8B0D[8C030300]      <1>       mov    ecx, [u.nread]
41658 0000E351 010D[90030300]      <1>       add    [u.break], ecx
41659                              <1> sysexec_17:
41660 0000E357 A1[51040300]        <1>       mov    eax, [ii] ; first cluster
41661 0000E35C E8610E0000          <1>       call   iclose
41662 0000E361 31C0                <1>       xor     eax, eax
41663 0000E363 FEC0                <1>       inc    al
41664 0000E365 66A3[AA030300]      <1>       mov    [u.intr], ax ; 1 (interrupt/time-out is enabled)
41665 0000E36B 66A3[AC030300]      <1>       mov    [u.quit], ax ; 1 ('crtl+brk' signal is enabled)
41666 0000E371 833D[BC030300]00    <1>         cmp  dword [u.ppgdir], 0  ; is the caller MainProg (kernel) ?
41667 0000E378 770C                <1>       ja     short sysexec_18 ; no, the caller is user process
41668                              <1>       ; If the caller is kernel (MainProg), 'sysexec' will come here
41669 0000E37A 8B15[20520100]      <1>       mov    edx, [k_page_dir] ; kernel's page directory
41670 0000E380 8915[BC030300]      <1>       mov    [u.ppgdir], edx ; next time 'sysexec' must not come here
41671                              <1> sysexec_18:
41672                              <1>       ; 13/11/2017
41673                              <1>       ; 02/05/2016
41674                              <1>       ; 24/04/2016 (TRDOS 386 = TRDOS v2.0)
41675                              <1>       ; 18/10/2015 (Retro UNIX 386 v1)
41676                              <1>       ; 05/08/2015
41677                              <1>       ; 29/07/2015
41678 0000E386 8B2D[4C040300]      <1>       mov    ebp, [argv] ; user's stack pointer must point to argument
41679                              <1>                          ; list pointers (argument count)
41680 0000E38C FA                  <1>       cli
41681 0000E38D 8B25[BC510100]      <1>         mov     esp, [tss.esp0] ; ring 0 (kernel) stack pointer
41682                              <1>       ;mov         esp, [u.sp] ; Restore Kernel stack
41683                              <1>                          ; for this process
41684                              <1>       ;add   esp, 20 ; --> EIP, CS, EFLAGS, ESP, SS
41685                              <1>       ;xor   eax, eax ; 0
41686 0000E393 FEC8                <1>       dec    al ; eax = 0
41687 0000E395 66BA2300            <1>       mov    dx, UDATA
41688 0000E399 6652                <1>       push   dx  ; user's stack segment
41689 0000E39B 55                  <1>       push   ebp ; user's stack pointer
41690                              <1>                  ; (points to number of arguments)
41691                              <1>
41692                              <1>       ; 04/01/2017
41693                              <1>       ; MainProg comes here while [sysflg]= 0FFh
41694                              <1>       ; (but sysexec comes here while [sysflg]= 0)
41695 0000E39C C605[5B030300]00    <1>       mov    byte [sysflg], 0 ; 04/01/2017
41696                              <1>                               ; (timer_int sysflg control)
41697 0000E3A3 FB                  <1>       sti
41698 0000E3A4 9C                  <1>       pushfd ; EFLAGS
41699                              <1>              ; Set IF for enabling interrupts in user mode
41700                              <1>       ;or    dword [esp], 200h
41701                              <1>       ;
41702                              <1>       ;mov   bx, UCODE
41703                              <1>       ;push  bx ; user's code segment
41704 0000E3A5 6A1B                <1>       push   UCODE
41705                              <1>       ;push  0
41706 0000E3A7 50                  <1>       push   eax ; EIP (=0) - start address -
41707 0000E3A8 8925[5C030300]      <1>       mov    [u.sp], esp ; 29/07/2015
```

```
41708                                  <1>       ; 05/08/2015
41709                                  <1>       ; Remedy of a General Protection Fault during 'iretd' is here !
41710                                  <1>       ; ('push dx' would cause to general protection fault,
41711                                  <1>       ; after 'pop ds' etc.)
41712                                  <1>       ;
41713                                  <1>       ;; push dx ; ds (UDATA)
41714                                  <1>       ;; push dx ; es (UDATA)
41715                                  <1>       ;; push dx ; fs (UDATA)
41716                                  <1>       ;; push dx ; gs (UDATA)
41717                                  <1>       ;
41718                                  <1>       ; This is a trick to prevent general protection fault
41719                                  <1>       ; during 'iretd' intruction at the end of 'sysrele' (in u1.s):
41720 0000E3AE 8EC2                   <1>       mov   es, dx ; UDATA
41721 0000E3B0 06                     <1>       push  es ; ds (UDATA)
41722 0000E3B1 06                     <1>       push  es ; es (UDATA)
41723 0000E3B2 06                     <1>       push  es ; fs (UDATA)
41724 0000E3B3 06                     <1>       push  es ; gs (UDATA)
41725 0000E3B4 66BA1000               <1>       mov   dx, KDATA
41726 0000E3B8 8EC2                   <1>       mov   es, dx
41727                                  <1>       ;
41728                                  <1>       ;; pushad simulation
41729 0000E3BA 89E5                   <1>       mov   ebp, esp ; esp before pushad
41730 0000E3BC 50                     <1>       push  eax ; eax (0)
41731 0000E3BD 50                     <1>       push  eax ; ecx (0)
41732 0000E3BE 50                     <1>       push  eax ; edx (0)
41733 0000E3BF 50                     <1>       push  eax ; ebx (0)
41734 0000E3C0 55                     <1>       push  ebp ; esp before pushad
41735 0000E3C1 50                     <1>       push  eax ; ebp (0)
41736 0000E3C2 50                     <1>       push  eax ; esi (0)
41737 0000E3C3 50                     <1>       push  eax ; edi (0)
41738                                  <1>       ;
41739 0000E3C4 A3[64030300]           <1>       mov   [u.r0], eax ; eax = 0
41740 0000E3C9 8925[60030300]         <1>       mov   [u.usp], esp
41741                                  <1>
41742                                  <1>       ; 14/11/2017
41743 0000E3CF E90BE1FFFF             <1>       jmp   sysret0
41744                                  <1>
41745                                  <1> ;     ; 02/05/2016
41746                                  <1> ;     ;inc  byte [sysflg] ; 0FFh -> 0
41747                                  <1> ;     ;mov  byte [sysflg], 0 ; 04/01/2017
41748                                  <1> ;     movzx ebx, byte [u.uno]
41749                                  <1> ;     shl   bl, 1 ; 13/11/2017
41750                                  <1> ;     cmp   word [ebx+p.ppid-2], 1 ; MainProg
41751                                  <1> ;     ja    sysret0 ; 03/05/2016
41752                                  <1> ;     push  sysret ; *
41753                                  <1> ;     mov   [u.usp], esp
41754                                  <1> ;     call  wswap ; save child process 'u' structure and
41755                                  <1> ;                 ; registers
41756                                  <1> ;     add   dword [u.usp], 4 ; 03/05/2016
41757                                  <1> ;sysexec_19: ; 02/05/2016
41758                                  <1> ;     retn ; * 'sysret' ; byte [sysflg] -> 0FFh
41759                                  <1>
41760                                  <1> readi:
41761                                  <1>       ; 01/05/2016
41762                                  <1>       ; 25/04/2016 - TRDOS 386 (TRDOS v2.0)
41763                                  <1>       ; 20/05/2015 - Retro UNIX 386 v1
41764                                  <1>       ; 11/03/2013 - 31/07/2013 (Retro UNIX 8086 v1)
41765                                  <1>       ;
41766                                  <1>       ; Reads from a file whose the first cluster number in EAX
41767                                  <1>       ;
41768                                  <1>       ; INPUTS ->
41769                                  <1>       ;    EAX - First cluster number of the file
41770                                  <1>       ;    u.count - byte count user desires
41771                                  <1>       ;    u.base - points to user buffer
41772                                  <1>       ;    u.fofp - points to dword with current file offset
41773                                  <1>       ;    i.size - file size
41774                                  <1>       ;    cdev - logical dos drive number of the file
41775                                  <1>       ; OUTPUTS ->
41776                                  <1>       ;    u.count - cleared
41777                                  <1>       ;    u.nread - accumulates total bytes passed back
41778                                  <1>       ;
41779                                  <1>       ; ((EAX)) input/output
41780                                  <1>       ; (Retro UNIX Prototype : 14/12/2012 - 01/03/2013, UNIXCOPY.ASM)
41781                                  <1>         ; ((Modified registers: edx, ebx, ecx, esi, edi))
41782                                  <1>
41783 0000E3D4 31D2                   <1>       xor   edx, edx ; 0
41784 0000E3D6 8915[8C030300]         <1>       mov   [u.nread], edx ; 0
41785 0000E3DC 668915[C4030300]       <1>       mov   [u.pcount], dx ; 19/05/2015
41786 0000E3E3 3915[88030300]         <1>       cmp   [u.count], edx ; 0
41787 0000E3E9 7701                   <1>       ja    short readi_1
41788 0000E3EB C3                     <1>       retn
41789                                  <1> readi_1:
41790                                  <1> dskr:
41791                                  <1>       ; 01/05/2016
41792                                  <1>       ; 25/04/2016 - TRDOS 386 (TRDOS v2.0)
41793                                  <1>       ; 24/05/2015 - 12/10/2015 (Retro UNIX 386 v1)
41794                                  <1>       ; 26/04/2013 - 03/08/2013 (Retro UNIX 8086 v1)
41795                                  <1> dskr_0:
41796 0000E3EC 8B15[55040300]         <1>         mov  edx, [i.size]
41797 0000E3F2 8B1D[74030300]         <1>       mov   ebx, [u.fofp]
41798 0000E3F8 2B13                   <1>       sub   edx, [ebx]
41799 0000E3FA 7647                   <1>       jna   short dskr_4
41800                                  <1>       ;
41801 0000E3FC 50                     <1>       push  eax ; 01/05/2016
41802 0000E3FD 3B15[88030300]         <1>       cmp    edx, [u.count]
41803 0000E403 7306                   <1>       jnb   short dskr_1
41804 0000E405 8915[88030300]         <1>       mov   [u.count], edx
41805                                  <1> dskr_1:
41806                                  <1>       ; EAX = First Cluster
41807                                  <1>       ; [Current_Drv] = Physical drive number
41808 0000E40B E83B000000             <1>       call  mget_r
41809                                  <1>       ; NOTE: in 'mget_r', relevant sector will be read in buffer
41810                                  <1>       ; if it is not already in buffer !
```

```
41811 0000E410 BB[8C050300]        <1>       mov    ebx, readi_buffer
41812 0000E415 803D[C6030300]00    <1>       cmp    byte [u.kcall], 0 ; the caller is 'namei' sign (=1)
41813 0000E41C 770F                <1>       ja     short dskr_3   ; zf=0 -> the caller is 'namei'
41814 0000E41E 66833D[C4030300]00  <1>       cmp    word [u.pcount], 0
41815 0000E426 7705                <1>       ja     short dskr_3
41816                              <1> dskr_2:
41817                              <1>       ; [u.base] = virtual address to transfer (as destination address)
41818 0000E428 E894010000          <1>       call   trans_addr_w ; translate virtual address to physical (w)
41819                              <1> dskr_3:
41820                              <1>       ; EBX (r5) = system (I/O) buffer address -physical-
41821 0000E42D E8F7010000          <1>       call   sioreg
41822 0000E432 87F7                <1>       xchg   esi, edi
41823                              <1>       ; EDI = file (user data) offset
41824                              <1>       ; ESI = sector (I/O) buffer offset
41825                              <1>       ; ECX = byte count
41826 0000E434 F3A4                <1>       rep    movsb
41827                              <1>       ; eax = remain bytes in buffer
41828                              <1>         ;       (check if remain bytes in the buffer > [u.pcount])
41829 0000E436 09C0                <1>       or     eax, eax
41830 0000E438 75EE                <1>       jnz    short dskr_2 ; (page end before system buffer end!)
41831 0000E43A 58                  <1>       pop    eax  ; (first cluster number)
41832 0000E43B 390D[88030300]      <1>       cmp    [u.count], ecx ; 0
41833 0000E441 77A9                <1>       ja     short dskr_0
41834                              <1> dskr_4:
41835 0000E443 C605[C6030300]00    <1>       mov    byte [u.kcall], 0
41836 0000E44A C3                  <1>       retn
41837                              <1>
41838                              <1> mget_r:
41839                              <1>       ; 24/10/2016
41840                              <1>       ; 22/10/2016
41841                              <1>       ; 12/10/2016
41842                              <1>       ; 29/04/2016
41843                              <1>       ; 25/04/2016 - TRDOS 386 (TRDOS v2.0)
41844                              <1>       ; 03/06/2015 (Retro UNIX 386 v1, 'mget', u.5s)
41845                              <1>       ; 22/03/2013 - 31/07/2013 (Retro UNIX 8086 v1)
41846                              <1>       ;
41847                              <1>       ; Get existing or (allocate) a new disk block for file
41848                              <1>       ;
41849                              <1>       ; INPUTS ->
41850                              <1>       ;    [u.fofp] = file offset pointer
41851                              <1>       ;    EAX = First Cluster
41852                              <1>       ;    [cdev] = Logical dos drive number
41853                              <1>       ;    ([u.off] = file offset)
41854                              <1>       ; OUTPUTS ->
41855                              <1>       ;    EAX = logical sector number
41856                              <1>       ;    ESI = Logical Dos Drive Description Table address
41857                              <1>       ;
41858                              <1>       ; Modified registers: EDX, EBX, ECX, ESI, EDI
41859                              <1>
41860 0000E44B 8B35[74030300]      <1>       mov     esi, [u.fofp]
41861 0000E451 8B1E                <1>       mov    ebx, [esi] ; (u.off)
41862                              <1>
41863 0000E453 29C9                <1>       sub    ecx, ecx
41864 0000E455 8A2D[46030300]      <1>       mov    ch, [cdev]
41865                              <1>
41866 0000E45B BE00010900          <1>       mov    esi, Logical_DOSDisks
41867 0000E460 01CE                <1>       add    esi, ecx
41868                              <1>
41869 0000E462 380D[585F0100]      <1>       cmp    [readi.valid], cl ; 0
41870 0000E468 7649                <1>       jna    short mget_r_0
41871                              <1>
41872 0000E46A 3A2D[595F0100]      <1>       cmp    ch, [readi.drv]
41873 0000E470 7541                <1>       jne    short mget_r_0
41874                              <1>
41875 0000E472 3B05[6C5F0100]      <1>       cmp    eax, [readi.fclust]
41876 0000E478 7565                <1>       jne    short mget_r_3
41877                              <1>
41878 0000E47A 89D8                <1>       mov    eax, ebx ; file offset
41879 0000E47C 668B0D[605F0100]    <1>       mov    cx, [readi.bpc]
41880 0000E483 41                  <1>       inc    ecx ; <= 65536
41881 0000E484 29D2                <1>       sub    edx, edx
41882 0000E486 F7F1                <1>       div    ecx
41883                              <1>
41884 0000E488 8B3D[685F0100]      <1>       mov    edi, [readi.c_index] ; cluster index
41885                              <1>
41886 0000E48E 39F8                <1>       cmp    eax, edi
41887 0000E490 757A                <1>         jne     short mget_r_4  ; (*)
41888                              <1>
41889                              <1>       ; edx = byte offset in cluster (<= 65535)
41890 0000E492 668915[625F0100]    <1>       mov    [readi.offset], dx
41891 0000E499 66C1EA09            <1>       shr    dx, 9 ; / 512
41892 0000E49D 8815[5B5F0100]      <1>       mov    [readi.s_index], dl ; sector index in cluster (0 to spc -1)
41893                              <1>
41894 0000E4A3 A1[645F0100]        <1>       mov    eax, [readi.cluster]  ; > 0 if [readi.valid] = 1
41895 0000E4A8 8B15[705F0100]      <1>       mov    edx, [readi.fs_index]
41896 0000E4AE E99A000000          <1>         jmp     mget_r_7
41897                              <1>
41898                              <1> mget_r_0:
41899 0000E4B3 882D[595F0100]      <1>       mov    [readi.drv], ch ; physical drive number
41900 0000E4B9 807E0300            <1>       cmp    byte [esi+LD_FATType], 0
41901 0000E4BD 7707                <1>       ja     short mget_r_1
41902 0000E4BF 8A4E12              <1>       mov    cl, [esi+LD_FS_BytesPerSec+1]
41903 0000E4C2 D0E9                <1>       shr    cl, 1 ;  ; 1 for 512 bytes, 4 for 2048 bytes
41904 0000E4C4 EB03                <1>       jmp    short mget_r_2
41905                              <1> mget_r_1:
41906 0000E4C6 8A4E13              <1>       mov    cl, [esi+LD_BPB+BPB_SecPerClust]
41907                              <1> mget_r_2:
41908 0000E4C9 880D[5A5F0100]      <1>       mov    [readi.spc], cl  ; sectors per cluster
41909                              <1>       ; NOTE: readi bytes per sector value is always 512 !
41910 0000E4CF 66C1E109            <1>       shl    cx, 9 ; * 512
41911 0000E4D3 6649                <1>       dec    cx ; bytes per cluster - 1
41912 0000E4D5 66890D[605F0100]    <1>       mov    [readi.bpc], cx
41913 0000E4DC 6629C9              <1>       sub    cx, cx
```

```
                                <1> mget_r_3:
0000E4DF A3[6C5F0100]           <1>       mov   [readi.fclust], eax ; first cluster (or FDT address)
0000E4E4 880D[585F0100]         <1>       mov   [readi.valid], cl ; 0
                                <1>       ;mov  [readi.s_index], cl ; 0
                                <1>       ;mov  [readi.offset], cx ; 0
0000E4EA 890D[685F0100]         <1>       mov   [readi.c_index], ecx ; 0
0000E4F0 890D[645F0100]         <1>       mov   [readi.cluster], ecx ; 0
0000E4F6 890D[5C5F0100]         <1>       mov   [readi.sector], ecx ; 0
                                <1>
0000E4FC 89D8                   <1>       mov   eax, ebx ; file offset
0000E4FE 668B0D[605F0100]       <1>       mov   cx, [readi.bpc]
0000E505 41                     <1>       inc   ecx ; <= 65536
0000E506 29D2                   <1>       sub   edx, edx
0000E508 F7F1                   <1>       div   ecx
                                <1>       ;mov  edi, [readi.c_index] ; previous cluster index
0000E50A 29FF                   <1>       sub   edi, edi
                                <1> mget_r_4:
0000E50C A3[685F0100]           <1>       mov   [readi.c_index], eax ; cluster index
                                <1>       ; edx = byte offset in cluster (<= 65535)
0000E511 668915[625F0100]       <1>       mov   [readi.offset], dx
0000E518 66C1EA09               <1>       shr   dx, 9 ; / 512
0000E51C 8815[5B5F0100]         <1>       mov   [readi.s_index], dl ; sector index in cluster (0 to spc -1)
                                <1>
0000E522 89C1                   <1>       mov   ecx, eax ; current cluster index
0000E524 A1[6C5F0100]           <1>       mov   eax, [readi.fclust]
0000E529 09C9                   <1>       or    ecx, ecx ; cluster index
0000E52B 741B                   <1>       jz    short mget_r_6
                                <1>
0000E52D 39CF                   <1>       cmp   edi, ecx
0000E52F 7710                   <1>       ja    short mget_r_5 ; old cluster index is higher
0000E531 8B15[645F0100]         <1>       mov   edx, [readi.cluster]
0000E537 21D2                   <1>       and   edx, edx
0000E539 7406                   <1>       jz    short mget_r_5
                                <1>       ; valid 'readi' parameters (*)
0000E53B 89D0                   <1>       mov   eax, edx
0000E53D 29F9                   <1>       sub   ecx, edi
0000E53F 740C                   <1>       jz    short mget_r_7
                                <1> mget_r_5:
                                <1>       ; EAX = Beginning cluster
                                <1>       ; EDX = Sector index in disk/file section
                                <1>       ;     (Only for SINGLIX file system!)
                                <1>       ; ECX = Cluster sequence number after the beginning cluster
                                <1>       ; ESI = Logical DOS Drive Description Table address
0000E541 E810DEFFFF             <1>       call  get_cluster_by_index
0000E546 724E                   <1>       jc    short mget_r_err
                                <1>       ; EAX = Cluster number
                                <1> mget_r_6:
0000E548 A3[645F0100]           <1>       mov   [readi.cluster], eax ; FDT number for Singlix File System
                                <1> mget_r_7:
0000E54D 807E0300               <1>       cmp   byte [esi+LD_FATType], 0
0000E551 765F                   <1>       jna   short mget_r_12
                                <1>
0000E553 83E802                 <1>       sub   eax, 2
0000E556 0FB615[5A5F0100]       <1>       movzx edx, byte [readi.spc]
0000E55D F7E2                   <1>       mul   edx
                                <1>
0000E55F 034668                 <1>       add   eax, [esi+LD_DATABegin]
0000E562 8A15[5B5F0100]         <1>       mov   dl, [readi.s_index]
0000E568 01D0                   <1>       add   eax, edx
                                <1> mget_r_8:
                                <1>       ; eax = logical sector number
0000E56A 803D[585F0100]00       <1>       cmp   byte [readi.valid], 0
0000E571 7608                   <1>       jna   short mget_r_9
0000E573 3B05[5C5F0100]         <1>       cmp   eax, [readi.sector]
0000E579 7436                   <1>       je    short mget_r_11 ; sector is already in 'readi' buffer
                                <1> mget_r_9:
0000E57B A3[5C5F0100]           <1>       mov   [readi.sector], eax
0000E580 BB[8C050300]           <1>       mov   ebx, readi_buffer ; buffer address
0000E585 B901000000             <1>       mov   ecx, 1
                                <1>       ; 29/04/2016
                                <1>       ;xor  dl, dl
                                <1>
                                <1>       ; EAX = Logical sector number
                                <1>       ; ECX = Sector count
                                <1>       ; EBX = Buffer address
                                <1>       ; (EDX = 0)
                                <1>       ; ESI = Logical DOS drive description table address
                                <1>
0000E58A E8490C0000             <1>       call  disk_read
0000E58F 7314                   <1>       jnc   short mget_r_10
                                <1>
                                <1>       ; 22/10/2016 (15h -> 17)
0000E591 B811000000             <1>       mov   eax, 17 ; Drive not ready or read error !
                                <1> mget_r_err:
0000E596 A3[C8030300]           <1>       mov   [u.error], eax
                                <1>       ; 12/10/2016
0000E59B A3[64030300]           <1>       mov   [u.r0], eax
0000E5A0 E918DFFFFF             <1>       jmp   error
                                <1> mget_r_10:
0000E5A5 C605[585F0100]01       <1>       mov   byte [readi.valid], 1 ; 24/10/2016
0000E5AC A1[5C5F0100]           <1>       mov   eax, [readi.sector]
                                <1> mget_r_11:
0000E5B1 C3                     <1>       retn
                                <1> mget_r_12:
                                <1>       ; EAX = FDT number
                                <1>       ; EDX = Sector index from FDT sector (0,1,2,3,4...)
0000E5B2 40                     <1>       inc   eax ; the first data sector in FS disk section
0000E5B3 8915[705F0100]         <1>       mov   [readi.fs_index], edx
0000E5B9 01D0                   <1>       add   eax, edx
0000E5BB EBAD                   <1>       jmp   short mget_r_8
                                <1>
                                <1> trans_addr_r:
                                <1>       ; 12/10/2016
```

```
42017                                  <1>       ; 02/05/2016 - TRDOS 386 (TRDOS v2.0)
42018                                  <1>       ; Translate virtual address to physical address
42019                                  <1>       ; for reading from user's memory space
42020                                  <1>       ; 04/06/2015 - 18/10/2015 (Retro UNIX 386 v1)
42021                                  <1>
42022 0000E5BD 31D2                    <1>       xor    edx, edx ; 0 (read access sign)
42023 0000E5BF EB04                    <1>       jmp    short trans_addr_rw
42024                                  <1>
42025                                  <1> trans_addr_w:
42026                                  <1>       ; 12/10/2016
42027                                  <1>       ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
42028                                  <1>       ; Translate virtual address to physical address
42029                                  <1>       ; for writing to user's memory space
42030                                  <1>       ; 04/06/2015 - 18/10/2015 (Retro UNIX 386 v1)
42031                                  <1>
42032 0000E5C1 29D2                    <1>       sub    edx, edx
42033 0000E5C3 FEC2                    <1>       inc    dl ; 1 (write access sign)
42034                                  <1> trans_addr_rw:
42035 0000E5C5 50                      <1>       push   eax
42036 0000E5C6 53                      <1>       push   ebx
42037 0000E5C7 52                      <1>       push   edx ; r/w sign (in DL)
42038                                  <1>       ;
42039 0000E5C8 8B1D[84030300]          <1>       mov    ebx, [u.base]
42040 0000E5CE E8BC6CFFFF              <1>       call   get_physical_addr ; get physical address
42041 0000E5D3 730F                    <1>       jnc    short passc_0
42042 0000E5D5 A3[C8030300]            <1>       mov    [u.error], eax
42043 0000E5DA A3[64030300]            <1>       mov    [u.r0], eax ; 12/10/2016
42044                                  <1>       ;pop   edx
42045                                  <1>       ;pop   ebx
42046                                  <1>       ;pop   eax
42047 0000E5DF E9D9DEFFFF              <1>       jmp    error
42048                                  <1> passc_0:
42049 0000E5E4 F6C202                  <1>       test   dl, PTE_A_WRITE ; writable page
42050 0000E5E7 5A                      <1>       pop    edx
42051 0000E5E8 751C                    <1>       jnz    short passc_1
42052                                  <1>
42053 0000E5EA 20D2                    <1>       and    dl, dl
42054 0000E5EC 7418                    <1>       jz     short passc_1
42055                                  <1>       ; read only (duplicated) page -must be copied to a new page-
42056                                  <1>       ; EBX = linear address
42057 0000E5EE 51                      <1>       push   ecx
42058 0000E5EF E83469FFFF              <1>       call   copy_page
42059 0000E5F4 59                      <1>       pop    ecx
42060 0000E5F5 721E                    <1>       jc     short passc_2
42061 0000E5F7 50                      <1>       push   eax ; physical address of the new/allocated page
42062 0000E5F8 E8BC6BFFFF              <1>       call   add_to_swap_queue
42063 0000E5FD 58                      <1>       pop    eax
42064 0000E5FE 81E3FF0F0000            <1>       and    ebx, PAGE_OFF ; 0FFFh
42065                                  <1>       ;mov   ecx, PAGE_SIZE
42066                                  <1>       ;sub   ecx, ebx
42067 0000E604 01D8                    <1>       add    eax, ebx
42068                                  <1> passc_1:
42069 0000E606 A3[C0030300]            <1>       mov    [u.pbase], eax ; physical address
42070 0000E60B 66890D[C4030300]        <1>       mov    [u.pcount], cx ; remain byte count in page (1-4096)
42071 0000E612 5B                      <1>       pop    ebx
42072 0000E613 58                      <1>       pop    eax
42073 0000E614 C3                      <1>       retn
42074                                  <1> passc_2:
42075 0000E615 B804000000              <1>       mov    eax, ERR_MINOR_IM ; "Insufficient memory !" error
42076 0000E61A A3[64030300]            <1>       mov    [u.r0], eax ; 12/10/2016
42077 0000E61F A3[C8030300]            <1>       mov    dword [u.error], eax
42078                                  <1>       ;pop   ebx
42079                                  <1>       ;pop   eax
42080 0000E624 E994DEFFFF              <1>       jmp    error
42081                                  <1>
42082                                  <1> sioreg:
42083                                  <1>       ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
42084                                  <1>       ; 19/05/2015 - 25/07/2015 (Retro UNIX 386 v1)
42085                                  <1>       ; 12/03/2013 - 22/07/2013 (Retro UNIX 8086 v1)
42086                                  <1>       ; INPUTS ->
42087                                  <1>       ;     EBX = system buffer (data) address (r5)
42088                                  <1>       ;     [u.fofp] = pointer to file offset pointer
42089                                  <1>       ;     [u.base] = virtual address of the user buffer
42090                                  <1>       ;     [u.pbase] = physical address of the user buffer
42091                                  <1>       ;     [u.count] = byte count
42092                                  <1>       ;     [u.pcount] = byte count within page frame
42093                                  <1>       ; OUTPUTS ->
42094                                  <1>       ;     ESI = user data offset (r1)
42095                                  <1>       ;     EDI = system (I/O) buffer offset (r2)
42096                                  <1>       ;     ECX = byte count (r3)
42097                                  <1>       ;     EAX = remain bytes after byte count within page frame
42098                                  <1>       ;      (If EAX > 0, transfer will continue from the next page)
42099                                  <1>         ;
42100                                  <1>       ; ((Modified registers:  EDX))
42101                                  <1>
42102 0000E629 8B35[74030300]          <1>         mov     esi, [u.fofp]
42103 0000E62F 8B3E                    <1>         mov     edi, [esi]
42104 0000E631 89F9                    <1>       mov    ecx, edi
42105 0000E633 81C900FEFFFF            <1>       or     ecx, 0FFFFFE00h
42106 0000E639 81E7FF010000            <1>       and    edi, 1FFh
42107 0000E63F 01DF                    <1>       add    edi, ebx ; EBX = system buffer (data) address
42108 0000E641 F7D9                    <1>       neg    ecx
42109 0000E643 3B0D[88030300]          <1>       cmp    ecx, [u.count]
42110 0000E649 7606                    <1>       jna    short sioreg_0
42111 0000E64B 8B0D[88030300]          <1>       mov    ecx, [u.count]
42112                                  <1> sioreg_0:
42113 0000E651 803D[C6030300]00        <1>       cmp    byte [u.kcall], 0
42114 0000E658 7613                    <1>       jna    short sioreg_1
42115                                  <1>        ; the caller is 'mkdir' or 'namei'
42116 0000E65A A1[84030300]            <1>       mov    eax, [u.base]
42117 0000E65F A3[C0030300]            <1>       mov    [u.pbase], eax ; physical address = virtual address
42118 0000E664 66890D[C4030300]        <1>       mov    word [u.pcount], cx ; remain bytes in buffer (1 sector)
42119 0000E66B EB0B                    <1>       jmp    short sioreg_2
```

```
                                <1> sioreg_1:
0000E66D 0FB715[C4030300]       <1>       movzx  edx, word [u.pcount]
0000E674 39D1                   <1>       cmp    ecx, edx
0000E676 772A                   <1>       ja     short sioreg_4 ; transfer count > [u.pcount]
                                <1> sioreg_2: ; 2:
0000E678 31C0                   <1>       xor    eax, eax
                                <1> sioreg_3:
0000E67A 010D[8C030300]         <1>       add    [u.nread], ecx
0000E680 290D[88030300]         <1>       sub    [u.count], ecx
0000E686 010D[84030300]         <1>       add    [u.base], ecx
0000E68C 010E                   <1>         add  [esi], ecx
0000E68E 8B35[C0030300]         <1>       mov    esi, [u.pbase]
0000E694 66290D[C4030300]       <1>       sub    [u.pcount], cx
0000E69B 010D[C0030300]         <1>       add    [u.pbase], ecx
0000E6A1 C3                     <1>         retn
                                <1> sioreg_4:
                                <1>       ; transfer count > [u.pcount]
                                <1>       ; (ecx > edx)
0000E6A2 89C8                   <1>       mov    eax, ecx
0000E6A4 29D0                   <1>       sub    eax, edx ; remain bytes for 1 sector (block) transfer
0000E6A6 89D1                   <1>       mov    ecx, edx ; current transfer count = [u.pcount]
0000E6A8 EBD0                   <1>       jmp    short sioreg_3
                                <1>
                                <1> tswitch: ; Retro UNIX 386 v1
                                <1> tswap:
                                <1>       ; 16/01/2017
                                <1>       ; 21/05/2016 - TRDOS 386 (TRDOS v2.0)
                                <1>       ; 10/05/2015 - 01/09/2015 (Retro UNIX 386 v1)
                                <1>       ; 14/04/2013 - 14/02/2014 (Retro UNIX 8086 v1)
                                <1>       ; time out swap, called when a user times out.
                                <1>       ; the user is put on the low priority queue.
                                <1>       ; This is done by making a link from the last user
                                <1>       ; on the low priority queue to him via a call to 'putlu'.
                                <1>       ; then he is swapped out.
                                <1>
                                <1>       ; TRDOS 386 (TRDOS v2.0) modification ->  ** 21/05/2016 **
                                <1>       ;     * when a high priority (event) process will be stopped
                                <1>       ;     (swapped out, swithched out/off), 'tswap/tswitch' will
                                <1>       ;     not add it to a run queue.
                                <1>       ;     /// What for: Process may be already in a run queue,
                                <1>       ;     it is unspeficied state because process might be started
                                <1>       ;     by a timer event which does not regard previous priority
                                <1>       ;     level and run queue of the process (for fast executing!).
                                <1>       ;     After the 'run for event', process will be sequenced
                                <1>       ;     to run by it's actual run queue. ///
                                <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>       ;    u.uno - users process number
                                <1>       ;    runq+4 - lowest priority queue
                                <1>       ; OUTPUTS ->
                                <1>       ;    r0 - users process number
                                <1>       ;    r2 - lowest priority queue address
                                <1>       ;
                                <1>       ; ((AX = R0, BX = R2)) output
                                <1>       ; ((Modified registers: EDX, EBX, ECX, ESI, EDI))
                                <1>       ;
                                <1>
                                <1>       NOTE:
                                <1>       ;* [u.pri] priority level is specified by run queue which is process
                                <1>       ;  comes to run from.
                                <1>       ;* Initial [u.pri] is 1 ('normal/regular') for programs
                                <1>       ;  (which are launched by MainProg or 'sysexec'), it is changed
                                <1>       ;  to 2 ('high') by timer event, if program uses 'systimer' system call.
                                <1>       ;* Program (Process) also can change it's running priority
                                <1>       ;  from 1 to 0 or up to 2 by using 'syspri' system call; but,
                                <1>       ;  if program selects priority level 2 (high) for running, next time
                                <1>       ;  it is reduced to 1 (normal/regular) because 'syspri' adds this
                                <1>       ;  program to 'run for normal' queue while running duration is a bit
                                <1>       ;  protected from swap/switch out immediate, behalf of other high
                                <1>       ;  priority process in sequence. Program (with high priority) will not
                                <1>       ;  be swapped/switched out (by timer event) before it's time quantum
                                <1>       ;  will be elapsed, but, this will be temporary if program is not using
                                <1>       ;  timer event function.
                                <1>
                                <1>       ;For example:
                                <1>       ;If a process frequently gets a timer event, it runs at high priority
                                <1>       ;level but when it returns from running it returns to actual run queue,
                                <1>       ;not to 'run for event' queue again.
                                <1>       ;'tswap' will not change the sequence at return/stop(swap out) stage.
                                <1>       ;But if priority level not high (=2, 'run for event'), 'tswap/tswitch'
                                <1>       ;will add the stopping process to relevant run queue according to
                                <1>       ;[u.pri] priority level.
                                <1>
                                <1>       ; 16/01/2017
0000E6AA BB[54030300]           <1>       mov    ebx, runq+2  ; 'runq_normal' ; normal/regular priority
                                <1>       ; 21/05/2016
                                <1>       ;cmp   byte [u.pri], 2     ; high priority (run for event) ?
                                <1>       ;jnb   short swap
                                <1>       ; 16/01/2017
                                <1>       ; (Normal and also high/event priority processes will be added to
                                <1>       ; normal priority run queue for ensuring circular running sequence!)
```

```
42223                                  <1>       ; (Timer interrupt or 'syspri' system call may change priority and run
42224                                  <1>       ; queue to high/event level.)
42225 0000E6AF 803D[A9030300]00        <1>       cmp    byte [u.pri], 0
42226 0000E6B6 7702                    <1>       ja     short tswap_1; normal priority run queue
42227                                  <1>       ;
42228 0000E6B8 43                      <1>       inc    ebx
42229 0000E6B9 43                      <1>       inc    ebx          ; runq+4, 'runq_background', low priority
42230                                  <1> tswap_1:
42231 0000E6BA A0[B3030300]            <1>       mov    al, [u.uno]
42232                                  <1>                     ; movb u.uno,r1 / move users process number to r1
42233                                  <1>               ; mov  $runq+4,r2
42234                                  <1>                     ; / move lowest priority queue address to r2
42235                                  <1>               ; ebx = run queue
42236 0000E6BF E8FE000000              <1>       call   putlu
42237                                  <1>               ; jsr r0,putlu / create link from last user on Q to
42238                                  <1>                           ; / u.uno's user
42239                                  <1>
42240                                  <1> switch: ; Retro UNIX 386 v1
42241                                  <1> swap:
42242                                  <1>       ; 14/11/2017
42243                                  <1>       ; 02/01/2017
42244                                  <1>       ; 02/05/2016, 20/05/2016, 21/05/2016
42245                                  <1>       ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
42246                                  <1>       ; 10/05/2015 - 02/09/2015 (Retro UNIX 386 v1)
42247                                  <1>       ; 14/04/2013 - 08/03/2014 (Retro UNIX 8086 v1)
42248                                  <1>       ;
42249                                  <1>       ; 'swap' is routine that controls the swapping of processes
42250                                  <1>       ; in and out of core.
42251                                  <1>       ;
42252                                  <1>       ; TRDOS 386 (TRDOS v2.0) modification ->  ** 20/05/2016 **
42253                                  <1>       ;     * 3 different priority level is applied
42254                                  <1>       ;     (just as original unix v1)
42255                                  <1>       ;     1) high priority (event) run queue, 'runq_event'
42256                                  <1>       ;     2) normal priority (regular) run queue, 'runq_normal'
42257                                  <1>       ;     3) low priority (background) run queue, 'runq_backgroud'
42258                                  <1>       ;     'swap' code will run a process which has max. priority
42259                                  <1>         ;       (for earliest event at first)
42260                                  <1>       ;
42261                                  <1>       ; Retro UNIX 386 v1 modification ->
42262                                  <1>       ;      swap (software task switch) is performed by changing
42263                                  <1>       ;     user's page directory (u.pgdir) instead of segment change
42264                                  <1>       ;     as in Retro UNIX 8086 v1.
42265                                  <1>       ;
42266                                  <1>       ; RETRO UNIX 8086 v1 modification ->
42267                                  <1>       ;      'swap to disk' is replaced with 'change running segment'
42268                                  <1>       ;     according to 8086 cpu (x86 real mode) architecture.
42269                                  <1>       ;     pdp-11 was using 64KB uniform memory while IBM PC
42270                                  <1>       ;     compatibles was using 1MB segmented memory
42271                                  <1>       ;     in 8086/8088 times.
42272                                  <1>       ;
42273                                  <1>       ; INPUTS ->
42274                                  <1>       ;    runq table - contains processes to run.
42275                                  <1>       ;    p.link - contains next process in line to be run.
42276                                  <1>       ;    u.uno - process number of process in core
42277                                  <1>       ;    s.stack - swap stack used as an internal stack for swapping.
42278                                  <1>       ; OUTPUTS ->
42279                                  <1>       ;    (original unix v1 -> present process to its disk block)
42280                                  <1>       ;    (original unix v1 -> new process into core ->
42281                                  <1>       ;        Retro Unix 8086 v1 -> segment registers changed
42282                                  <1>       ;        for new process)
42283                                  <1>       ;    u.quant = 3 (Time quantum for a process)
42284                                  <1>       ;     ((INT 1Ch count down speed -> 18.2 times per second)
42285                                  <1>       ;    RETRO UNIX 8086 v1 will use INT 1Ch (18.2 times per second)
42286                                  <1>       ;      for now, it will swap the process if there is not
42287                                  <1>       ;      a keyboard event (keystroke) (Int 15h, function 4Fh)
42288                                  <1>       ;      or will count down from 3 to 0 even if there is a
42289                                  <1>       ;        keyboard event locking due to repetitive key strokes.
42290                                  <1>       ;      u.quant will be reset to 3 for RETRO UNIX 8086 v1.
42291                                  <1>       ;
42292                                  <1>       ; ((Modified registers: EAX, EDX, EBX, ECX, ESI, EDI))
42293                                  <1>
42294                                  <1>       ;NOTE:
42295                                  <1>       ;High priority queue is the first for selecting a process to run.
42296                                  <1>       ;If there is not a process in high priority level run queue,
42297                                  <1>       ;a process in normal priority run queue will be selected
42298                                  <1>       ;or a proces in low priority run queue will be selected if normal
42299                                  <1>       ;priority level run queue is empty.
42300                                  <1>
42301                                  <1>       ; 21/05/2016 - (3 priority levels, 3 run queues)
42302 0000E6C4 BE[52030300]            <1>       mov    esi, runq ; 'runq_event' ; high priority, 'run for event'
42303 0000E6C9 C605[B45F0100]03        <1>       mov    byte [priority], 3 ; high priority + 1
42304 0000E6D0 31DB                    <1>       xor    ebx, ebx ; 02/01/2017
42305                                  <1> swap_0: ; 1: / search runq table for highest priority process
42306 0000E6D2 66AD                    <1>       lodsw  ; mov ax, [esi], add esi+2
42307                                  <1>       ;xor   ebx, ebx ; 02/05/2016
42308 0000E6D4 6621C0                  <1>       and    ax, ax ; are there any processes to run in this Q entry
42309 0000E6D7 750E                    <1>       jnz    short swap_2
42310                                  <1>       ; 21/05/2026
42311                                  <1>       ; runq_normal = runq+2, runq_background = runq+4
42312 0000E6D9 FE0D[B45F0100]          <1>       dec    byte [priority] ; 3 -> 3, 2 -> 1, 1-> 0
42313 0000E6DF 75F1                    <1>       jnz    short swap_0
42314                                  <1>       ;cmp   esi, runq+6  ; if zero compare address to end of table
42315                                  <1>       ;jb    short swap_0 ; if not at end, go back
42316                                  <1> swap_1:
42317                                  <1>       ; 02/05/2016
42318                                  <1>       ; 29/04/2016 (TRDOS 386 = TRDOS v2.0)
42319                                  <1>       ; No user process to run...
42320                                  <1>       ; Run the kernel process... MainProg: Internal Command Interpreter
42321 0000E6E1 FEC0                    <1>       inc    al ; mov al, 1  ; process number of MainProg
42322 0000E6E3 FEC3                    <1>       inc    bl ; mov bl, al ; 1
42323 0000E6E5 EB1E                    <1>       jmp    short swap_4
42324                                  <1> swap_2:
42325                                  <1>       ; 21/05/2016
```

```
42326 0000E6E7 FE0D[B45F0100]      <1>       dec    byte [priority] ; priority level of present user/process
42327                              <1>                               ; 0, 1, 2
42328 0000E6ED 4E                  <1>       dec    esi
42329 0000E6EE 4E                  <1>         dec     esi
42330                              <1>       ;
42331 0000E6EF 88C3                <1>       mov    bl, al
42332 0000E6F1 38E0                <1>       cmp    al, ah ; is there only 1 process in the queue to be run
42333 0000E6F3 740A                <1>       je     short swap_3 ; yes
42334 0000E6F5 8AA3[9F000300]      <1>       mov    ah, [ebx+p.link-1]
42335 0000E6FB 8826                <1>              mov    [esi], ah ; move next process in line into run queue
42336 0000E6FD EB06                <1>       jmp    short swap_4
42337                              <1> swap_3:
42338 0000E6FF 6631D2              <1>       xor    dx, dx
42339 0000E702 668916              <1>       mov    [esi], dx ; zero the entry; no processes on the Q
42340                              <1> swap_4:
42341 0000E705 8A25[B3030300]      <1>       mov    ah, [u.uno]
42342 0000E70B 38C4                <1>       cmp    ah, al ; is this process the same as the process in core?
42343 0000E70D 743B                <1>              je     short swap_8 ; yes, don't have to swap
42344 0000E70F 08E4                <1>       or     ah, ah ; is the process # = 0
42345 0000E711 740D                <1>              jz     short swap_6 ; 'sysexit'
42346                              <1>       ;cmp   ah, al ;is this process the same as the process in core?
42347                              <1>              ;je    short swap_8 ; yes, don't have to swap
42348 0000E713 8925[60030300]      <1>       mov    [u.usp], esp ; return  address for 'syswait' & 'sleep'
42349 0000E719 E834000000          <1>       call   wswap  ; write out core to disk
42350 0000E71E EB1C                <1>       jmp    short swap_7
42351                              <1> swap_6:
42352                              <1>       ; Deallocate memory pages belong to the process
42353                              <1>       ; which is being terminated.
42354                              <1>       ; (Retro UNIX 386 v1 modification !)
42355                              <1>       ;
42356 0000E720 53                  <1>       push   ebx
42357 0000E721 A1[B8030300]        <1>       mov    eax, [u.pgdir]  ; page directory of the process
42358 0000E726 8B1D[BC030300]      <1>       mov    ebx, [u.ppgdir] ; page directory of the parent process
42359 0000E72C E88265FFFF          <1>       call   deallocate_page_dir
42360 0000E731 A1[B4030300]        <1>       mov    eax, [u.upage] ; 'user' structure page of the process
42361 0000E736 E81D66FFFF          <1>       call   deallocate_page
42362 0000E73B 5B                  <1>       pop    ebx
42363                              <1> swap_7:
42364 0000E73C C0E302              <1>       shl    bl, 2 ; * 4
42365 0000E73F 8B83[BC000300]      <1>       mov    eax, [ebx+p.upage-4] ; the 'u' page of the new process
42366 0000E745 E840000000          <1>       call   rswap ; read new process into core
42367                              <1> swap_8:
42368                              <1>       ; Retro UNIX  8086 v1 modification !
42369 0000E74A C605[A8030300]04    <1>       mov    byte [u.quant], time_count
42370 0000E751 C3                  <1>       retn
42371                              <1>
42372                              <1> wswap:  ; < swap out, swap to disk >
42373                              <1>       ; 28/02/2017 (fnsave)
42374                              <1>       ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
42375                              <1>       ; 09/05/2015 (Retro UNIX 386 v1)
42376                              <1>       ; 26/05/2013 - 08/03/2014 (Retro UNIX 8086 v1)
42377                              <1>       ; 'wswap' writes out the process that is in core onto its
42378                              <1>       ; appropriate disk area.
42379                              <1>       ;
42380                              <1>       ; Retro UNIX 386 v1 modification ->
42381                              <1>       ;       User (u) structure content and the user's register content
42382                              <1>       ;     will be copied to the process's/user's UPAGE (a page for
42383                              <1>       ;     saving 'u' structure and user registers for task switching).
42384                              <1>       ;     u.usp - points to kernel stack address which contains
42385                              <1>       ;             user's registers while entering system call.
42386                              <1>       ;     u.sp  - points to kernel stack address
42387                              <1>       ;             to return from system call -for IRET-.
42388                              <1>       ;     [u.usp]+32+16 = [u.sp]
42389                              <1>       ;     [u.usp] -> edi, esi, ebp, esp (= [u.usp]+32), ebx,
42390                              <1>       ;             edx, ecx, eax, gs, fs, es, ds, -> [u.sp].
42391                              <1>       ;
42392                              <1>       ; Retro UNIX 8086 v1 modification ->
42393                              <1>       ;       'swap to disk' is replaced with 'change running segment'
42394                              <1>       ;     according to 8086 cpu (x86 real mode) architecture.
42395                              <1>       ;     pdp-11 was using 64KB uniform memory while IBM PC
42396                              <1>       ;     compatibles was using 1MB segmented memory
42397                              <1>       ;      in 8086/8088 times.
42398                              <1>       ;
42399                              <1>       ; INPUTS ->
42400                              <1>       ;    u.break - points to end of program
42401                              <1>       ;    u.usp - stack pointer at the moment of swap
42402                              <1>       ;    core - beginning of process program
42403                              <1>       ;    ecore - end of core
42404                              <1>       ;    user - start of user parameter area
42405                              <1>       ;    u.uno - user process number
42406                              <1>       ;    p.dska - holds block number of process
42407                              <1>       ; OUTPUTS ->
42408                              <1>       ;    swp I/O queue
42409                              <1>       ;    p.break - negative word count of process
42410                              <1>       ;    r1 - process disk address
42411                              <1>       ;    r2 - negative word count
42412                              <1>       ;
42413                              <1>       ; RETRO UNIX 8086 v1 input/output:
42414                              <1>       ;
42415                              <1>       ; INPUTS ->
42416                              <1>       ;    u.uno - process number (to be swapped out)
42417                              <1>       ; OUTPUTS ->
42418                              <1>       ;    none
42419                              <1>       ;
42420                              <1>       ;   ((Modified registers: ECX, ESI, EDI))
42421                              <1>       ;
42422                              <1>
42423                              <1>       ; 28/02/2017
42424                              <1>       ;cmp   byte [multi_tasking], 0  ; Musti tasking mode ?
42425                              <1>       ;jna   short wswp
42426 0000E752 803D[DA030300]00    <1>       cmp    byte [u.fpsave], 0 ; 28/02/2017
42427 0000E759 7606                <1>       jna    short wswp
42428 0000E75B DD35[DC030300]      <1>       fnsave [u.fpregs] ; save floating point registers (94 bytes)
```

```
                                <1> wswp:
0000E761 8B3D[B4030300]         <1>       mov    edi, [u.upage] ; process's user (u) structure page addr
0000E767 B938000000             <1>       mov    ecx, (U_SIZE + 3) / 4
0000E76C BE[5C030300]           <1>       mov    esi, user ; active user (u) structure
0000E771 F3A5                   <1>       rep    movsd
                                <1>       ;
0000E773 8B35[60030300]         <1>       mov    esi, [u.usp] ; esp (system stack pointer,
                                <1>                           ;      points to user registers)
0000E779 8B0D[5C030300]         <1>       mov    ecx, [u.sp]  ; return address from the system call
                                <1>                           ; (for IRET)
                                <1>                           ; [u.sp] -> EIP (user)
                                <1>                           ; [u.sp+4]-> CS (user)
                                <1>                           ; [u.sp+8] -> EFLAGS (user)
                                <1>                           ; [u.sp+12] -> ESP (user)
                                <1>                           ; [u.sp+16] -> SS (user)
0000E77F 29F1                   <1>       sub    ecx, esi     ; required space for user registers
0000E781 83C114                 <1>       add    ecx, 20           ; +5 dwords to return from system call
                                <1>                           ; (for IRET)
0000E784 C1E902                 <1>       shr    ecx, 2
0000E787 F3A5                   <1>       rep    movsd
0000E789 C3                     <1>       retn
                                <1>
                                <1> rswap:  ; < swap in, swap from disk >
                                <1>       ; 28/02/2017 (frstor)
                                <1>       ; 15/01/2017
                                <1>       ; 14/01/2017
                                <1>       ; 21/05/2016
                                <1>       ; 03/05/2016
                                <1>       ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
                                <1>       ; 09/05/2015 - 15/09/2015 (Retro UNIX 386 v1)
                                <1>       ; 26/05/2013 - 08/03/2014 (Retro UNIX 8086 v1)
                                <1>       ; 'rswap' reads a process whose number is in r1,
                                <1>       ; from disk into core.
                                <1>       ;
                                <1>       ; Retro UNIX 386 v1 modification ->
                                <1>       ;      User (u) structure content and the user's register content
                                <1>       ;     will be restored from process's/user's UPAGE (a page for
                                <1>       ;     saving 'u' structure and user registers for task switching).
                                <1>       ;     u.usp - points to kernel stack address which contains
                                <1>       ;             user's registers while entering system call.
                                <1>       ;     u.sp  - points to kernel stack address
                                <1>       ;             to return from system call -for IRET-.
                                <1>       ;     [u.usp]+32+16 = [u.sp]
                                <1>       ;     [u.usp] -> edi, esi, ebp, esp (= [u.usp]+32), ebx,
                                <1>       ;             edx, ecx, eax, gs, fs, es, ds, -> [u.sp].
                                <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>       ;    r1 - process number of process to be read in
                                <1>       ;    p.break - negative of word count of process
                                <1>       ;    p.dska - disk address of the process
                                <1>       ;    u.emt - determines handling of emt's
                                <1>       ;    u.ilgins - determines handling of illegal instructions
                                <1>       ; OUTPUTS ->
                                <1>       ;    8 = (u.ilgins)
                                <1>       ;    24 = (u.emt)
                                <1>       ;    swp - bit 10 is set to indicate read
                                <1>       ;            (bit 15=0 when reading is done)
                                <1>       ;    swp+2 - disk block address
                                <1>       ;    swp+4 - negative word count
                                <1>       ;      ((swp+6 - address of user structure))
                                <1>       ;
                                <1>       ; RETRO UNIX 8086 v1 input/output:
                                <1>       ;
                                <1>       ; INPUTS ->
                                <1>       ;    AL      - new process number (to be swapped in)
                                <1>       ; OUTPUTS ->
                                <1>       ;    none
                                <1>       ;
                                <1>       ;   ((Modified registers: EAX, ECX, ESI, EDI, ESP))
                                <1>       ;
                                <1>       ; Retro UNIX 386 v1 - modification ! 14/05/2015
0000E78A 89C6                   <1>       mov    esi, eax  ; process's user (u) structure page addr
0000E78C B938000000             <1>       mov    ecx, (U_SIZE + 3) / 4
0000E791 BF[5C030300]           <1>       mov    edi, user ; active user (u) structure
0000E796 F3A5                   <1>       rep    movsd
0000E798 58                     <1>       pop    eax    ; 'rswap' return address
                                <1>       ;
                                <1>       ;cli
0000E799 8B3D[60030300]         <1>       mov    edi, [u.usp] ; esp (system stack pointer,
                                <1>                           ;     points to user registers)
0000E79F 89FC                   <1>       mov    esp, edi     ; 14/01/2017
0000E7A1 8B0D[5C030300]         <1>       mov    ecx, [u.sp]  ; return address from the system call
                                <1>                           ; (for IRET)
                                <1>                           ; [u.sp] -> EIP (user)
                                <1>                           ; [u.sp+4]-> CS (user)
                                <1>                           ; [u.sp+8] -> EFLAGS (user)
                                <1>                           ; [u.sp+12] -> ESP (user)
                                <1>                           ; [u.sp+16] -> SS (user)
0000E7A7 29F9                   <1>       sub    ecx, edi     ; required space for user registers
0000E7A9 83C114                 <1>       add    ecx, 20           ; +5 dwords to return from system call
                                <1>                           ; (for IRET)
0000E7AC C1E902                 <1>       shr    ecx, 2
0000E7AF F3A5                   <1>       rep    movsd
                                <1>       ;mov   esp, [u.usp] ; 15/09/2015
                                <1>       ;sti
                                <1>       ; 28/02/2017
```

```
42532                                  <1>       ;cmp   byte [multi_tasking], 0  ; Musti tasking mode ?
42533                                  <1>       ;jna   short rswp_retn
42534 0000E7B1 803D[DA030300]00        <1>       cmp    byte [u.fpsave], 0
42535 0000E7B8 7606                    <1>       jna    short rswp_retn
42536 0000E7BA DD25[DC030300]          <1>       frstor [u.fpregs] ; restore floating point regs (94 bytes)
42537                                  <1> rswp_retn:
42538 0000E7C0 50                      <1>       push   eax    ; 'rswap' return address
42539 0000E7C1 C3                      <1>       retn
42540                                  <1>
42541                                  <1> putlu:
42542                                  <1>       ; 20/05/2016
42543                                  <1>       ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
42544                                  <1>       ; 10/05/2015 - 12/09/2015 (Retro UNIX 386 v1)
42545                                  <1>       ; 15/04/2013 - 23/02/2014 (Retro UNIX 8086 v1)
42546                                  <1>       ; 'putlu' is called with a process number in r1 and a pointer
42547                                  <1>       ; to lowest priority Q (runq+4) in r2. A link is created from
42548                                  <1>       ; the last process on the queue to process in r1 by putting
42549                                  <1>       ; the process number in r1 into the last process's link.
42550                                  <1>       ;
42551                                  <1>       ; INPUTS ->
42552                                  <1>       ;    r1 - user process number
42553                                  <1>       ;    r2 - points to lowest priority queue
42554                                  <1>       ;    p.dska - disk address of the process
42555                                  <1>       ;    u.emt - determines handling of emt's
42556                                  <1>       ;    u.ilgins - determines handling of illegal instructions
42557                                  <1>       ; OUTPUTS ->
42558                                  <1>       ;    r3 - process number of last process on the queue upon
42559                                  <1>       ;       entering putlu
42560                                  <1>       ;    p.link-1 + r3 - process number in r1
42561                                  <1>       ;    r2 - points to lowest priority queue
42562                                  <1>       ;
42563                                  <1>       ; ((Modified registers: EDX, EBX))
42564                                  <1>       ;
42565                                  <1>       ; / r1 = user process no.; r2 points to lowest priority queue
42566                                  <1>
42567                                  <1>       ; EBX = r2
42568                                  <1>       ; EAX = r1 (AL=r1b)
42569                                  <1>
42570                                  <1>       ; 20/05/2016
42571                                  <1>       ; AL = process number (1 to 16) // Retro UNIX 8086, 386 v1 //
42572                                  <1>       ;     (max. 16 processes available for current kernel version)
42573                                  <1>       ; EBX = run queue address ; 20/05/2016 (TRDOS 386)
42574                                  <1>             ; which is one of following addresses:
42575                                  <1>             ;  1) 'runq_event' high priority run queue
42576                                  <1>             ;  2) 'runq_normal' normal/regular priority run queue
42577                                  <1>             ;  3) 'runq_background' low priority run queue
42578                                  <1>
42579                                  <1>       ;mov   ebx, runq
42580 0000E7C2 0FB613                 <1>       movzx        edx, byte [ebx]
42581 0000E7C5 43                      <1>       inc    ebx
42582 0000E7C6 20D2                    <1>       and    dl, dl
42583                                  <1>             ; tstb (r2)+ / is queue empty?
42584 0000E7C8 740A                    <1>             jz     short putlu_1
42585                                  <1>             ; beq 1f / yes, branch
42586 0000E7CA 8A13                    <1>       mov    dl, [ebx] ; 12/09/2015
42587                                  <1>             ; movb (r2),r3 / no, save the "last user" process number
42588                                  <1>                       ; / in r3
42589 0000E7CC 8882[9F000300]          <1>             mov    [edx+p.link-1], al
42590                                  <1>             ; movb r1,p.link-1(r3) / put pointer to user on
42591                                  <1>                       ; / "last users" link
42592 0000E7D2 EB03                    <1>       jmp    short putlu_2
42593                                  <1>             ; br 2f /
42594                                  <1> putlu_1: ; 1:
42595 0000E7D4 8843FF                  <1>       mov    [ebx-1], al
42596                                  <1>                     ; movb r1,-1(r2) / user is only user;
42597                                  <1>                         ; / put process no. at beginning and at end
42598                                  <1> putlu_2: ; 2:
42599 0000E7D7 8803                    <1>       mov    [ebx], al
42600                                  <1>                     ; movb r1,(r2) / user process in r1 is now the last entry
42601                                  <1>                         ; / on the queue
42602 0000E7D9 88C2                    <1>       mov    dl, al
42603 0000E7DB 88B2[9F000300]          <1>         mov     [edx+p.link-1], dh ; 0
42604                                  <1>             ; dec r2 / restore r2
42605 0000E7E1 C3                      <1>         retn
42606                                  <1>             ; rts r0
42607                                  <1> sysver:
42608                                  <1>       ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
42609 0000E7E2 C705[64030300]0002-     <1>       mov    dword [u.r0], 200h ; AH = major version, AL = minor version
42610 0000E7EA 0000                    <1>
42611 0000E7EC E9ECDCFFFF              <1>       jmp    sysret
42612                                  <1>
42613                                  <1> sysreserved1:
42614                                  <1>       ; 19/05/2016 - TRDOS 386 (TRDOS v2.0)
42615                                  <1>       ; // name and content will be changed later //
42616 0000E7F1 C705[64030300]E007-     <1>       mov    dword [u.r0], 2016
42617 0000E7F9 0000                    <1>
42618 0000E7FB E9DDDCFFFF              <1>       jmp    sysret
42619                                  <1>
42620                                  <1> syspri: ; change running priority (of the process)
42621                                  <1>       ; 21/05/2016
42622                                  <1>       ; 20/05/2026 - TRDOS 386 (TRDOS v2.0)
42623                                  <1>       ; INPUT ->
42624                                  <1>       ;     BL = priority level
42625                                  <1>       ;        0 = low running priority (running on background)
42626                                  <1>       ;        1 = normal/regular priority (running as regular)
42627                                  <1>       ;        2 = high/event priority (running for event)
42628                                  <1>       ;        >2 = invalid, it will accepted as 2 (event)
42629                                  <1>       ;        0FFh = get/return current running priority only
42630                                  <1>       ; OUTPUT ->
42631                                  <1>       ;     * if current [u.pri] < 2
42632                                  <1>       ;       if BL input < 0FFh ->
42633                                  <1>       ;          [u.pri] is updated as in BL input (0,1,2)
42634                                  <1>       ;       if BL input = 0FFh -> AL = [u.pri] (current)
```

```
42635                                  <1>       ;
42636                                  <1>       ;     * if current [u.pri] = 2
42637                                  <1>       ;       if BL input < 0FFh -> cf = 1 & AL = 2
42638                                  <1>       ;       if BL input = 0FFh -> cf = 0 & AL = 2
42639                                  <1>       ;
42640                                  <1>       ;     NOTE:
42641                                  <1>       ;     If [u.pri] = 2, it can not be changed to 1 or 0;
42642                                  <1>       ;     because, run queue of the running process is unspecified
42643                                  <1>       ;     at this      stage. Process might be started by a timer event
42644                                  <1>       ;     or priority might be changed to high by previous
42645                                  <1>       ;     'syspri' system     call. In both cases, the process is in
42646                                  <1>       ;     'runq_normal' or 'runq_background' queue.
42647                                  <1>       ;     As result of this fact, when the [u.quant] time quantum
42648                                  <1>       ;     of the process is elapsed or 'sysrele' system call is
42649                                  <1>       ;     instructed by the process, 'tswap' ('tswitch') procedure
42650                                  <1>       ;     will be called (to 'swap' or 'switch' out the procedure)
42651                                  <1>       ;     and it will not call 'putlu' to add the (stopping)
42652                                  <1>       ;     process to relevant run queue when [u.pri] = 2.
42653                                  <1>       ;     (Otherwise, it would be possible to add process to
42654                                  <1>       ;     a run queue while it is already in a run queue, wrongly.)
42655                                  <1>       ;
42656                                  <1>       ;     If [u.pri]< 2, 'tswap/tswitch' procedure will call
42657                                  <1>       ;     'putlu' to add process to relevant run queue
42658                                  <1>       ;     according to [u.pri] value. ('runq_normal' for 1,
42659                                  <1>       ;     'runq_background' for 0).
42660                                  <1>       ;
42661                                  <1>       ;     If BL input >= 2 and < 0FFh while [u.pri] < 2,
42662                                  <1>       ;     process will be added to 'runq_normal' queue and
42663                                  <1>       ;     [u.pri] will be set to 2. (in 'syspri' system call)
42664                                  <1>       ;
42665                                  <1>
42666 0000E800 29C0                   <1>       sub    eax, eax ; 0
42667 0000E802 A3[C8030300]           <1>       mov    [u.error], eax
42668                                  <1>
42669 0000E807 A0[A9030300]           <1>       mov    al, [u.pri]
42670 0000E80C A3[64030300]           <1>       mov    [u.r0], eax
42671                                  <1>
42672 0000E811 FEC3                   <1>       inc    bl
42673 0000E813 0F84C4DCFFFF           <1>       jz     sysret ; 0FFh -> 0, get priority level
42674                                  <1>
42675 0000E819 3C02                   <1>       cmp    al, 2
42676 0000E81B 0F839CDCFFFF           <1>       jnb    error ; CF = 1 & AL = 2 (& last error = 0)
42677                                  <1>
42678 0000E821 FECB                   <1>       dec    bl
42679 0000E823 80FB02                 <1>       cmp    bl, 2
42680 0000E826 7602                   <1>       jna    short syspri_1
42681 0000E828 B302                   <1>       mov    bl, 2
42682                                  <1> syspri_1:
42683 0000E82A 881D[A9030300]         <1>       mov    [u.pri], bl
42684 0000E830 80FB02                 <1>       cmp    bl, 2
42685 0000E833 0F82A4DCFFFF           <1>         jb     sysret
42686                                  <1>
42687                                  <1>       ; here...
42688                                  <1>       ; Priority of current process has been changed to high
42689                                  <1>       ; ('run for event') but current process will be added to
42690                                  <1>       ; 'run as normal' queue. ('run for event' high priority
42691                                  <1>       ; queue is under control of timer -& RTC- interrupt only!)
42692                                  <1>       ;
42693                                  <1>       ; (Otherwise, process can fall into black hole!
42694                                  <1>       ; e.g. if it is not in waiting list and it has not got
42695                                  <1>       ; a timer event and it is not in a run queue!
42696                                  <1>       ; Because, when [u.pri] is 2, 'tswap/tswitch' will not
42697                                  <1>       ; add the stopping process to a run queue.)
42698                                  <1>
42699 0000E839 A0[B3030300]           <1>       mov    al, [u.uno]
42700 0000E83E BB[54030300]           <1>       mov    ebx, runq_normal ; normal priority !
42701                                  <1>                               ; [u.pri] is set to high
42702                                  <1>                               ; but 'runq_event' queue is set
42703                                  <1>                               ; only by the kernel's timer
42704                                  <1>                               ; event function (timer interrupt).
42705 0000E843 E87AFFFFFF             <1>       call   putlu
42706 0000E848 E990DCFFFF             <1>       jmp    sysret
42707                                  <1>
42708                                  <1> cpass: ; / get next character from user area of core and put it in AL (r1)
42709                                  <1>       ; 02/05/2016 - TRDOS 386 (TRDOS v2.0)
42710                                  <1>       ; 19/05/2015 - 18/10/2015 (Retro UNIX 386 v1)
42711                                  <1>       ; 14/08/2013 - 20/09/2013 (Retro UNIX 8086 v1)
42712                                  <1>       ; INPUTS ->
42713                                  <1>       ;     [u.base] = virtual address in user area
42714                                  <1>       ;     [u.count] = byte count (max.)
42715                                  <1>       ;     [u.pcount] = byte count in page (0 = reset)
42716                                  <1>       ; OUTPUTS ->
42717                                  <1>       ;     AL = the character which is pointed by [u.base]
42718                                  <1>       ;     zf = 1 -> transfer count has been completed
42719                                  <1>         ;
42720                                  <1>       ; ((Modified registers:  EAX, EDX, ECX))
42721                                  <1>       ;
42722 0000E84D 833D[88030300]00       <1>       cmp    dword [u.count], 0  ; have all the characters been transferred
42723                                  <1>                                  ; i.e., u.count, # of chars. left
42724 0000E854 763F                   <1>       jna    short cpass_3    ; to be transferred = 0?) yes, branch
42725 0000E856 FF0D[88030300]         <1>       dec    dword [u.count]        ; no, decrement u.count
42726                                  <1>         ; 19/05/2015
42727                                  <1>       ;(Retro UNIX 386 v1 - translation from user's virtual address
42728                                  <1>       ;                  to physical address
42729 0000E85C 66833D[C4030300]00     <1>       cmp    word [u.pcount], 0 ; byte count in page = 0 (initial value)
42730                                  <1>                           ; 1-4095 --> use previous physical base address
42731                                  <1>                           ; in [u.pbase]
42732 0000E864 770E                   <1>       ja     short cpass_1
42733 0000E866 833D[BC030300]00       <1>       cmp     dword [u.ppgdir], 0  ; is the caller os kernel
42734 0000E86D 7427                   <1>         je      short cpass_k       ; (sysexec, '/etc/init') ?  (MainProg)
42735 0000E86F E849FDFFFF             <1>       call   trans_addr_r
42736                                  <1> cpass_1:
42737 0000E874 66FF0D[C4030300]       <1>       dec    word [u.pcount]
```

```
                                <1> cpass_2:
0000E87B 8B15[C0030300]         <1>       mov   edx, [u.pbase]
0000E881 8A02                   <1>       mov   al, [edx]    ; take the character pointed to
                                <1>                          ; by u.base and put it in r1
0000E883 FF05[8C030300]         <1>       inc   dword [u.nread] ; increment no. of bytes transferred
0000E889 FF05[84030300]         <1>       inc   dword [u.base]  ; increment the buffer address to point to the
                                <1>                             ; next byte
0000E88F FF05[C0030300]         <1>       inc   dword [u.pbase]
                                <1> cpass_3:
0000E895 C3                     <1>       retn
                                <1> cpass_k:
                                <1>       ; 02/07/2015
                                <1>       ; The caller is os kernel
                                <1>       ; (get sysexec arguments from kernel's memory space)
0000E896 8B1D[84030300]         <1>       mov   ebx, [u.base]
0000E89C 66C705[C4030300]00-    <1>         mov     word [u.pcount], PAGE_SIZE ; 4096
0000E8A4 10                     <1>
0000E8A5 891D[C0030300]         <1>       mov   [u.pbase], ebx
0000E8AB EBCE                   <1>       jmp   short cpass_2
                                <1>
                                <1> transfer_to_user_buffer: ; fast transfer
                                <1>       ; 27/05/2016
                                <1>       ; 16/05/2016 - TRDOS 386 (TRDOS v2.0)
                                <1>       ;
                                <1>       ; INPUT ->
                                <1>       ;     ESI = source address in system space
                                <1>       ;     EDI = user's buffer address
                                <1>       ;     ECX = transfer (byte) count
                                <1>       ;     [u.pgdir] = user's page directory
                                <1>       ; OUTPUT ->
                                <1>       ;     ECX = actual transfer count
                                <1>       ;     cf = 1 -> error
                                <1>       ;     [u.count] = remain byte count
                                <1>       ;
                                <1>       ; Modified registers: eax, ecx
                                <1>       ;
                                <1>
0000E8AD 21C9                   <1>       and   ecx, ecx
0000E8AF 743B                   <1>       jz    short ttub_4
                                <1>
0000E8B1 890D[88030300]         <1>       mov   [u.count], ecx
                                <1>
0000E8B7 57                     <1>       push  edi
0000E8B8 56                     <1>       push  esi
0000E8B9 53                     <1>       push  ebx
0000E8BA 52                     <1>       push  edx
0000E8BB 51                     <1>       push  ecx
                                <1>
0000E8BC 89FB                   <1>       mov   ebx, edi
0000E8BE 81C300004000           <1>       add   ebx, CORE ; 27/05/2016
                                <1> ttub_1:
                                <1>       ; ebx = virtual (linear) address
                                <1>       ; [u.pgdir] = user's page directory
0000E8C4 E8CC69FFFF             <1>             call  get_physical_addr_x ; get physical address
0000E8C9 7222                   <1>       jc    short ttub_5
                                <1>       ; eax = physical address
                                <1>       ; ecx = remain byte count in page (1-4096)
0000E8CB 89C7                   <1>       mov   edi, eax
0000E8CD A1[88030300]           <1>       mov   eax, [u.count]
0000E8D2 39C1                   <1>       cmp   ecx, eax
0000E8D4 7602                   <1>       jna   short ttub_2
0000E8D6 89C1                   <1>       mov   ecx, eax
                                <1> ttub_2:
0000E8D8 29C8                   <1>       sub   eax, ecx
0000E8DA 01CB                   <1>       add   ebx, ecx
0000E8DC F3A4                   <1>       rep   movsb
0000E8DE A3[88030300]           <1>       mov   [u.count], eax
0000E8E3 09C0                   <1>       or    eax, eax
0000E8E5 75DD                   <1>       jnz   short ttub_1
                                <1> ttub_retn:
                                <1> tfub_retn:
0000E8E7 59                     <1>       pop   ecx ; transfer count = actual transfer count
                                <1> ttub_3:
0000E8E8 5A                     <1>       pop   edx
0000E8E9 5B                     <1>       pop   ebx
0000E8EA 5E                     <1>       pop   esi
0000E8EB 5F                     <1>       pop   edi
                                <1> ttub_4:
0000E8EC C3                     <1>       retn
                                <1> ttub_5:
0000E8ED 59                     <1>       pop   ecx
0000E8EE 2B0D[88030300]         <1>       sub   ecx, [u.count] ; actual transfer count
0000E8F4 F9                     <1>       stc
0000E8F5 EBF1                   <1>       jmp   short ttub_3
                                <1>
                                <1> transfer_from_user_buffer: ; fast transfer
                                <1>       ; 27/05/2016
                                <1>       ; 16/05/2016 - TRDOS 386 (TRDOS v2.0)
                                <1>       ;
                                <1>       ; INPUT ->
                                <1>       ;     ESI = user's buffer address
                                <1>       ;     EDI = destination address in system space
                                <1>       ;     ECX = transfer (byte) count
                                <1>       ;     [u.pgdir] = user's page directory
                                <1>       ; OUTPUT ->
                                <1>       ;     ecx = actual transfer count
                                <1>       ;     cf = 1 -> error
                                <1>       ;     [u.count] = remain byte count
                                <1>       ;
                                <1>       ; Modified registers: eax, ecx
                                <1>       ;
                                <1>
0000E8F7 21C9                   <1>       and   ecx, ecx
```

```
42841                                 <1>       ;jz    short tfub_4
42842 0000E8F9 74F1                   <1>       jz     short ttub_4
42843                                 <1>
42844 0000E8FB 890D[88030300]         <1>       mov    [u.count], ecx
42845                                 <1>
42846 0000E901 57                     <1>       push   edi
42847 0000E902 56                     <1>       push   esi
42848 0000E903 53                     <1>       push   ebx
42849 0000E904 52                     <1>       push   edx
42850 0000E905 51                     <1>       push   ecx
42851                                 <1>
42852 0000E906 89F3                   <1>       mov    ebx, esi
42853 0000E908 81C300004000           <1>       add    ebx, CORE ; 27/05/2016
42854                                 <1> tfub_1:
42855                                 <1>       ; ebx = virtual (linear) address
42856                                 <1>       ; [u.pgdir] = user's page directory
42857 0000E90E E88269FFFF             <1>             call   get_physical_addr_x ; get physical address
42858                                 <1>       ;jc    short tfub_5
42859 0000E913 72D8                   <1>       jc     short ttub_5
42860                                 <1>       ; eax = physical address
42861                                 <1>       ; ecx = remain byte count in page (1-4096)
42862 0000E915 89C6                   <1>       mov    esi, eax
42863 0000E917 A1[88030300]           <1>       mov    eax, [u.count]
42864 0000E91C 39C1                   <1>       cmp    ecx, eax
42865 0000E91E 7602                   <1>       jna    short tfub_2
42866 0000E920 89C1                   <1>       mov    ecx, eax
42867                                 <1> tfub_2:
42868 0000E922 29C8                   <1>       sub    eax, ecx
42869 0000E924 01CB                   <1>       add    ebx, ecx
42870 0000E926 F3A4                   <1>       rep    movsb
42871 0000E928 A3[88030300]           <1>       mov    [u.count], eax
42872 0000E92D 09C0                   <1>       or     eax, eax
42873 0000E92F 75DD                   <1>       jnz    short tfub_1
42874                                 <1>
42875 0000E931 EBB4                   <1>       jmp    short tfub_retn
42876                                 <1>
42877                                 <1> ;tfub_retn:
42878                                 <1> ;     pop    ecx ; transfer count = actual transfer count
42879                                 <1> ;tfub_3:
42880                                 <1> ;     pop    edx
42881                                 <1> ;     pop    ebx
42882                                 <1> ;     pop    esi
42883                                 <1> ;     pop    edi
42884                                 <1> ;tfub_4:
42885                                 <1> ;     retn
42886                                 <1> ;tfub_5:
42887                                 <1> ;     pop    ecx
42888                                 <1> ;     sub    ecx, [u.count] ; actual transfer count
42889                                 <1> ;     stc
42890                                 <1> ;     jmp    short tfub_3
42891                                 <1>
42892                                 <1> sysfff: ; <Find First File>
42893                                 <1>       ; 17/10/2016
42894                                 <1>       ; 16/10/2016
42895                                 <1>       ; 15/10/2016 TRDOS 386 (TRDOS v2.0) feature only !
42896                                 <1>       ;           -derived from TRDOS v1.0, INT_21H.ASM-
42897                                 <1>       ;            ("loc_INT21h_find_first_file")
42898                                 <1>       ; TRDOS 8086 (v1.0)
42899                                 <1>         ;   07/08/2011
42900                                 <1>         ;   Find First File
42901                                 <1>       ;     INPUT:
42902                                 <1>         ;       CX= Attributes
42903                                 <1>         ;       DS:DX= Pointer to filename
42904                                 <1>       ;     MSDOS OUTPUT:
42905                                 <1>       ;         DTA: (Default address: PSP offset 80h)
42906                                 <1>       ;         Offset  Descrription
42907                                 <1>       ;         0     Reserved for use find next file
42908                                 <1>       ;         21    Attribute of file found
42909                                 <1>       ;         22    Time stamp of file
42910                                 <1>       ;         24    Date stamp of file
42911                                 <1>       ;         26    File size in bytes
42912                                 <1>       ;         30    Filename and extension (zero terminated)
42913                                 <1>       ;     If cf = 1:
42914                                 <1>       ;         Error Codes: (in AX)
42915                                 <1>       ;           2 - File not found
42916                                 <1>       ;           18 - No more files
42917                                 <1>         ;
42918                                 <1>       ; TRDOS 386 (v2.0)
42919                                 <1>       ; 15/10/2016
42920                                 <1>       ;
42921                                 <1>         ; INPUT ->
42922                                 <1>         ;       CL = File attributes
42923                                 <1>       ;           bit 0 (1) - Read only file (R)
42924                                 <1>       ;            bit 1 (1) - Hidden file (H)
42925                                 <1>         ;            bit 2 (1) - System file (R)
42926                                 <1>       ;            bit 3 (1) - Volume label/name (V)
42927                                 <1>         ;            bit 4 (1) - Subdirectory (D)
42928                                 <1>       ;           bit 5 (1) - File has been archived (A)
42929                                 <1>       ;         CH = 0 -> Return basic parameters (24 bytes)
42930                                 <1>       ;         CH > 0 -> Return FindFile structure/table (128 bytes)
42931                                 <1>         ;         EBX = Pointer to filename (ASCIIZ) -path-
42932                                 <1>       ;         EDX = File parameters buffer address
42933                                 <1>       ;           (buffer size = 24 bytes if CH input = 0)
42934                                 <1>       ;           (buffer size = 128 bytes if CH input > 0)
42935                                 <1>       ;
42936                                 <1>       ; OUTPUT ->
42937                                 <1>       ;         EAX = 0 if CH input > 0
42938                                 <1>       ;         EAX = First cluster number of file if CH input = 0
42939                                 <1>       ;         EDX = File parameters table/structure address
42940                                 <1>       ;         Basic Parameters:
42941                                 <1>       ;           Offset  Description
42942                                 <1>       ;           ------ ---------------
42943                                 <1>       ;           0     File Attributes
```

```
42944                                  <1>       ;            1     Ambiguous filename chars are used sign
42945                                  <1>       ;                  (0 = filename fits exactly with request)
42946                                  <1>       ;                  (>0 = ambiguous filename chars are used)
42947                                  <1>       ;            2     Time stamp of file
42948                                  <1>       ;            4     Date stamp of file
42949                                  <1>       ;            6     File size in bytes
42950                                  <1>       ;            10    Short Filename (ASCIIZ, max. 13 bytes)
42951                                  <1>       ;            23    Longname Length (1-255) if existing
42952                                  <1>       ;
42953                                  <1>       ;         cf = 1 -> Error code in AL
42954                                  <1>       ;
42955                                  <1>       ; Modified Registers: EAX (at the return of system call)
42956                                  <1>       ;
42957                                  <1>       ; TR-DOS FindFile (FFF) Structure (128 bytes):
42958                                  <1>       ; 09/10/2011 (DIR.ASM) - 10/02/2016 (trdoskx.s)
42959                                  <1>       ;
42960                                  <1>       ; Offset     Parameter           Size
42961                                  <1>       ; ------     ------------------  --------
42962                                  <1>       ; 0          FindFile_Drv        1 byte
42963                                  <1>       ; 1          FindFile_Directory  65 bytes
42964                                  <1>       ; 66         FindFile_Name       13 bytes
42965                                  <1>       ; 79         FindFile_LongNameEntryLength 1 byte
42966                                  <1>       ;Above 80 bytes form
42967                                  <1>       ;TR-DOS Source/Destination File FullName Format/Structure
42968                                  <1>       ; 80         FindFile_AttributesMask 1 word
42969                                  <1>       ; 82         FindFile_DirEntry   32 bytes (*)
42970                                  <1>       ; 114        FindFile_DirFirstCluster 1 double word
42971                                  <1>       ; 118        FindFile_DirCluster 1 double word
42972                                  <1>       ; 122        FindFile_DirEntryNumber 1 word
42973                                  <1>       ; 124        FindFile_MatchCounter     1 word
42974                                  <1>       ; 126        FindFile_Reserved   1 word
42975                                  <1>       ; (*) MS-DOS, FAT 12-16-32 classic directory entry (32 bytes)
42976                                  <1>
42977                                  <1>       ;mov   [u.namep], ebx
42978                                  <1>       ; 16/10/2016
42979 0000E933 8915[D45F0100]          <1>       mov    [FFF_UBuffer], edx
42980 0000E939 66890D[D95F0100]        <1>       mov    [FFF_Attrib], cx ; [FFF_RType] = ch
42981                                  <1>                   ; Attributes in CL, return data type in CH
42982 0000E940 89DE                    <1>       mov    esi, ebx
42983                                  <1>       ; file name is forced, change directory as temporary
42984                                  <1>       ;mov   ax, 1
42985                                  <1>       ;mov   [FFF_Valid], ah ; 0 ; reset ; 17/10/2016
42986                                  <1>       ;call  set_working_path
42987 0000E942 E8B40C0000              <1>       call   set_working_path_x ; 17/10/2016
42988 0000E947 731D                    <1>       jnc    short sysfff_0
42989                                  <1>
42990 0000E949 21C0                    <1>       and    eax, eax  ; 0 -> Bad Path!
42991 0000E94B 7505                    <1>       jnz    short sysfff_err
42992                                  <1>
42993                                  <1>       ; eax = 0
42994 0000E94D B80C000000              <1>       mov    eax, ERR_DIR_NOT_FOUND ; Directory not found !
42995                                  <1> sysfff_err:
42996 0000E952 A3[64030300]            <1>       mov    [u.r0], eax
42997 0000E957 A3[C8030300]            <1>       mov    [u.error], eax
42998 0000E95C E86F0D0000              <1>         call       reset_working_path
42999 0000E961 E957DBFFFF              <1>       jmp    error
43000                                  <1>
43001                                  <1> sysfff_0:
43002                                  <1>       ;sub   ah, ah ; ah = 0
43003 0000E966 8A0424                  <1>       mov    al, [esp]
43004 0000E969 08C0                    <1>       or     al, al
43005 0000E96B 7412                    <1>       jz     short sysfff_2
43006 0000E96D B410                    <1>       mov    ah, 10h
43007 0000E96F A808                    <1>       test   al, 08h
43008 0000E971 7503                    <1>       jnz    short sysfff_1
43009 0000E973 80CC08                  <1>       or     ah, 08h
43010                                  <1> sysfff_1:
43011 0000E976 2410                    <1>       and    al, 10h ; Directory
43012 0000E978 7405                    <1>       jz     short sysfff_2
43013 0000E97A 80E408                  <1>       and    ah, 08h
43014 0000E97D 30C0                    <1>       xor    al, al ; When a directory is searched,
43015                                  <1>                   ; filename will be returned even if
43016                                  <1>                   ; it is not a directory!
43017                                  <1>                   ; Because: (in order to prevent
43018                                  <1>                   ; creating a dir with existing file name)
43019                                  <1>                   ; Dir and file names must not be same!
43020                                  <1>                   ; (return attribute must be checked)
43021                                  <1> sysfff_2:
43022                                  <1>       ; AX = Attributes mask
43023                                  <1>             ; AL = AND mask (result must be equal to AL)
43024                                  <1>             ; AH = Negative AND mask (result must be ZERO)
43025                                  <1>       ; ESI = FindFile_Name address
43026                                  <1>
43027 0000E97F E88796FFFF              <1>       call   find_first_file
43028 0000E984 72CC                    <1>       jc     short sysfff_err ; eax = 2 (File not found !)
43029                                  <1>
43030                                  <1>       ; ESI = Directory Entry (FindFile_DirEntry) Location
43031                                  <1>       ; EDI = Directory Buffer Directory Entry Location
43032                                  <1>       ; EAX = File Size
43033                                  <1>       ;  BL = Attributes of The File/Directory
43034                                  <1>       ;  BH = Long Name Yes/No Status (>0 is YES)
43035                                  <1>       ;  DX > 0 : Ambiguous filename chars are used
43036                                  <1>
43037                                  <1> sysfff_3:
43038                                  <1>       ; 16/10/2016
43039 0000E986 668B0D[D95F0100]        <1>       mov    cx, [FFF_Attrib]
43040                                  <1>       ; Attribs in CL, return data type in CH
43041                                  <1>
43042                                  <1>       ;or    cl, cl
43043                                  <1>       ;jz    short sysfff_4 ; 0 = No filter
43044 0000E98D 80F1FF                  <1>       xor    cl, 0FFh
43045 0000E990 20D9                    <1>       and    cl, bl
43046 0000E992 7409                    <1>       jz     short sysfff_4
```

```
43047                                  <1>
43048                                  <1>       ;mov   eax, 2 ; 'file not found !' error
43049                                  <1>       ;jmp   short sysfff_err_1
43050                                  <1>
43051                                  <1>       ; 16/10/2016
43052 0000E994 E82197FFFF              <1>       call   find_next_file
43053 0000E999 72B7                    <1>       jc     short sysfff_err ; eax = 12 (no more files !)
43054 0000E99B EBE9                    <1>       jmp    short sysfff_3
43055                                  <1>
43056                                  <1> sysfff_4:
43057 0000E99D 20ED                    <1>       and    ch, ch ; [FFF_RType]
43058 0000E99F 7412                    <1>       jz     short sysfff_5
43059 0000E9A1 B980000000              <1>       mov    ecx, 128 ; ; transfer length
43060 0000E9A6 880D[D85F0100]          <1>       mov    [FFF_Valid], cl
43061                                  <1> sysfnf_11:
43062 0000E9AC BE[8A5C0100]            <1>       mov    esi, FindFile_Drv
43063 0000E9B1 EB43                    <1>       jmp    short sysfff_6
43064                                  <1> sysfff_5:
43065                                  <1>       ;mov   esi, FindFile_DirEntry
43066 0000E9B3 B918000000              <1>       mov    ecx, 24  ; transfer length
43067 0000E9B8 880D[D85F0100]          <1>       mov    [FFF_Valid], cl
43068                                  <1> sysfnf_12:
43069 0000E9BE BF[94640100]            <1>       mov    edi, DTA ; FFF data transfer address
43070                                  <1>       ;mov   al, [esi+DirEntry_Attr] ; 11
43071 0000E9C3 88D8                    <1>       mov    al, bl ; File/Dir Attributes
43072 0000E9C5 887F17                  <1>       mov    [edi+23], bh ; Longname length (0= none)
43073 0000E9C8 AA                      <1>       stosb
43074 0000E9C9 88D0                    <1>       mov    al, dl ; DL is for '?'
43075 0000E9CB 00F0                    <1>       add    al, dh ; DH is for '*'
43076                                  <1>       ; AL > 0 if ambiguous file name wildcards are used
43077 0000E9CD AA                      <1>       stosb
43078 0000E9CE 8B4616                  <1>       mov    eax, [esi+DirEntry_WrtTime] ; 22
43079 0000E9D1 AB                      <1>         stosd      ; DirEntry_WrtTime & DirEntry_WrtDate
43080 0000E9D2 8B461C                  <1>         mov  eax, [esi+DirEntry_FileSize] ; 28
43081 0000E9D5 AB                      <1>         stosd
43082 0000E9D6 668B4614                <1>       mov    ax, [esi+DirEntry_FstClusHI] ; 20
43083 0000E9DA C1E010                  <1>       shl    eax, 16 ; 13/11/2017
43084 0000E9DD 668B461A                <1>       mov    ax, [esi+DirEntry_FstClusLO] ; 26
43085 0000E9E1 A3[64030300]            <1>       mov    [u.r0], eax ; First Cluster
43086                                  <1>
43087                                  <1>         ;mov esi, FindFile_DirEntry
43088 0000E9E6 E8220D0000              <1>       call   get_file_name
43089                                  <1>
43090 0000E9EB 8A0D[D85F0100]          <1>       mov    cl, [FFF_Valid]
43091 0000E9F1 BE[94640100]            <1>              mov    esi, DTA ; FFF data transfer address
43092                                  <1> sysfff_6:
43093 0000E9F6 8B3D[D45F0100]          <1>       mov    edi, [FFF_UBuffer] ; user's buffer address (edx)
43094 0000E9FC E8ACFEFFFF              <1>       call   transfer_to_user_buffer
43095                                  <1>
43096 0000EA01 890D[64030300]          <1>       mov    [u.r0], ecx ; actual transfer count
43097 0000EA07 E8C40C0000              <1>         call       reset_working_path
43098 0000EA0C E9CCDAFFFF              <1>       jmp    sysret
43099                                  <1>
43100                                  <1> sysfnf: ; <Find Next File>
43101                                  <1>       ; 13/11/2017
43102                                  <1>       ; 16/10/2016 TRDOS 386 (TRDOS v2.0) feature only !
43103                                  <1>       ;           -derived from TRDOS v1.0, INT_21H.ASM-
43104                                  <1>       ;            ("loc_INT21h_find_next_file")
43105                                  <1>       ; TRDOS 8086 (v1.0)
43106                                  <1>         ;   07/08/2011
43107                                  <1>         ;   Find First File
43108                                  <1>       ;     INPUT:
43109                                  <1>         ;       none
43110                                  <1>       ;     MSDOS OUTPUT:
43111                                  <1>       ;         DTA: (Default address: PSP offset 80h)
43112                                  <1>       ;         Offset  Descrription
43113                                  <1>       ;         0     Reserved for use find next file
43114                                  <1>       ;         21    Attribute of file found
43115                                  <1>       ;         22    Time stamp of file
43116                                  <1>       ;         24    Date stamp of file
43117                                  <1>       ;         26    File size in bytes
43118                                  <1>       ;         30    Filename and extension (zero terminated)
43119                                  <1>       ;     If cf = 1:
43120                                  <1>       ;         Error Codes: (in AX)
43121                                  <1>       ;            18 - No more files
43122                                  <1>         ;
43123                                  <1>       ; TRDOS 386 (v2.0)
43124                                  <1>       ; 16/10/2016
43125                                  <1>       ;
43126                                  <1>         ; INPUT ->
43127                                  <1>             ;        none
43128                                  <1>       ; OUTPUT ->
43129                                  <1>       ;        EAX = 0 if CH input of 'Find First File' > 0
43130                                  <1>       ;        EAX = First cluster number of file
43131                                  <1>       ;             if CH input of 'Find First File' = 0
43132                                  <1>       ;        EDX = File parameters table/structure address
43133                                  <1>       ;
43134                                  <1>       ;         cf = 1 -> Error code in AL
43135                                  <1>       ;
43136                                  <1>       ; Modified Registers: EAX (at the return of system call)
43137                                  <1>
43138                                  <1>       ;
43139                                  <1>       ; Note: If byte [FFF_Valid] = 0
43140                                  <1>       ;     'sysfnf' will return with 'no more files' error.
43141                                  <1>       ;     If byte [FFF_Valid] = 24
43142                                  <1>       ;     'sysfnf' will return with 32 bytes basic parameters
43143                                  <1>       ;     at the address which is in EDX.
43144                                  <1>       ;     If byte [FFF_Valid] = 128
43145                                  <1>       ;     'sysfnf' will return with 128 bytes Find File
43146                                  <1>       ;     Structure/Table at the address which is in EDX.
43147                                  <1>
43148 0000EA11 803D[D85F0100]00        <1>       cmp    byte [FFF_Valid], 0
43149 0000EA18 7714                    <1>       ja     short stsfnf_0
```

```
                                <1>       ; 'no more files !' error
0000EA1A B80C000000             <1>       mov   eax, ERR_NO_MORE_FILES ; 12
0000EA1F A3[64030300]           <1>       mov   [u.r0], eax
0000EA24 A3[C8030300]           <1>       mov   [u.error], eax
0000EA29 E98FDAFFFF             <1>       jmp   error
                                <1> stsfnf_0:
                                <1>       ;cmp  byte [FFF_Valid], 128
                                <1>       ;je   short stsfnf_1
                                <1>       ;cmp  byte [FFF_Valid], 24
                                <1>       ;je   short stsfnf_1
                                <1>       ;mov  [FFF_Valid], 24 ; Default
                                <1> stsfnf_1:
0000EA2E 0FB61D[E6520100]       <1>       movzx ebx, byte [Current_Drv]
0000EA35 66891D[DE5F0100]       <1>       mov   [SWP_DRV], bx
0000EA3C 8A15[8A5C0100]         <1>       mov   dl, [FindFile_Drv]
0000EA42 38DA                   <1>       cmp   dl, bl
0000EA44 750B                   <1>       jne   short stsfnf_2
0000EA46 86FB                   <1>       xchg  bh, bl
0000EA48 BE00010900             <1>       mov   esi, Logical_DOSDisks
0000EA4D 01DE                   <1>       add   esi, ebx
0000EA4F EB0D                   <1>       jmp   short sysfnf_3
                                <1>
                                <1> stsfnf_2:
0000EA51 FE05[DF5F0100]         <1>       inc   byte [SWP_DRV_chg]
                                <1>
0000EA57 E81482FFFF             <1>       call  change_current_drive
0000EA5C 7245                   <1>       jc    short sysfnf_err_1 ; read error !
                                <1>                               ; (do not stop, because
                                <1>                               ; we don't have a
                                <1>                               ; 'no more files'
                                <1>                               ; -file not found- error,
                                <1>                               ; next sysfnf system call
                                <1>                               ; may solve the problem,
                                <1>                               ; after re-placing the disk)
                                <1> sysfnf_3:
0000EA5E A1[005D0100]           <1>       mov   eax, [FindFile_DirCluster]
0000EA63 21C0                   <1>       and   eax, eax
0000EA65 7550                   <1>       jnz   short sysfnf_6
                                <1>
0000EA67 803D[E5520100]02       <1>       cmp   byte [Current_FATType], 2
0000EA6E 772C                   <1>       ja    short sysfnf_err_0 ; invalid, we neeed to stop !?
0000EA70 803D[E5520100]01       <1>       cmp   byte [Current_FATType], 1
0000EA77 7223                   <1>       jb    short sysfnf_err_0 ; invalid, we neeed to stop !?
                                <1>
0000EA79 3805[105B0100]         <1>       cmp   byte [DirBuff_ValidData], al ; 0
0000EA7F 7608                   <1>       jna   short sysfnf_4
                                <1>
0000EA81 3B05[155B0100]         <1>       cmp   eax, [DirBuff_Cluster] ; 0 ?
0000EA87 745E                   <1>       je    short sysfnf_9
                                <1>
                                <1>       ;cmp  byte [Current_Dir_Level], 0
                                <1>         ;ja  short sysfnf_4
                                <1>         ;jna short sysfnf_9
                                <1>
                                <1> sysfnf_4:
0000EA89 FE05[DF5F0100]         <1>       inc   byte [SWP_DRV_chg]
0000EA8F E80ED0FFFF             <1>       call  load_FAT_root_directory
0000EA94 7351                   <1>       jnc   short sysfnf_9
                                <1>       ; eax = error code (17, 'drv not ready or read error')
0000EA96 EB0B                   <1>       jmp   short sysfnf_err_1 ; read error ! (no FNF stop)
                                <1>                               ; (if you want, try again,
                                <1>                               ;  after re-placing the disk)
                                <1> sysfnf_5:
0000EA98 3C0C                   <1>       cmp   al, 12 ; 'no more files' error
0000EA9A 7507                   <1>       jne   short sysfnf_err_1 ; (no FNF stop -sysfnf will try
                                <1>                               ;  to read the directory again,
                                <1>                               ;  if the user calls sysfnf
                                <1>                               ;  just after this error return-)
                                <1>       ; (FNF stop -sysfnf will not try
                                <1>       ;  to read the directory again-)
                                <1>
                                <1> sysfnf_err_0:
0000EA9C C605[D85F0100]00       <1>       mov   byte [FFF_Valid], 0 ; FNF stop sign
                                <1> sysfnf_err_1:
0000EAA3 A3[64030300]           <1>       mov   [u.r0], eax
0000EAA8 A3[C8030300]           <1>       mov   [u.error], eax
0000EAAD E81E0C0000             <1>       call  reset_working_path
0000EAB2 E906DAFFFF             <1>       jmp   error
                                <1>
                                <1> sysfnf_6:
0000EAB7 803D[105B0100]00       <1>       cmp   byte [DirBuff_ValidData], 0
0000EABE 7608                   <1>       jna   short sysfnf_7
                                <1>
0000EAC0 3B05[155B0100]         <1>       cmp   eax, [DirBuff_Cluster]
0000EAC6 741F                   <1>       je    short sysfnf_9
                                <1>
                                <1> sysfnf_7:
0000EAC8 FE05[DF5F0100]         <1>       inc   byte [SWP_DRV_chg]
0000EACE 803D[E5520100]01       <1>       cmp   byte [Current_FATType], 1
0000EAD5 7309                   <1>       jnb   short sysfnf_8
                                <1>
                                <1>       ; Singlix (TRFS) File System
                                <1>       ; (access via compatibility buffer)
0000EAD7 E88ED0FFFF             <1>       call  load_FS_sub_directory
0000EADC 7309                   <1>       jnc   short sysfnf_9
                                <1>
0000EADE EBC3                   <1>       jmp   short sysfnf_err_1 ; read error (no FNF stop)
                                <1>
                                <1> sysfnf_8:
0000EAE0 E848D0FFFF             <1>       call  load_FAT_sub_directory
0000EAE5 72BC                   <1>       jc    short sysfnf_err_1 ; read error (no FNF stop)
                                <1>
                                <1> sysfnf_9:
```

```
43253 0000EAE7 E8CE95FFFF          <1>       call   find_next_file
43254 0000EAEC 72AA                <1>       jc     short sysfnf_5
43255                              <1>
43256 0000EAEE A0[D95F0100]        <1>       mov    al, [FFF_Attrib]
43257                              <1>       ;or    al, al
43258                              <1>       ;jz    short sysfnf_10 ; 0 = No filter
43259 0000EAF3 34FF                <1>       xor    al, 0FFh
43260 0000EAF5 20D8                <1>       and    al, bl
43261 0000EAF7 75EE                <1>       jnz    short sysfnf_9 ; search for next file until
43262                              <1>                             ; an error return from
43263                              <1>                             ; find_next_file procedure
43264                              <1> sysfnf_10:
43265 0000EAF9 0FB60D[D85F0100]    <1>         movzx      ecx, byte [FFF_Valid]
43266 0000EB00 80F980              <1>       cmp    cl, 128 ; complete FindFile structure/table
43267 0000EB03 0F84A3FEFFFF        <1>       je     sysfnf_11
43268                              <1>       ;cmp   cl, 24  ; basic parameters
43269                              <1>       ;je    sysfnf_12
43270 0000EB09 E9B0FEFFFF          <1>       jmp    sysfnf_12
43271                              <1>
43272                              <1> writei:
43273                              <1>       ; 26/10/2016
43274                              <1>       ; 25/10/2016
43275                              <1>       ; 23/10/2016
43276                              <1>       ; 22/10/2016
43277                              <1>       ; 19/10/2016 - TRDOS 386 (TRDOS v2.0)
43278                              <1>       ; 19/05/2015 - 20/05/2015 (Retro UNIX 386 v1)
43279                              <1>       ; 12/03/2013 - 31/07/2013 (Retro UNIX 8086 v1)
43280                              <1>       ;
43281                              <1>       ; Write data to file with first cluster number in EAX
43282                              <1>       ;
43283                              <1>       ; INPUTS ->
43284                              <1>       ;    EAX - First cluster number of the file
43285                              <1>       ;    EBX - File number (Open file index number)
43286                              <1>       ;    u.count - byte count to be written
43287                              <1>       ;    u.base - points to user buffer
43288                              <1>       ;    u.fofp - points to dword with current file offset
43289                              <1>       ;    i.size - file size
43290                              <1>       ;    cdev - logical dos drive number of the file
43291                              <1>       ; OUTPUTS ->
43292                              <1>       ;    u.count - cleared
43293                              <1>       ;    u.nread - accumulates total bytes passed back
43294                              <1>       ;    i.size - new file size (if file byte offset overs file size)
43295                              <1>       ;    u.fofp - points to u.off (with new offset value)
43296                              <1>       ;
43297                              <1>       ; (Retro UNIX Prototype : 11/11/2012 - 18/11/2012, UNIXCOPY.ASM)
43298                              <1>       ; ((Modified registers: eax, edx, ebx, ecx, esi, edi, ebp))
43299                              <1>
43300 0000EB0E 31C9                <1>       xor    ecx, ecx
43301 0000EB10 890D[8C030300]      <1>       mov    [u.nread], ecx  ; 0
43302 0000EB16 66890D[C4030300]    <1>       mov    [u.pcount], cx ; 19/05/2015
43303 0000EB1D 390D[88030300]      <1>       cmp    [u.count], ecx
43304 0000EB23 7701                <1>       ja     short writei_1
43305 0000EB25 C3                  <1>       retn
43306                              <1> writei_1:
43307 0000EB26 881D[985F0100]      <1>       mov    [writei.ofn], bl ; Open file number
43308 0000EB2C 880D[D35F0100]      <1>       mov    [setfmod], cl ; 0 ; reset 'update lm date&time' sign
43309                              <1> dskw_0:
43310                              <1>       ; 26/10/2016
43311                              <1>       ; 22/10/2016, 23/10/2016, 25/10/2016
43312                              <1>       ; 19/10/2016 - TRDOS 386 (TRDOS v2.0)
43313                              <1>       ; 31/05/2015 - 25/07/2015 (Retro UNIX 386 v1)
43314                              <1>       ; 26/04/2013 - 20/09/2013 (Retro UNIX 8086 v1)
43315                              <1>       ;
43316                              <1>       ; 01/08/2013 (mkdir_w check)
43317 0000EB32 E8D7000000          <1>       call   mget_w
43318                              <1>       ; eax = sector/block number
43319                              <1>
43320 0000EB37 8B1D[74030300]      <1>       mov     ebx, [u.fofp]
43321 0000EB3D 8B13                <1>       mov    edx, [ebx]
43322 0000EB3F 81E2FF010000        <1>       and    edx, 1FFh  ; / test the lower 9 bits of the file offset
43323 0000EB45 750C                <1>       jnz    short dskw_1 ; / if its non-zero, branch
43324                              <1>                           ; if zero, file offset = 0,
43325                              <1>                                ; / 512, 1024,...(i.e., start of new block)
43326 0000EB47 813D[88030300]0002- <1>       cmp    dword [u.count], 512
43327 0000EB4F 0000                <1>
43328                              <1>                            ; / if zero, is there enough data to fill
43329                              <1>                            ; / an entire block? (i.e., no. of
43330 0000EB51 7337                <1>       jnb    short dskw_2 ; / bytes to be written greater than 512.?
43331                              <1>                           ; / Yes, branch. Don't have to read block
43332                              <1> dskw_1: ; in as no past info. is to be saved
43333                              <1>       ; (the entire block will be overwritten).
43334                              <1>       ; 23/10/2016
43335                              <1>
43336 0000EB53 BB[94070300]        <1>       mov    ebx, writei_buffer
43337                              <1>       ; esi = logical dos drive description table address
43338                              <1>       ; eax = sector number
43339                              <1>       ; ebx = buffer address (in kernel's memory space)
43340                              <1>       ; ecx = sector count
43341 0000EB58 B901000000          <1>       mov    ecx, 1
43342 0000EB5D E876060000          <1>       call   disk_read
43343                              <1>       ;call  dskrd  ; / no, must retain old info..
43344                              <1>                            ; / Hence, read block 'r1' into an I/O buffer
43345 0000EB62 7326                <1>       jnc    short dskw_2
43346                              <1>
43347                              <1>       ; disk read error
43348 0000EB64 B811000000          <1>       mov    eax, 17 ; drive not ready or READ ERROR !
43349                              <1> dskw_err: ; jump from disk write error
43350 0000EB69 A3[64030300]        <1>       mov    [u.r0], eax
43351 0000EB6E A3[C8030300]        <1>       mov    [u.error], eax
43352                              <1>
43353 0000EB73 803D[D35F0100]00    <1>       cmp    byte [setfmod], 0
43354 0000EB7A 0F863DD9FFFF        <1>       jna    error
43355                              <1>
```

```
43356 0000EB80 E8AF030000          <1>       call   update_file_lmdt ; update last modif. date&time of the file
43357                              <1>       ;mov   byte [setfmod], 0
43358                              <1>
43359 0000EB85 E933D9FFFF          <1>       jmp    error
43360                              <1>
43361                              <1> dskw_2: ; 3:
43362                              <1>       ; 23/10/2016
43363 0000EB8A C605[745F0100]01    <1>       mov    byte [writei.valid], 1 ; writei buffer contains valid data
43364 0000EB91 56                  <1>       push   esi ; logical dos drive description table address
43365                              <1>       ; EAX (r1) = block/sector number
43366                              <1>       ;call  wslot
43367                              <1>              ; jsr r0,wslot / set write and inhibit bits in I/O queue,
43368                              <1>                       ; / proc. status=0, r5 points to 1st word of data
43369 0000EB92 803D[C6030300]00    <1>       cmp    byte [u.kcall], 0
43370 0000EB99 770F                <1>       ja     short dskw_4 ; zf=0 -> the caller is 'mkdir'
43371                              <1>       ;
43372 0000EB9B 66833D[C4030300]00  <1>       cmp    word [u.pcount], 0
43373 0000EBA3 7705                <1>       ja     short dskw_4
43374                              <1> dskw_3:
43375                              <1>       ; [u.base] = virtual address to transfer (as source address)
43376 0000EBA5 E813FAFFFF          <1>       call   trans_addr_r ; translate virtual address to physical (r)
43377                              <1> dskw_4:
43378 0000EBAA BB[94070300]        <1>       mov    ebx, writei_buffer
43379                              <1>       ; EBX (r5) = system (I/O) buffer address
43380 0000EBAF E875FAFFFF          <1>       call   sioreg
43381                              <1>       ; ESI = file (user data) offset
43382                              <1>       ; EDI = sector (I/O) buffer offset
43383                              <1>       ; ECX = byte count
43384                              <1>       ;
43385 0000EBB4 F3A4                <1>       rep    movsb
43386                              <1>       ; 25/07/2015
43387                              <1>       ; eax = remain bytes in buffer
43388                              <1>         ;       (check if remain bytes in the buffer > [u.pcount])
43389 0000EBB6 09C0                <1>       or     eax, eax
43390 0000EBB8 75EB                <1>       jnz    short dskw_3 ; (page end before system buffer end!)
43391                              <1>
43392                              <1>       ; 23/10/2016
43393 0000EBBA B101                <1>       mov    cl, 1
43394 0000EBBC 5E                  <1>       pop    esi
43395 0000EBBD A1[785F0100]        <1>       mov    eax, [writei.sector]
43396                              <1>       ; esi = logical dos drive description table address
43397                              <1>       ; eax = sector number
43398                              <1>       ; ebx = writei buffer address
43399                              <1>       ; ecx = sector count
43400 0000EBC2 E802060000          <1>       call   disk_write ; / yes, write the block
43401 0000EBC7 7307                <1>       jnc    short dskw_5
43402                              <1>
43403 0000EBC9 B812000000          <1>       mov    eax, 18 ; drive not ready or WRITE ERROR !
43404 0000EBCE EB99                <1>       jmp    short dskw_err
43405                              <1>
43406                              <1> dskw_5:
43407                              <1>       ; 26/10/2016
43408 0000EBD0 0FB61D[985F0100]    <1>       movzx  ebx, byte [writei.ofn] ; open file number
43409 0000EBD7 C0E302              <1>       shl    bl, 2 ; *4
43410 0000EBDA 8B83[68630100]      <1>       mov    eax, [ebx+OF_POINTER]
43411 0000EBE0 3B83[90630100]      <1>       cmp    eax, [ebx+OF_SIZE]
43412 0000EBE6 7606                <1>       jna    short dskw_6
43413 0000EBE8 8983[90630100]      <1>       mov    [ebx+OF_SIZE], eax
43414                              <1> dskw_6:
43415                              <1>       ;shr   bl, 2
43416 0000EBEE 833D[88030300]00    <1>         cmp     dword [u.count], 0 ; / any more data to write?
43417 0000EBF5 760A                <1>       jna    short dskw_7
43418 0000EBF7 A1[885F0100]        <1>       mov    eax, [writei.fclust]
43419 0000EBFC E931FFFFFF          <1>       jmp    dskw_0 ; / yes, branch
43420                              <1> dskw_7:
43421                              <1>       ; update last modif. date&time of the file
43422                              <1>       ; (also updates file size as OF_SIZE)
43423 0000EC01 E82E030000          <1>       call   update_file_lmdt
43424                              <1>       ;mov   byte [setfmod], 0
43425                              <1>
43426                              <1>       ; 03/08/2013
43427 0000EC06 C605[C6030300]00    <1>       mov    byte [u.kcall], 0
43428                              <1>       ; 23/10/2016
43429                              <1>       ;mov   eax, [writei.fclust]
43430 0000EC0D C3                  <1>       retn
43431                              <1>
43432                              <1> mget_w:
43433                              <1>       ; 02/11/2016
43434                              <1>       ; 01/11/2016
43435                              <1>       ; 23/10/2016, 31/10/2016
43436                              <1>       ; 22/10/2016 - TRDOS 386 (TRDOS v2.0)
43437                              <1>       ; 03/06/2015 (Retro UNIX 386 v1, 'mget', u.5s)
43438                              <1>       ; 22/03/2013 - 31/07/2013 (Retro UNIX 8086 v1)
43439                              <1>       ;
43440                              <1>       ; Get existing or (allocate) a new disk block for file
43441                              <1>       ;
43442                              <1>       ; INPUTS ->
43443                              <1>       ;    [u.fofp] = file offset pointer
43444                              <1>       ;    [i.size] = file size
43445                              <1>       ;    [u.count] = byte count
43446                              <1>       ;    EAX = First cluster
43447                              <1>       ;    [cdev] = Logical dos drive number
43448                              <1>       ;    [writei.ofn] = File Number
43449                              <1>       ;                (Open file index, 0 based)
43450                              <1>       ;    ([u.off] = file offset)
43451                              <1>       ; OUTPUTS ->
43452                              <1>       ;    EAX = logical sector number
43453                              <1>       ;    ESI = Logical Dos Drive Description Table address
43454                              <1>       ;
43455                              <1>       ; Modified registers: EDX, EBX, ECX, ESI, EDI, EBP
43456                              <1>
43457 0000EC0E 8B35[74030300]      <1>         mov     esi, [u.fofp]
43458 0000EC14 8B2E                <1>       mov    ebp, [esi] ; u.off (or EBX*4+OF_POINTER)
```

```
43459                                  <1>
43460 0000EC16 29C9                    <1>       sub    ecx, ecx
43461 0000EC18 8A2D[46030300]          <1>       mov    ch, [cdev]
43462                                  <1>
43463 0000EC1E BE00010900              <1>       mov    esi, Logical_DOSDisks
43464 0000EC23 01CE                    <1>       add    esi, ecx
43465                                  <1>
43466                                  <1>       ; 31/10/2016
43467 0000EC25 89C3                    <1>       mov    ebx, eax ; First Cluster or FDT address
43468                                  <1>
43469 0000EC27 807E0300                <1>       cmp    byte [esi+LD_FATType], 0
43470 0000EC2B 0F86DD010000            <1>       jna    mget_w_14 ; Singlix FS
43471                                  <1>
43472 0000EC31 0FB74611                <1>       movzx  eax, word [esi+LD_BPB+BytesPerSec]
43473 0000EC35 0FB65613                <1>       movzx  edx, byte [esi+LD_BPB+SecPerClust]
43474 0000EC39 8815[765F0100]          <1>       mov    [writei.spc], dl  ; sectors per cluster
43475 0000EC3F F7E2                    <1>       mul    edx
43476                                  <1>       ; edx = 0
43477                                  <1>       ; eax = bytes per cluster (<= 65536)
43478                                  <1>
43479                                  <1>       ; 02/11/2016
43480 0000EC41 89C1                    <1>       mov    ecx, eax
43481 0000EC43 48                      <1>       dec    eax
43482 0000EC44 66A3[7C5F0100]          <1>       mov    [writei.bpc], ax
43483                                  <1>
43484 0000EC4A 89E8                    <1>       mov    eax, ebp
43485 0000EC4C 0305[88030300]          <1>       add    eax, [u.count] ; next file position
43486 0000EC52 3B05[55040300]          <1>       cmp    eax, [i.size] ; <= file size ?
43487 0000EC58 0F86FC000000            <1>       jna    mget_w_4 ; no
43488                                  <1>
43489 0000EC5E F7F1                    <1>       div    ecx
43490 0000EC60 A3[845F0100]            <1>       mov    [writei.c_index], eax ; cluster index
43491                                  <1>       ; edx = byte offset in cluster (<= 65535)
43492                                  <1>       ;mov   [writei.offset], dx
43493                                  <1>       ;shr   dx, 9 ; / 512
43494                                  <1>       ;mov   [writei.s_index], dl ; sector index in cluster (0 to spc -1)
43495                                  <1>
43496 0000EC65 29D2                    <1>       sub    edx, edx ; 01/11/2016
43497 0000EC67 8915[785F0100]          <1>       mov    [writei.sector], edx ; 0
43498 0000EC6D 668915[7E5F0100]        <1>       mov    [writei.offset], dx  ; byte offset in cluster
43499 0000EC74 8815[775F0100]          <1>       mov    [writei.s_index], dl ; sector index in cluster (0 to spc -1)
43500                                  <1>
43501 0000EC7A 89D8                    <1>       mov    eax, ebx ; First Cluster
43502                                  <1>
43503                                  <1>       ; is this the 1st mget_w or a next mget_w call ? (by 'writei')
43504 0000EC7C 3815[745F0100]          <1>       cmp    byte [writei.valid], dl ; 0
43505 0000EC82 7624                    <1>       jna    short mget_w_0
43506                                  <1>
43507 0000EC84 8815[745F0100]          <1>       mov    byte [writei.valid], dl ; 0 ; reset ('writei' will set it)
43508                                  <1>
43509 0000EC8A 3B05[885F0100]          <1>       cmp    eax, [writei.fclust]
43510 0000EC90 7516                    <1>       jne    short mget_w_0
43511                                  <1>
43512 0000EC92 8A0D[46030300]          <1>       mov    cl, [cdev]
43513 0000EC98 3A0D[755F0100]          <1>       cmp    cl, [writei.drv]
43514 0000EC9E 7508                    <1>       jne    short mget_w_0
43515                                  <1>       ; [writei.l_clust] & [writei.l_index] are valid,
43516                                  <1>       ;  we don't need to get last cluster & last cluster index
43517 0000ECA0 8B0D[945F0100]          <1>       mov    ecx, [writei.l_index]
43518 0000ECA6 EB64                    <1>       jmp    short mget_w_2
43519                                  <1> mget_w_0:
43520 0000ECA8 A3[885F0100]            <1>       mov    [writei.fclust], eax ; first cluster
43521                                  <1>       ; edx = 0
43522 0000ECAD A3[805F0100]            <1>       mov    [writei.cluster], eax ; first cluster ; 01/11/2016
43523 0000ECB2 8915[8C5F0100]          <1>       mov    [writei.fs_index], edx ; 0 ; curret cluster index
43524                                  <1>
43525                                  <1>       ; FAT file system (FAT12, FAT16, FAT32)
43526 0000ECB8 E885D4FFFF              <1>       call   get_last_cluster
43527 0000ECBD 0F822B010000            <1>       jc     mget_w_err ; eax = error code
43528                                  <1>
43529 0000ECC3 A3[905F0100]            <1>       mov    [writei.lclust], eax ; last cluster
43530                                  <1>
43531 0000ECC8 8B0D[B45D0100]          <1>       mov    ecx, [glc_index] ; last cluster index
43532 0000ECCE 890D[945F0100]          <1>       mov    [writei.l_index], ecx
43533                                  <1>
43534 0000ECD4 A0[985F0100]            <1>       mov    al, [writei.ofn]
43535 0000ECD9 FEC0                    <1>       inc    al
43536 0000ECDB A2[D35F0100]            <1>       mov    [setfmod], al ; update lm date&time sign
43537                                  <1>
43538                                  <1> mget_w_1:
43539 0000ECE0 3B0D[845F0100]          <1>       cmp    ecx, [writei.c_index]  ; last cluster index
43540 0000ECE6 7324                    <1>       jnb    short mget_w_2 ; 01/11/2016
43541                                  <1>
43542 0000ECE8 A1[905F0100]            <1>       mov    eax, [writei.lclust]
43543                                  <1>       ; EAX = Last cluster
43544 0000ECED E85ED5FFFF              <1>       call   add_new_cluster
43545 0000ECF2 0F82F6000000            <1>       jc     mget_w_err ; eax = error code
43546                                  <1>       ; edx = 0
43547 0000ECF8 A3[905F0100]            <1>       mov    [writei.lclust], eax ; (new) last cluster
43548 0000ECFD 8B0D[945F0100]          <1>       mov    ecx, [writei.l_index]
43549 0000ED03 41                      <1>       inc    ecx ; add 1 to last cluster index
43550 0000ED04 890D[945F0100]          <1>       mov    [writei.l_index], ecx ; current last cluster index
43551                                  <1>
43552 0000ED0A EBD4                    <1>       jmp    short mget_w_1
43553                                  <1>
43554                                  <1> mget_w_2:
43555 0000ED0C 89E9                    <1>       mov    ecx, ebp
43556 0000ED0E 030D[88030300]          <1>       add    ecx, [u.count]
43557 0000ED14 890D[55040300]          <1>       mov    [i.size], ecx ; save new file size
43558                                  <1>       ;sub   edx, edx ; 0
43559                                  <1>
43560 0000ED1A A0[46030300]            <1>       mov    al, [cdev]
43561 0000ED1F A2[755F0100]            <1>       mov    [writei.drv], al ; physical drive number
```

```
43562                                  <1>       ; edx = 0
43563 0000ED24 89E8                    <1>       mov    eax, ebp ; file offset
43564 0000ED26 0FB70D[7C5F0100]        <1>       movzx  ecx, word [writei.bpc] ; bytes per cluster - 1
43565 0000ED2D 41                      <1>       inc    ecx ; bytes per cluster
43566 0000ED2E F7F1                    <1>       div    ecx
43567                                  <1>       ; edx = byte offset in cluster (<= 65535)
43568                                  <1>       ; eax = cluster index
43569 0000ED30 A3[845F0100]            <1>       mov    [writei.c_index], eax
43570 0000ED35 668915[7E5F0100]        <1>       mov    [writei.offset], dx
43571 0000ED3C 66C1EA09                <1>       shr    dx, 9 ; / 512
43572 0000ED40 8815[775F0100]          <1>       mov    [writei.s_index], dl ; sector index in cluster (0 to spc -1)
43573                                  <1>
43574                                  <1> mget_w_3:
43575 0000ED46 3B05[945F0100]          <1>       cmp    eax, [writei.l_index] ; last cluster index
43576 0000ED4C 752A                    <1>       jne    short mget_w_5
43577                                  <1>
43578 0000ED4E A3[8C5F0100]            <1>       mov    [writei.fs_index], eax ; cluster index (for next check)
43579 0000ED53 A1[905F0100]            <1>       mov    eax, [writei.lclust] ; last cluster
43580 0000ED58 EB60                    <1>       jmp    short mget_w_10
43581                                  <1>
43582                                  <1> mget_w_4: ; 02/11/2016
43583                                  <1>       ; eax = next file position
43584 0000ED5A 2B05[88030300]          <1>       sub    eax, [u.count] ; current file position
43585                                  <1>       ; edx = 0
43586                                  <1>       ; ecx = bytes per cluster
43587 0000ED60 F7F1                    <1>       div    ecx
43588 0000ED62 A3[845F0100]            <1>       mov    [writei.c_index], eax ; cluster index
43589 0000ED67 668915[7E5F0100]        <1>       mov    [writei.offset], dx
43590 0000ED6E 66C1EA09                <1>       shr    dx, 9 ; / 512
43591 0000ED72 8815[775F0100]          <1>       mov    [writei.s_index], dl ; sector index in cluster (0 to spc -1)
43592                                  <1>
43593                                  <1> mget_w_5:
43594 0000ED78 21C0                    <1>       and    eax, eax ; 0 = First Cluster's index number
43595 0000ED7A 750C                    <1>       jnz    short mget_w_6
43596                                  <1>
43597 0000ED7C A3[8C5F0100]            <1>       mov    [writei.fs_index], eax ; cluster index (for next check)
43598 0000ED81 A1[885F0100]            <1>       mov    eax, [writei.fclust] ; first cluster
43599 0000ED86 EB32                    <1>       jmp    short mget_w_10
43600                                  <1>
43601                                  <1> mget_w_6:
43602 0000ED88 3B05[8C5F0100]          <1>       cmp    eax, [writei.fs_index] ; current cluster index (>0)
43603 0000ED8E 7507                    <1>       jne    short mget_w_7
43604 0000ED90 A1[805F0100]            <1>       mov    eax, [writei.cluster] ; current cluster
43605 0000ED95 EB3A                    <1>       jmp    short mget_w_11
43606                                  <1>
43607                                  <1> mget_w_7:
43608 0000ED97 89C1                    <1>       mov    ecx, eax
43609 0000ED99 2B0D[8C5F0100]          <1>       sub    ecx, [writei.fs_index]
43610 0000ED9F 730D                    <1>       jnc    short mget_w_8
43611                                  <1>       ; get cluster by index from the first cluster
43612 0000EDA1 A1[885F0100]            <1>       mov    eax, [writei.fclust]
43613 0000EDA6 8B0D[845F0100]          <1>       mov    ecx, [writei.c_index]
43614 0000EDAC EB05                    <1>       jmp    short mget_w_9
43615                                  <1>
43616                                  <1> mget_w_8:
43617 0000EDAE A1[805F0100]            <1>       mov    eax, [writei.cluster] ; beginning cluster
43618                                  <1>       ; ecx = cluster sequence number after the beginning cluster
43619                                  <1>       ; sub  edx, edx ; 0
43620                                  <1>
43621                                  <1> mget_w_9:
43622                                  <1>       ; EAX = Beginning cluster
43623                                  <1>       ; EDX = Sector index in disk/file section
43624                                  <1>       ;     (Only for SINGLIX file system!)
43625                                  <1>       ; ECX = Cluster sequence number after the beginning cluster
43626                                  <1>       ; ESI = Logical DOS Drive Description Table address
43627 0000EDB3 E89ED5FFFF              <1>       call   get_cluster_by_index
43628 0000EDB8 7234                    <1>       jc     short mget_w_err ; error code in EAX
43629                                  <1>       ; EAX = Cluster number
43630                                  <1> mget_w_10:
43631 0000EDBA A3[805F0100]            <1>       mov    [writei.cluster], eax ; FDT number for Singlix File System
43632                                  <1>
43633 0000EDBF 807E0300                <1>       cmp    byte [esi+LD_FATType], 0
43634 0000EDC3 7638                    <1>       jna    short mget_w_13
43635                                  <1>       ; 01/11/2016
43636 0000EDC5 8B15[845F0100]          <1>       mov    edx, [writei.c_index]
43637 0000EDCB 8915[8C5F0100]          <1>       mov    [writei.fs_index], edx
43638                                  <1> mget_w_11:
43639 0000EDD1 83E802                  <1>       sub    eax, 2
43640 0000EDD4 0FB615[765F0100]        <1>       movzx  edx, byte [writei.spc]
43641 0000EDDB F7E2                    <1>       mul    edx
43642                                  <1>
43643 0000EDDD 034668                  <1>       add    eax, [esi+LD_DATABegin]
43644 0000EDE0 8A15[775F0100]          <1>       mov    dl, [writei.s_index]
43645 0000EDE6 01D0                    <1>       add    eax, edx
43646                                  <1> mget_w_12:
43647 0000EDE8 A3[785F0100]            <1>       mov    [writei.sector], eax
43648                                  <1>       ;; buffer validation must be done in writei
43649                                  <1>       ;;mov  byte [writei.valid], 1
43650 0000EDED C3                      <1>       retn
43651                                  <1>
43652                                  <1> mget_w_err:
43653 0000EDEE A3[C8030300]            <1>       mov    [u.error], eax
43654 0000EDF3 A3[64030300]            <1>       mov    [u.r0], eax
43655 0000EDF8 E9C0D6FFFF              <1>       jmp    error
43656                                  <1>
43657                                  <1> mget_w_13:
43658                                  <1>       ; EAX = FDT number (Current Section)
43659                                  <1>       ; EDX = Sector index from the first section (0,1,2,3,4...)
43660 0000EDFD 2B15[8C5F0100]          <1>       sub    edx, [writei.fs_index]
43661                                  <1>       ; EDX = Sector index from current section
43662 0000EE03 8915[8C5F0100]          <1>       mov    [writei.fs_index], edx
43663 0000EE09 40                      <1>       inc    eax ; the first data sector in FS disk section
43664 0000EE0A 01D0                    <1>       add    eax, edx
```

```
43665 0000EE0C EBDA               <1>       jmp    short mget_w_12
43666                             <1>
43667                             <1> mget_w_14:
43668 0000EE0E 8A4E12             <1>       mov    cl, [esi+LD_FS_BytesPerSec+1]
43669 0000EE11 D0E9               <1>       shr    cl, 1 ;  ; 1 for 512 bytes, 4 for 2048 bytes
43670 0000EE13 880D[765F0100]     <1>       mov    [writei.spc], cl  ; sectors per cluster
43671                             <1>       ; NOTE: writei bytes per sector value is always 512 !
43672 0000EE19 66C705[7C5F0100]00- <1>      mov    word [writei.bpc], 512
43673 0000EE21 02                 <1>
43674                             <1>
43675 0000EE22 89E9               <1>       mov    ecx, ebp
43676 0000EE24 030D[88030300]     <1>       add    ecx, [u.count] ; next file position
43677 0000EE2A 3B0D[55040300]     <1>       cmp    ecx, [i.size] ; <= file size ?
43678 0000EE30 0F86C8000000       <1>       jna    mget_w_19 ; no
43679                             <1>
43680 0000EE36 29D2               <1>       sub    edx, edx ; 0
43681 0000EE38 8915[785F0100]     <1>       mov    [writei.sector], edx ; 0
43682 0000EE3E 668915[7E5F0100]   <1>       mov    [writei.offset], dx  ; byte offset in cluster
43683 0000EE45 8815[775F0100]     <1>       mov    [writei.s_index], dl ; sector index in cluster (0 to spc -1)
43684                             <1>
43685 0000EE4B C1E909             <1>       shr    ecx, 9 ; 1 cluster = 512 bytes
43686 0000EE4E 890D[845F0100]     <1>       mov    [writei.c_index], ecx ; section/cluster index
43687                             <1>
43688 0000EE54 89D8               <1>       mov    eax, ebx ; FDT number (First FDT address)
43689                             <1>
43690                             <1>       ; is this the 1st mget_w or a next mget_w call ? (by 'writei')
43691 0000EE56 3815[745F0100]     <1>       cmp    byte [writei.valid], dl ; 0
43692 0000EE5C 7624               <1>       jna    short mget_w_15
43693                             <1>
43694 0000EE5E 8815[745F0100]     <1>       mov    byte [writei.valid], dl ; 0 ; reset ('writei' will set it)
43695                             <1>
43696 0000EE64 3B05[885F0100]     <1>       cmp    eax, [writei.fclust]
43697 0000EE6A 7516               <1>       jne    short mget_w_15
43698                             <1>
43699 0000EE6C 8A0D[46030300]     <1>       mov    cl, [cdev]
43700 0000EE72 3A0D[755F0100]     <1>       cmp    cl, [writei.drv]
43701 0000EE78 7508               <1>       jne    short mget_w_15
43702                             <1>       ; [writei.l_clust] & [writei.l_index] are valid,
43703                             <1>       ;  we don't need to get last cluster & last cluster index
43704 0000EE7A 8B0D[945F0100]     <1>       mov    ecx, [writei.l_index]
43705 0000EE80 EB49               <1>       jmp    short mget_w_17
43706                             <1> mget_w_15:
43707 0000EE82 A3[885F0100]       <1>       mov    [writei.fclust], eax ; first section (FDT number)
43708                             <1>       ; edx = 0
43709 0000EE87 8915[805F0100]     <1>       mov    [writei.cluster], edx ; 0 ; current section
43710 0000EE8D 8915[8C5F0100]     <1>       mov    [writei.fs_index], edx ; 0 ; curret section index
43711                             <1>
43712                             <1>       ; eax = FDT number (section 0 header address)
43713 0000EE93 E8E8D4FFFF         <1>       call   get_last_section
43714 0000EE98 0F8250FFFFFF       <1>       jc     mget_w_err ; eax = error code
43715                             <1>
43716 0000EE9E 8915[8C5F0100]     <1>       mov    [writei.fs_index], edx ; sector index in last section
43717                             <1>
43718 0000EEA4 A3[905F0100]       <1>       mov    [writei.lclust], eax ; last section address
43719                             <1>
43720 0000EEA9 8B0D[B45D0100]     <1>       mov    ecx, [glc_index] ; last section index
43721 0000EEAF 890D[945F0100]     <1>       mov    [writei.l_index], ecx
43722                             <1>
43723 0000EEB5 A0[985F0100]       <1>       mov    al, [writei.ofn]
43724 0000EEBA FEC0               <1>       inc    al
43725 0000EEBC A2[D35F0100]       <1>       mov    [setfmod], al ; update lm date&time sign
43726                             <1>
43727                             <1> mget_w_16:
43728                             <1>       ; edx = (existing) last section (sector) index
43729 0000EEC1 8B0D[845F0100]     <1>       mov    ecx, [writei.c_index] ; final section (sector) index
43730 0000EEC7 29D1               <1>       sub    ecx, edx
43731 0000EEC9 7633               <1>       jna    short mget_w_19
43732                             <1>        ; ecx = sector count
43733                             <1> mget_w_17:
43734 0000EECB A1[905F0100]       <1>       mov    eax, [writei.lclust]
43735                             <1>       ; ESI = Logical dos drv desc. table address
43736                             <1>         ; EAX = Last section
43737                             <1>         ; (ECX = 0 for directory)
43738                             <1>         ; ECX = sector count (except FDT)
43739 0000EED0 E86ECAFFFF         <1>       call   add_new_fs_section
43740 0000EED5 7312               <1>       jnc    short mget_w_18
43741                             <1>
43742                             <1>       ; If error number = 27h (insufficient disk space)
43743                             <1>       ; it is needed to check free consequent sectors
43744                             <1>       ; (1 data sector at least and +1 section header sector)
43745                             <1>
43746 0000EED7 83F827             <1>       cmp    eax, 27h
43747 0000EEDA 0F850EFFFFFF       <1>       jne    mget_w_err ; eax = error code
43748                             <1>
43749                             <1>       ; ecx = count of free consequent sectors
43750                             <1>       ; ecx must be > 1 (1 data + 1 header sector)
43751 0000EEE0 49                 <1>       dec    ecx
43752 0000EEE1 0F8407FFFFFF       <1>       jz     mget_w_err
43753 0000EEE7 EBE2               <1>       jmp    short mget_w_17
43754                             <1>
43755                             <1> mget_w_18:
43756 0000EEE9 A3[905F0100]       <1>       mov    [writei.lclust], eax ; (new) last section
43757                             <1>       ; ecx = sector count (except section header)
43758 0000EEEE 8B15[945F0100]     <1>       mov    edx, [writei.l_index]
43759 0000EEF4 01CA               <1>       add    edx, ecx ; add sector count to index
43760 0000EEF6 8915[945F0100]     <1>       mov    [writei.l_index], edx
43761 0000EEFC EBC3               <1>       jmp    short mget_w_16
43762                             <1>
43763                             <1> mget_w_19:
43764 0000EEFE 89E9               <1>       mov    ecx, ebp
43765 0000EF00 030D[88030300]     <1>       add    ecx, [u.count]
43766 0000EF06 890D[55040300]     <1>       mov    [i.size], ecx ; save new file size
43767                             <1>       ;sub   edx, edx ; 0
```

```
43768                                  <1>
43769 0000EF0C A0[46030300]            <1>       mov   al, [cdev]
43770 0000EF11 A2[755F0100]            <1>       mov   [writei.drv], al ; physical drive number
43771                                  <1>       ; edx = 0
43772 0000EF16 89E8                    <1>       mov   eax, ebp ; file offset
43773 0000EF18 89C2                    <1>       mov   edx, eax
43774                                  <1>       ; 1 cluster = 512 bytes (for Singlix FS)
43775 0000EF1A C1E809                  <1>       shr   eax, 9  ; / 512
43776 0000EF1D 81E2FF010000            <1>       and   edx, 1FFh
43777                                  <1>       ; edx = byte offset in cluster/sector (<= 511)
43778                                  <1>       ; eax = section (sector/cluster) index
43779 0000EF23 A3[845F0100]            <1>       mov   [writei.c_index], eax
43780 0000EF28 668915[7E5F0100]        <1>       mov   [writei.offset], dx
43781                                  <1>       ;mov  byte [writei.s_index], 0 ; sector index in cluster
43782 0000EF2F E912FEFFFF              <1>       jmp   mget_w_3
43783                                  <1>
43784                                  <1> update_file_lmdt: ; & update file size
43785                                  <1>       ; 26/10/2016
43786                                  <1>       ; 24/10/2016
43787                                  <1>       ; 23/10/2016
43788                                  <1>       ; 22/10/2016 - TRDOS 386 (TRDOS v2.0)
43789                                  <1>       ;
43790                                  <1>       ; Update last modification date&time of file
43791                                  <1>       ; (call from syswrite -> writei)
43792                                  <1>       ; ((also updates file size)) // 26/10/2016
43793                                  <1>       ;
43794                                  <1>       ; INPUT:
43795                                  <1>       ;     byte [setfmod] = open file number
43796                                  <1>       ; OUTPUT:
43797                                  <1>       ;     cf = 0 -> success !
43798                                  <1>       ;     cf = 1 -> lmdt update has been failed!
43799                                  <1>       ;
43800                                  <1>       ; Modified registers: eax, ebx, ecx, edx, esi, edi
43801                                  <1>       ;
43802                                  <1>
43803                                  <1>       ;cmp  byte [setfmod], 0
43804                                  <1>       ;jna  short uflmdt_2 ; nothing to do
43805                                  <1>
43806 0000EF34 31C0                    <1>       xor   eax, eax
43807                                  <1>
43808 0000EF36 0FB61D[D35F0100]        <1>       movzx ebx, byte [setfmod]
43809 0000EF3D FECB                    <1>       dec   bl ; open file index number (0 based)
43810                                  <1>
43811 0000EF3F 8AA3[40630100]          <1>       mov   ah, [ebx+OF_DRIVE]
43812 0000EF45 BE00010900              <1>       mov   esi, Logical_DOSDisks
43813 0000EF4A 01C6                    <1>       add   esi, eax
43814 0000EF4C C0E302                  <1>       shl   bl, 2 ; *4
43815 0000EF4F 8B8B[18630100]          <1>       mov   ecx, [ebx+OF_FCLUSTER] ; first cluster
43816 0000EF55 8B93[E0630100]          <1>       mov   edx, [ebx+OF_DIRCLUSTER] ; dir cluster
43817                                  <1>
43818 0000EF5B D0EB                    <1>       shr   bl, 1 ; /2
43819 0000EF5D 0FB7BB[80640100]        <1>       movzx edi, word [ebx+OF_DIRENTRY]
43820                                  <1>
43821 0000EF64 803D[105B0100]01        <1>       cmp   byte [DirBuff_ValidData], 1
43822 0000EF6B 726E                    <1>       jb    short uflmdt_4
43823                                  <1>
43824 0000EF6D A0[0E5B0100]            <1>       mov   al, [DirBuff_DRV]
43825 0000EF72 2C41                    <1>       sub   al, 'A'
43826 0000EF74 38E0                    <1>       cmp   al, ah
43827 0000EF76 7563                    <1>       jne   short uflmdt_4 ; different drive
43828 0000EF78 8A4603                  <1>       mov   al, [esi+LD_FATType]
43829 0000EF7B 3A05[0F5B0100]          <1>       cmp   al, [DirBuff_FATType]
43830 0000EF81 755B                    <1>       jne   short uflmdt_5 ; different FS type
43831 0000EF83 3B15[155B0100]          <1>       cmp   edx, [DirBuff_Cluster]
43832 0000EF89 7553                    <1>       jne   short uflmdt_5 ; different cluster
43833                                  <1>
43834                                  <1> uflmdt_1:
43835                                  <1>       ; Directory buffer is ready here!
43836                                  <1>       ; OF_FCLUSTER must be compared/verified
43837 0000EF8B BE00000800              <1>       mov   esi, Directory_Buffer
43838 0000EF90 66C1E705                <1>       shl   di, 5 ; dir entry index * 32
43839 0000EF94 01FE                    <1>       add   esi, edi ; offset
43840                                  <1>       ;
43841 0000EF96 F6460B18                <1>       test  byte [esi+DirEntry_Attr], 18h ; Vol & Dir
43842 0000EF9A 750F                    <1>       jnz   short uflmdt_2 ; not a valid file !
43843 0000EF9C 668B4614                <1>       mov   ax, [esi+DirEntry_FstClusHI]
43844 0000EFA0 C1E010                  <1>       shl   eax, 16
43845 0000EFA3 668B461A                <1>       mov   ax, [esi+DirEntry_FstClusLO]
43846 0000EFA7 39C8                    <1>       cmp   eax, ecx ; same first cluster ?
43847 0000EFA9 7407                    <1>       je    short uflmdt_3 ; yes, it is OK !!!
43848                                  <1>
43849                                  <1> uflmdt_2:
43850                                  <1>       ; save directory buffer if has modified/changed sign
43851                                  <1>       ; (It is good to save dir buff even if the searched
43852                                  <1>       ; directory entry is not found !?)
43853 0000EFAB E8E5B6FFFF              <1>       call  save_directory_buffer
43854 0000EFB0 F9                      <1>       stc   ; update failed
43855 0000EFB1 C3                      <1>       retn
43856                                  <1>
43857                                  <1> uflmdt_3:
43858                                  <1>       ; Update directory entry
43859                                  <1>       ; 26/10/2016
43860 0000EFB2 D0E3                    <1>       shl   bl, 1 ; *2
43861 0000EFB4 8B83[90630100]          <1>       mov   eax, [ebx+OF_SIZE] ; file size
43862 0000EFBA 89461C                  <1>       mov   [esi+DirEntry_FileSize], eax
43863                                  <1>       ;
43864 0000EFBD E835B6FFFF              <1>       call  convert_current_date_time
43865                                  <1>       ; OUTPUT -> DX = Date in dos dir entry format
43866                                  <1>         ;       AX = Time in dos dir entry format
43867 0000EFC2 66894616                <1>       mov   [esi+DirEntry_WrtTime], ax
43868 0000EFC6 66895618                <1>       mov   [esi+DirEntry_WrtDate], dx
43869 0000EFCA 66895612                <1>       mov   [esi+DirEntry_LastAccDate], dx
43870 0000EFCE C605[105B0100]02        <1>       mov   byte [DirBuff_ValidData], 2
```

```
43871 0000EFD5 E8BBB6FFFF          <1>       call   save_directory_buffer
43872 0000EFDA C3                  <1>       retn
43873                              <1>
43874                              <1> uflmdt_4:
43875                              <1>       ; Directory buffer sector read&write
43876                              <1>       ; 23/10/2016
43877                              <1>       ;
43878 0000EFDB 8A4603              <1>       mov    al, [esi+LD_FATType]
43879                              <1> uflmdt_5:
43880 0000EFDE BB[9C090300]        <1>       mov    ebx, rw_buffer ; Common r/w sector buffer addr
43881                              <1>
43882 0000EFE3 20C0                <1>       and    al, al ; 0 = Singlix FS
43883 0000EFE5 0F8492000000        <1>       jz     uflmdt_11
43884                              <1>
43885 0000EFEB 21D2                <1>       and    edx, edx
43886 0000EFED 7521                <1>       jnz    short uflmdt_9
43887                              <1>
43888 0000EFEF 3C02                <1>       cmp    al, 2  ; 3 = FAT32
43889 0000EFF1 771A                <1>       ja     short uflmdt_8
43890                              <1>
43891 0000EFF3 89F8                <1>       mov    eax, edi ; directory entry index number
43892 0000EFF5 66C1E804            <1>       shr    ax, 4 ; 16 entries per sector
43893 0000EFF9 034664              <1>       add    eax, [esi+LD_ROOTBegin]
43894                              <1>       ; eax = root directory sector
43895                              <1> uflmdt_6:
43896 0000EFFC 50                  <1>       push   eax ; * ; disk sector address
43897 0000EFFD 51                  <1>       push   ecx ; first cluster
43898 0000EFFE B901000000          <1>       mov    ecx, 1
43899                              <1>       ; ecx = sector count
43900 0000F003 E8D0010000          <1>       call   disk_read
43901 0000F008 59                  <1>       pop    ecx
43902 0000F009 731A                <1>       jnc    short uflmdt_10
43903 0000F00B 58                  <1>       pop    eax ; *
43904                              <1> uflmdt_7:
43905 0000F00C C3                  <1>       retn
43906                              <1>
43907                              <1> uflmdt_8:
43908 0000F00D 8B5632              <1>       mov    edx, [esi+LD_BPB+FAT32_RootFClust]
43909                              <1> uflmdt_9:
43910 0000F010 83FA02              <1>       cmp    edx, 2
43911 0000F013 72F7                <1>       jb     short uflmdt_7 ; invalid, nothing to do
43912                              <1>
43913 0000F015 83EA02              <1>       sub    edx, 2
43914 0000F018 89D0                <1>       mov    eax, edx
43915 0000F01A 0FB65613            <1>       movzx  edx, byte [esi+LD_BPB+SecPerClust]
43916 0000F01E F7E2                <1>       mul    edx
43917 0000F020 034668              <1>       add    eax, [esi+LD_DATABegin]
43918                              <1>       ; eax = sub directory (data) sector
43919 0000F023 EBD7                <1>       jmp    short uflmdt_6
43920                              <1>
43921                              <1> uflmdt_10:
43922                              <1>       ; Directory sector buffer is ready here!
43923                              <1>       ; OF_FCLUSTER must be compared/verified
43924                              <1>       ; edi = dir entry index number (<= 2047)
43925 0000F025 6683E70F            <1>       and    di, 0Fh ; 16 entries per sector
43926 0000F029 66C1E705            <1>       shl    di, 5 ; dir entry index * 32
43927 0000F02D 81C7[9C090300]      <1>       add    edi, rw_buffer
43928                              <1>       ;
43929 0000F033 F6470B18            <1>       test   byte [edi+DirEntry_Attr], 18h ; Vol & Dir
43930 0000F037 0F856EFFFFFF        <1>       jnz    uflmdt_2 ; not a valid file !
43931 0000F03D 668B5714            <1>       mov    dx, [edi+DirEntry_FstClusHI]
43932 0000F041 C1E210              <1>       shl    edx, 16
43933 0000F044 668B571A            <1>       mov    dx, [edi+DirEntry_FstClusLO]
43934 0000F048 39CA                <1>       cmp    edx, ecx ; same first cluster ?
43935 0000F04A 0F855BFFFFFF        <1>       jne    uflmdt_2 ; no !?
43936                              <1>
43937                              <1>       ; Update directory entry
43938 0000F050 E8A2B5FFFF          <1>       call   convert_current_date_time
43939                              <1>       ; OUTPUT -> DX = Date in dos dir entry format
43940                              <1>         ;       AX = Time in dos dir entry format
43941 0000F055 66894716            <1>       mov    [edi+DirEntry_WrtTime], ax
43942 0000F059 66895718            <1>       mov    [edi+DirEntry_WrtDate], dx
43943 0000F05D 66895712            <1>       mov    [edi+DirEntry_LastAccDate], dx
43944                              <1>
43945 0000F061 58                  <1>       pop    eax ; *
43946                              <1>
43947 0000F062 BB[9C090300]        <1>       mov    ebx, rw_buffer ; Common r/w sector buffer addr
43948 0000F067 B901000000          <1>       mov    ecx, 1
43949                              <1>       ; esi = logical dos description table address
43950                              <1>       ; eax = disk sector number/address (LBA)
43951                              <1>       ; ecx = sector count
43952                              <1>       ; ebx = buffer address
43953 0000F06C E858010000          <1>       call   disk_write
43954 0000F071 0F8234FFFFFF        <1>       jc     uflmdt_2
43955                              <1>
43956                              <1>       ; save directory buffer if has modified/changed sign
43957 0000F077 E819B6FFFF          <1>       call   save_directory_buffer
43958 0000F07C C3                  <1>       retn
43959                              <1>
43960                              <1> uflmdt_11:
43961                              <1>       ; 24/10/2016
43962                              <1>       ; Update last modification date & time of a file
43963                              <1>       ; on a disk with Singlix File System.
43964                              <1>       ;
43965                              <1>       ; (Method: Read the FDT -File Description Table-
43966                              <1>       ; sector of the file and update the lmdt data fields,
43967                              <1>       ; then write FDT sector to the disk.
43968                              <1>       ; /// It is easy but there is compatibility buffer
43969                              <1>       ; method also for changing directory entry data and
43970                              <1>       ; also there are some programming issues for Singlix
43971                              <1>       ; file system (TRFS), which are not completed yet!)
43972                              <1>       ;
43973                              <1>       ; Not ready yet ! (24/10/2016)
```

```
                                <1>       ; /// Temporary code for error return ! ///
0000F07D 31C0                   <1>       xor    eax, eax
0000F07F F9                     <1>       stc
0000F080 C3                     <1>       retn
                                <1>
                                <1> sysalloc:
                                <1>       ; 14/10/2017
                                <1>       ; 20/08/2017, 01/09/2017
                                <1>       ; 20/02/2017, 04/03/2017, 15/05/2017
                                <1>       ; 19/02/2017 - TRDOS 386 (TRDOS v2.0)
                                <1>       ; (TRDOS 386 feature only!)
                                <1>       ;
                                <1>       ; Allocate Contiguous Memory Block/Pages (for user)
                                <1>       ; (System call for DMA Buffer allocation etc.)
                                <1>       ;
                                <1>       ; INPUT ->
                                <1>       ;     EBX = Virtual address (for user)
                                <1>       ;          (Physical memory block/aperture
                                <1>       ;          will be mapped to this virtual address)
                                <1>       ;     ECX = Byte Count
                                <1>       ;          (will be rounded up to page border)
                                <1>       ;     If ECX = 0
                                <1>       ;         System call will return with an error (cf=1)
                                <1>       ;         but ECX will contain maximum size of
                                <1>       ;         available memory aperture and physical
                                <1>       ;         (beginning) address of that aperture
                                <1>       ;         (which have maximum size) will be in EAX.
                                <1>       ;     EDX = Upper limit of the requested physical memory
                                <1>       ;           block/pages.
                                <1>       ;          (The last byte address of the memory aperture
                                <1>       ;           must not be equal to or above this limit.)
                                <1>       ;     If EDX = 0
                                <1>       ;        there is NOLIMIT !
                                <1>       ;     If EDX = 0FFFFFFFFh (-1)
                                <1>       ;        ESI = Lower Limit !
                                <1>       ;           (Beginning of the block must not be 'less'
                                <1>       ;           than this.) (Must be equal to or above...)
                                <1>       ;        EDI = Upper Limit !
                                <1>       ;           (End of the block must be !less! than this)
                                <1>       ;           (The last byte addr of the memory aperture
                                <1>       ;           must not be equal to or above this limit.)
                                <1>       ;
                                <1>       ; OUTPUT ->
                                <1>       ;     If CF = 0
                                <1>       ;     EAX = Physical address of the allocated memory block
                                <1>       ;     ECX = Allocated bytes (as rounded up to page borders)
                                <1>       ;     EBX = Virtual address (as rounded up)
                                <1>       ;     IF CF = 1
                                <1>       ;         Requested (size of) Memory block could not be
                                <1>       ;         allocated to the user!
                                <1>       ;     IF CF = 1 & EAX = 0 (Insufficient memory error!)
                                <1>       ;        ECX = Total number of free bytes
                                <1>       ;              (not size of available contiguous bytes!)
                                <1>       ;     If CF = 1 & EAX > 0
                                <1>       ;        there is not a memory aperture with requested size
                                <1>       ;        but total free mem is not less than requested size.
                                <1>       ;        EAX = Physical addr of available memory aperture
                                <1>       ;            with max size
                                <1>       ;             (but it doesn't fit to the conditions!)
                                <1>       ;        ECX = Size of available memory aperture in bytes.
                                <1>       ;     If CF = 1 -> EAX = 0FFFFFFFFh
                                <1>       ;        Conditions/Parameters are wrong !
                                <1>       ;        ECX is same with input value.
                                <1>       ;
                                <1>       ; Note:      Previously allocated pages will be deallocated if
                                <1>       ;      new allocation conditions are met.
                                <1>       ;
                                <1>       ; Note: u.break control may be included in future versions
                                <1>       ;
                                <1>
0000F081 31C0                   <1>       xor    eax, eax ; 0
                                <1>       ; 14/10/2017
0000F083 4A                     <1>       dec    edx ; is there a limit ?
0000F084 7810                   <1>       js     short sysalloc_1 ;  0 -> 0FFFFFFFFh -> NO LIMIT
0000F086 42                     <1>       inc    edx ; > 0
                                <1>       ; Check upper address limit
                                <1>       ;(round up to page borders)
0000F087 81C1FF0F0000           <1>       add    ecx, PAGE_SIZE-1 ; 4095
0000F08D 6681E100F0             <1>       and    cx, ~PAGE_OFF ; not 4095
0000F092 39CA                   <1>       cmp    edx, ecx ; upper limit - block size
0000F094 7224                   <1>       jb     short sysalloc_err
                                <1> sysalloc_1:
                                <1>       ; EAX = Beginning address (physical)
                                <1>       ; EAX = 0 -> Allocate mem block from the 1st proper aperture
                                <1>       ; ECX = Number of bytes to be allocated
0000F096 E88963FFFF             <1>       call   allocate_memory_block
0000F09B 721D                   <1>       jc     short sysalloc_err
                                <1>       ; 01/09/2017
0000F09D 29C2                   <1>       sub    edx, eax ; upper limit address - beginning address
0000F09F 760F                   <1>       jna    short sysalloc_3 ; begin addr not less than the limit
0000F0A1 39CA                   <1>       cmp    edx, ecx
0000F0A3 720B                   <1>       jb     short sysalloc_3 ; end address overs the limit
                                <1> sysalloc_2:
                                <1>       ; EAX = Beginning (physical) addr of the allocated mem block
                                <1>       ; ECX = Num of allocated bytes (rounded up to page borders)
0000F0A5 50                     <1>       push   eax ; * ; 04/03/2017
                                <1>       ; Here, requested contiquous memory pages have been allocated
                                <1>       ; on Memory Allocation Table but user's page directory
                                <1>       ; and page tables have not been updated yet!
0000F0A6 51                     <1>       push   ecx ; **
                                <1>       ; ebx = virtual address (will be rounded up to page border)
                                <1>       ; ecx = number of bytes to be deallocated
                                <1>       ;     will be adjusted to ebx+ecx round down - ebx round up
```

```
44077 0000F0A7 E8D366FFFF          <1>       call   deallocate_user_pages
44078 0000F0AC 731F                <1>       jnc    short sysalloc_4 ; EAX = Deallocated memory bytes
44079 0000F0AE 59                  <1>       pop    ecx ; **
44080 0000F0AF 58                  <1>       pop    eax ; *
44081                              <1> sysalloc_3:
44082                              <1>       ; error !
44083                              <1>       ; restore Memory Allocation Table Content
44084 0000F0B0 E87C65FFFF          <1>       call   deallocate_memory_block
44085 0000F0B5 31C0                <1>       xor    eax, eax ; 0
44086 0000F0B7 48                  <1>       dec    eax ; 0FFFFFFFFh ; 15/05/2017
44087 0000F0B8 EB09                <1>       jmp    short sysalloc_wrong
44088                              <1> sysalloc_err:
44089 0000F0BA 8B2D[60030300]      <1>       mov    ebp, [u.usp]  ; ebp points to user's registers
44090 0000F0C0 894D18              <1>       mov    [ebp+24], ecx ; return to user with ecx value
44091                              <1> sysalloc_wrong:
44092                              <1>       ; eax = 0FFFFFFFFh
44093 0000F0C3 A3[64030300]        <1>       mov    [u.r0], eax
44094 0000F0C8 E9F0D3FFFF          <1>       jmp    error
44095                              <1> sysalloc_4:
44096 0000F0CD 8B2D[60030300]      <1>       mov    ebp, [u.usp]  ; ebp points to user's registers
44097 0000F0D3 894518              <1>       mov    [ebp+24], eax ; return to user with ecx value
44098 0000F0D6 895D10              <1>       mov    [ebp+16], ebx ; new value of ebx (rounded up)
44099 0000F0D9 89C1                <1>       mov    ecx, eax ; byte count (from 'deallocate_user_pages')
44100 0000F0DB 5A                  <1>       pop    edx ; ** ; discard (another) byte count
44101 0000F0DC 58                  <1>       pop    eax ; *
44102 0000F0DD A3[64030300]        <1>       mov    [u.r0], eax ; physical address
44103                              <1>
44104 0000F0E2 51                  <1>       push   ecx ; 20/08/2017
44105                              <1>       ;
44106                              <1>       ; Write newly allocated contiguous (physical) pages
44107                              <1>       ; on page dir and page tables of current user/process
44108                              <1>       ; as PRESENT, USER, WRITABLE
44109                              <1>       ; (then clear allocated pages)
44110 0000F0E3 E88C67FFFF          <1>       call   allocate_user_pages
44111                              <1>       ;jnc   sysret ; OK! return to process with success...
44112                              <1>
44113                              <1>       ; 20/08/2017 ('sysdma' modification)
44114 0000F0E8 59                  <1>       pop    ecx
44115 0000F0E9 A1[64030300]        <1>       mov    eax, [u.r0]   ; physical address (of the block)
44116                              <1>
44117 0000F0EE 721D                <1>       jc     short sysalloc_6
44118                              <1>
44119 0000F0F0 833D[E8690100]FF    <1>       cmp    dword [dma_addr], 0FFFFFFFFh ; -1
44120 0000F0F7 0F82E0D3FFFF        <1>       jb     sysret
44121                              <1>
44122 0000F0FD A3[E8690100]        <1>       mov    [dma_addr], eax ; save dma address for sysdma
44123 0000F102 890D[EC690100]      <1>       mov    [dma_size], ecx ; save dma buff size for sysdma
44124                              <1>
44125 0000F108 E9D0D3FFFF          <1>       jmp    sysret
44126                              <1>
44127                              <1> sysalloc_6:
44128                              <1>       ;
44129                              <1>       ; unexpected error ! insufficient memory !? conflict !?
44130                              <1>       ; (!!?there is not a free page for a new page table?!!)
44131                              <1>       ; We need to terminate process with error message !!!
44132                              <1>       ;
44133 0000F10D 8B2D[60030300]      <1>       mov    ebp, [u.usp]  ; ebp points to user's registers
44134 0000F113 8B4D18              <1>       mov    ecx, [ebp+24] ; byte count
44135                              <1>
44136                              <1>       ; 20/08/2017
44137                              <1>       ;mov   eax, [u.r0]   ; physical address (of the block)
44138                              <1>
44139                              <1>       ;
44140                              <1>       ; restore Memory Allocation Table Content
44141 0000F116 E81665FFFF          <1>       call   deallocate_memory_block
44142                              <1>       ;
44143 0000F11B 803D[C25E0000]03    <1>       cmp    byte [CRT_MODE], 3 ; 80x25 text mode?
44144 0000F122 7407                <1>       je     short sysalloc_7 ; yes
44145                              <1>       ; Current mode is VGA (or CGA graphics) mode,
44146                              <1>       ; We need to return to text mode for displaying
44147                              <1>       ; error message just before 'sysexit'.
44148 0000F124 B003                <1>       mov    al, 3
44149 0000F126 E83A24FFFF          <1>       call   _set_mode
44150                              <1> sysalloc_7:
44151 0000F12B BE[340A0100]        <1>       mov    esi, beep_Insufficient_Memory ; error message
44152 0000F130 E82872FFFF          <1>       call   print_msg ; print/display the message
44153 0000F135 B801000000          <1>       mov    eax, 1 ; ax=1 is needed for 'sysexit' procedure
44154 0000F13A E925D5FFFF          <1>       jmp    sysexit ; and terminate the process !
44155                              <1>
44156                              <1> sysdalloc:
44157                              <1>       ; 19/02/2017 - TRDOS 386 (TRDOS v2.0)
44158                              <1>       ; (TRDOS 386 feature only!)
44159                              <1>       ;
44160                              <1>       ; Deallocate Memory Block/Pages (for user)
44161                              <1>       ; (Complementary call for sysalloc.)
44162                              <1>       ;
44163                              <1>       ; INPUT ->
44164                              <1>       ;     EBX = Virtual address (for user)
44165                              <1>       ;           (will be rounded up to page border)
44166                              <1>       ;     ECX = Byte Count
44167                              <1>       ;          (will be adjusted to page borders)
44168                              <1>       ;     If ICX = 0
44169                              <1>       ;        nothing to do
44170                              <1>       ;     If EBX + ECX > User's ESP
44171                              <1>       ;        nothing to do
44172                              <1>       ;
44173                              <1>       ; Note: u.break control may be included in future versions
44174                              <1>       ;
44175                              <1>       ; OUTPUT ->
44176                              <1>       ;     If CF = 0
44177                              <1>       ;        EAX = Deallocated memory bytes
44178                              <1>       ;        EBX = Virtual address (as rounded up)
44179                              <1>       ;     IF CF = 1
```

```
44180                                  <1>       ;        EAX = 0
44181                                  <1>       ;
44182                                  <1>       ; Note:      Main purpose of this call is to deallocate/release
44183                                  <1>       ;     previously allocated (physically) contiguous memory
44184                                  <1>       ;     pages but beginning (virtual) address may not be
44185                                  <1>       ;     followed by physically contiguous pages. So, this
44186                                  <1>       ;     system call will deallocate user's virtually
44187                                  <1>       ;     contiguous memory pages. Also, there is not any
44188                                  <1>       ;     objections to use this system call without sysalloc
44189                                  <1>       ;     system call; only possible objection is to lost data
44190                                  <1>       ;     within user's memory space, if the beginning address
44191                                  <1>       ;     and size is not proper.
44192                                  <1>       ;
44193                                  <1>       ; Note: Empty page tables will not be deallocated!!!
44194                                  <1>       ;       (they will be deallocated at process termination)
44195                                  <1>       ;
44196                                  <1>       ; Note: When the program terminates itself or when it is
44197                                  <1>       ;     terminated by operating system kernel, all allocated
44198                                  <1>       ;     memory pages will be deallocated during termination
44199                                  <1>       ;     stage. So, 'sysdalloc' is not necessary except
44200                                  <1>       ;     forgiving memory block to other programs/processes.
44201                                  <1>       ;
44202 0000F13F 8B15[5C030300]         <1>       mov    edx, [u.sp]
44203 0000F145 8B420C                <1>       mov    eax, [edx+12] ; user's stack pointer
44204 0000F148 29C8                  <1>       sub    eax, ecx ; esp - byte count
44205 0000F14A 24FC                  <1>       and    al, 0FCh ; dword alignment
44206 0000F14C 39D8                  <1>       cmp    eax, ebx
44207 0000F14E 7220                  <1>       jb     short sysdalloc_err ; deallocation overlaps with stack
44208                                  <1>
44209 0000F150 31C0                  <1>       xor    eax, eax
44210 0000F152 21C9                  <1>       and    ecx, ecx
44211 0000F154 7407                  <1>       jz     short sysdalloc_2
44212                                  <1>
44213 0000F156 E82466FFFF            <1>       call   deallocate_user_pages
44214 0000F15B 7213                  <1>       jc     short sysdalloc_err
44215                                  <1>
44216                                  <1> sysdalloc_2:
44217 0000F15D A3[64030300]          <1>       mov    [u.r0], eax
44218 0000F162 8B2D[60030300]        <1>       mov    ebp, [u.usp]
44219 0000F168 895D10                <1>       mov    [ebp+16], ebx ; new value of ebx
44220 0000F16B E96DD3FFFF            <1>       jmp    sysret
44221                                  <1>
44222                                  <1> sysdalloc_err:
44223 0000F170 A3[64030300]          <1>       mov    [u.r0], eax ; 0
44224 0000F175 E943D3FFFF            <1>       jmp    error
44225                                  <1>
44226                                  <1> syscalbac:
44227                                  <1>       ; SYS CALLBACK
44228                                  <1>       ; 16/04/2017
44229                                  <1>       ; 14/04/2017
44230                                  <1>       ; 13/04/2017
44231                                  <1>       ; 28/02/2017
44232                                  <1>       ; 26/02/2017
44233                                  <1>       ; 24/02/2017
44234                                  <1>       ; 21/02/2017 - TRDOS 386 (TRDOS v2.0)
44235                                  <1>       ; (TRDOS 386 feature only!)
44236                                  <1>       ;
44237                                  <1>       ; Link or unlink IRQ callback service to/from user (ring 3)
44238                                  <1>       ;
44239                                  <1>       ; INPUT ->
44240                                  <1>       ;     BL = IRQ number (Hardware interrupt request number)
44241                                  <1>       ;          (0 t0 15 but IRQ 0,1,2,6,8,14,15 are prohibited)
44242                                  <1>       ;          IRQ numbers 3,4,5,7,9,10,11,12,13 are valid
44243                                  <1>       ;          (numbers >15 are invalid)
44244                                  <1>       ;
44245                                  <1>       ;     BH = 0 = Unlink IRQ (in BL) from user (ring 3) service
44246                                  <1>       ;          1 = Link IRQ by using Signal Response Byte method
44247                                  <1>       ;          2 = Link IRQ by using Callback service method
44248                                  <1>       ;          3 = Link IRQ by using Auto Increment S.R.B. method
44249                                  <1>       ;         >3 = invalid
44250                                  <1>       ;
44251                                  <1>       ;     CL = Signal Return/Response Byte value
44252                                  <1>       ;
44253                                  <1>       ;     If BH = 2, kernel will put a counter value
44254                                  <1>       ;               (into the S.R.B. addr)
44255                                  <1>       ;              between 0 to 255. (start value = CL+1)
44256                                  <1>       ;
44257                                  <1>       ;     NOTE: counter value, for example: even and odd numbers
44258                                  <1>       ;           may be used for -audio- DMA buffer switch
44259                                  <1>       ;           within double buffer method, etc.
44260                                  <1>       ;
44261                                  <1>       ;     EDX = Signal return (Response) byte address
44262                                  <1>       ;                           - or -
44263                                  <1>       ;           Interrupt/Callback service/routine address
44264                                  <1>       ;
44265                                  <1>       ;           (virtual address in user's memory space)
44266                                  <1>       ;
44267                                  <1>       ; OUTPUT ->
44268                                  <1>       ;     CF = 0 & EAX = 0 -> Successful setting
44269                                  <1>       ;     CF = 1 & EAX > 0 -> IRQ is prohibited or locked
44270                                  <1>       ;                         by another process
44271                                  <1>       ;              eax = ERR_PERM_DENIED -> prohibited or locked
44272                                  <1>       ;              eax = ERR_INV_PARAMETER ->
44273                                  <1>       ;                    invalid parameter/option or bad address
44274                                  <1>       ;
44275                                  <1>       ;     NOTE: Timer callbacks are set by using 'systimer'
44276                                  <1>       ;           system call (IRQ 0, PIT and IRQ 8, RTC)
44277                                  <1>       ;
44278                                  <1>       ;           Direct keyboard access is performed by using
44279                                  <1>       ;           Keyboard Interrupt (INT 32h)
44280                                  <1>       ;
44281                                  <1>       ;           It is prohibited here because:
44282                                  <1>       ;            1) Signal Response Byte method has not advantage
```

```
                                        ;               against INT 32h, function AH = 1. Also,
                                        ;               keyboard service interrupt will return with
                                        ;               ascii and scan codes (AL, AH) while
                                        ;               SRB method has only 1 byte space for ascii code
                                        ;               or scan code. One byte signal response is used
                                        ;               for ensuring very simple and very fast
                                        ;               virtual to physical memory address conversion
                                        ;               without any memory page crossover risk.
                                        ;               (Otherwise double page conversion or word
                                        ;               alignment would be needed.)
                                        ;            2) Badly written user code (callback code)
                                        ;               can prevent keyboard and timesharing functions
                                        ;               of the operating system via continuous and long
                                        ;               keyboard event handling by callback service.
                                        ;               (It can cause to lose immediate keystroke
                                        ;               response from hardware to user.)
                                        ;            3) If user will check any keyboard events, 'getkey'
                                        ;               (or 'getchar') must have more priority than other
                                        ;               (video etc.) events because only control ability
                                        ;               on a procedural infinite loop is a keyboard or
                                        ;               mouse event. So user can use keyboard function
                                        ;               at the end or at the beginning of a loop.
                                        ;               In this case, INT 32h is used for that purpose
                                        ;               and timer interrupt etc. callbacks can be used
                                        ;               for dynamic and synchronized data refresh/transfer
                                        ;               while cpu is in a static loop (without polling).
                                        ;               Keyboard Int callback is not more useful because
                                        ;               already a manual check (a key is pressed or not)
                                        ;               can be performed (via INT 32h, AH = 1) efficiently
                                        ;               in a loop to prevent a locked infinitive loop.
                                        ;
                                        ;         Disk IRQs (6,14,15) have been phohibited from ring 3
                                        ;          callback because, disk operations (file system services
                                        ;          etc.) are independent from user program, for fast disk r/w.
                                        ;          They are not more useful at ring 3 while they are in use
                                        ;          by standard diskio functions which are mandatory part of
                                        ;          (monolithic) OS kernel and mainprog command interpreter.
                                        ;          INT 33h diskio functions are enough for user level disk
                                        ;          r/w.
                                        ;
                                        ; TRDOS 386 - IRQ CALLBACK structures (parameters):
                                        ;
                                        ;         [u.irqlock] = 1 word, IRQ flags (0-15) that indicates
                                        ;                   which IRQs are locked by (that) user.
                                        ;                     Lock and unlock (by user) will change
                                        ;                   these flags or 'terminate process' (sysexit)
                                        ;                   will clear these flags and unlock those IRQs.
                                        ;
                                        ;                   Bit 0 is for IRQ 0 and Bit 15 is for IRQ 15
                                        ;
                                        ;         IRQ(x).owner      : 1 byte, user, [u.uno], 0 = free (unlocked)
                                        ;
                                        ;         IRQ(x).method : 1 byte for callback method & status
                                        ;                       0 = Signal Response Byte method
                                        ;                       1 = Callback service method
                                        ;                       >1 = invalid for current 'syscalback'.
                                        ;                   or(+) 80h = IRQ is in use by system (ring 0)
                                        ;                               function (audio etc.) or
                                        ;                             a device driver.
                                        ;                   (system function will ignore the lock/owner)
                                        ;
                                        ;         IRQ(x).srb: 1 byte, Signal Return/Response byte value
                                        ;                    (a fixed value by user or a counter value
                                        ;                    from 0 to 255, which is increased by every
                                        ;                    interrupt just before putting it into
                                        ;                    the Signal Response byte address
                                        ;                    (This is not used in callback serv method)
                                        ;
                                        ;         IRQ(x).addr      : 1 dword
                                        ;                    Signal Response Byte address (physical)
                                        ;                              -or-
                                        ;                    Callback service address (virtual)
                                        ;
                                        ;            IRQ(x).dev: 1 byte
                                        ;                    0 = Default device or kernel function
                                        ;                              -or-
                                        ;                    1-255 = Assigned device driver number
                                        ;
                                        ;            (x) = 3,4,5,7,9,10,11,12,13
                                        ;
                                        ;
                                        ;         NOTE: If user's process/program calls the kernel (INT 40h)
                                        ;               while it is already running in a (ring 3) callback
                                        ;               service, kernel will force (convert) system call to
                                        ;               'sysrele' (sys release). So, this feature provides
                                        ;               easy and simple usage of callback services without
                                        ;               falling into deepless <please 'callback me' then
                                        ;               let me 'callback you'> cycles! (User must return
                                        ;               from callback service by using 'sysrele' system
                                        ;               call, without a significant delay. Otherwise user
                                        ;               process/program may be late to catch the next event
                                        ;               within same callback purpose.
                                        ;
0000F17A 30C0                           xor   al, al ; the caller is 'syscalbac' sign/flag
0000F17C E827110000                     call  set_irq_callback_service
                                        ; 16/04/2017
0000F181 A3[64030300]                   mov   [u.r0], eax
0000F186 0F8351D3FFFF                   jnc   sysret
0000F18C A3[C8030300]                   mov   dword [u.error], eax
0000F191 E927D3FFFF                     jmp   error
                                        
                                        sysfpstat:
```

```
                                <1>       ; 28/02/2017 - TRDOS 386 (TRDOS v2.0)
                                <1>       ; (TRDOS 386 feature only!)
                                <1>       ;
                                <1>       ; Set or reset FPU registers save/restore option (for user)
                                <1>       ;            (during software task switching, wswap-rswap)
                                <1>       ;
                                <1>       ; INPUT ->
                                <1>       ;     BL = 0 -> reset
                                <1>       ;     BL = 1 -> set (FPU register will be saved and restored)
                                <1>       ;
                                <1>       ; OUTPUT ->
                                <1>       ;     cf = 0 -> no error, FPU is ready...
                                <1>       ;               (EAX = 0)
                                <1>       ;     Cf = 1 -> error, 80387 FPU is not ready !
                                <1>       ;               (EAX = 0FFFFFFFFh)
                                <1>
0000F196 31C0                   <1>       xor    eax, eax
0000F198 803D[E05F0100]00       <1>       cmp    byte [fpready], 0
0000F19F 7613                   <1>       jna    short sysfpstat_err
                                <1>
0000F1A1 80E301                 <1>       and    bl, 1 ; use BIT 0 only !
0000F1A4 881D[DA030300]         <1>       mov    [u.fpsave], bl
0000F1AA A3[64030300]           <1>       mov    [u.r0], eax ; 0
0000F1AF E929D3FFFF             <1>       jmp    sysret
                                <1>
                                <1> sysfpstat_err:
0000F1B4 48                     <1>       dec    eax ; 0FFFFFFFFh
0000F1B5 A3[64030300]           <1>       mov    [u.r0], eax ; -1
0000F1BA E9FED2FFFF             <1>       jmp    error
                                <1>
                                <1> ;maknod:
                                <1>       ; 26/10/2016
                                <1>       ; temporary
                                <1> ;     retn
                                <1>
                                <1> ; temporary - 24/01/2016
                                <1>
                                <1> iget:
0000F1BF C3                     <1>       retn
                                <1> isintr:
0000F1C0 C3                     <1>       retn
                                <1> iopen:
0000F1C1 C3                     <1>       retn
                                <1> iclose:
0000F1C2 C3                     <1>       retn
                                <1> setimod:
0000F1C3 C3                     <1>       retn
                                <1> sndc:
0000F1C4 C3                     <1>       retn
                                <1> access:
0000F1C5 C3                     <1>       retn
                                <1> epoch:
0000F1C6 C3                     <1>       retn
                                <1> sleep:
0000F1C7 C3                     <1>       retn
                                <1> set_date_time:
0000F1C8 C3                     <1>       retn
                                    %include 'trdosk7.s' ; 24/01/2016
                                <1> ; ****************************************************************************
                                <1> ; TRDOS386.ASM (TRDOS 386 Kernel - v2.0.0) - DISK READ&WRITE : trdosk7.s
                                <1> ; ----------------------------------------------------------------------------
                                <1> ; Last Update: 25/02/2016
                                <1> ; ----------------------------------------------------------------------------
                                <1> ; Beginning: 24/01/2016
                                <1> ; ----------------------------------------------------------------------------
                                <1> ; Assembler: NASM version 2.11 (trdos386.s)
                                <1> ; ----------------------------------------------------------------------------
                                <1> ; Derived from TRDOS Operating System v1.0 (8086) source code by Erdogan Tan
                                <1> ; DISK_IO.ASM (20/07/2011)
                                <1> ; ****************************************************************************
                                <1> ; DISK_IO.ASM (c) 2009-2011 Erdogan TAN [ 04/07/2009 ] Last Update: 20/07/2011
                                <1>
                                <1> disk_write:
                                <1>       ; 25/02/2016
                                <1>       ; 24/02/2016
                                <1>       ; 23/02/2016
0000F1C9 807E0500               <1>       cmp    byte [esi+LD_LBAYes], 0
0000F1CD 777B                   <1>         ja      short lba_write
                                <1>
                                <1> chs_write:
                                <1>       ; 25/02/2016
                                <1>       ; 23/02/2016
0000F1CF C605[D95B0100]03       <1>       mov    byte [disk_rw_op], 3 ; CHS write
0000F1D6 EB0D                   <1>       jmp    short chs_rw
                                <1>
                                <1> disk_read:
                                <1>       ; 25/02/2016
                                <1>       ; 24/02/2016
                                <1>       ; 23/02/2016
                                <1>       ; 17/02/2016
                                <1>       ; 14/02/2016
                                <1>       ; 31/01/2016 (TRDOS 386 =  TRDOS v2.0)
                                <1>       ; 17/10/2010
                                <1>       ; 18/04/2010
                                <1>       ;
                                <1>       ; INPUT -> EAX = Logical Block Address
                                <1>       ;         ESI = Logical Dos Disk Table Offset (DRV)
                                <1>       ;         ECX = Sector Count
                                <1>       ;         EBX = Destination Buffer
                                <1>       ; OUTPUT ->
                                <1>       ;         cf = 0 or cf = 1
                                <1>       ; (Modified registers: EAX, EBX, ECX, EDX)
                                <1>
```

```
44489 0000F1D8 807E0500            <1>       cmp    byte [esi+LD_LBAYes], 0
44490 0000F1DC 7775                <1>         ja      short lba_read
44491                              <1>
44492                              <1> chs_read:
44493                              <1>       ; 25/02/2016
44494                              <1>       ; 24/02/2016
44495                              <1>       ; 23/02/2016
44496                              <1>       ; 31/01/2016 (TRDOS 386 =  TRDOS v2.0)
44497                              <1>       ; 20/07/2011
44498                              <1>       ; 04/07/2009
44499                              <1>       ;
44500                              <1>       ; INPUT -> EAX = Logical Block Address
44501                              <1>       ;         ECX = Number of sectors to read
44502                              <1>       ;         ESI = Logical Dos Disk Table Offset (DRV)
44503                              <1>       ;         EBX = Destination Buffer
44504                              <1>       ; OUTPUT ->
44505                              <1>       ;         cf = 0 or cf = 1
44506                              <1>       ; (Modified registers: EAX; EBX, ECX, EDX)
44507                              <1>
44508                              <1>       ; 23/02/2016
44509 0000F1DE C605[D95B0100]02    <1>       mov    byte [disk_rw_op], 2 ; CHS read
44510                              <1>
44511                              <1> chs_rw:
44512                              <1>       ;;movzx      edx, word [esi+LD_BPB+SecPerTrack]
44513                              <1>       ;movzx edx, byte [esi+LD_BPB+SecPerTrack] ; <= 63
44514                              <1>       ;mov   [disk_rw_spt], dl
44515                              <1>
44516                              <1> chs_read_next_sector:
44517 0000F1E5 C605[DA5B0100]04    <1>       mov    byte [retry_count], 4
44518                              <1>
44519                              <1> chs_read_retry:
44520                              <1>       ;mov   [sector_count], ecx ; 23/02/2016
44521                              <1>
44522 0000F1EC 50                  <1>       push   eax                ; Linear sector #
44523 0000F1ED 51                  <1>       push   ecx                ; # of FAT/FILE/DIR sectors
44524                              <1>
44525 0000F1EE 0FB74E1E            <1>       movzx  ecx, word [esi+LD_BPB+SecPerTrack]
44526                              <1>       ;movzx ecx, byte [disk_rw_spt] ; 23/02/2016
44527 0000F1F2 29D2                <1>       sub    edx, edx
44528 0000F1F4 F7F1                <1>       div    ecx
44529                              <1>       ; eax = track, dx (dl ) = sector (on track)
44530                              <1>       ;sub   cl, dl ; 24/02/2016 (spt - sec)
44531                              <1>       ;push  ecx ; *
44532 0000F1F6 6689D1              <1>       mov    cx, dx             ; Sector (zero based)
44533 0000F1F9 6641                <1>       inc    cx                 ; To make it 1 based
44534 0000F1FB 6651                <1>       push   cx
44535 0000F1FD 668B4E20            <1>       mov    cx, [esi+LD_BPB+Heads]
44536 0000F201 6629D2              <1>       sub    dx, dx
44537 0000F204 F7F1                <1>       div    ecx                ; Convert track to head & cyl
44538                              <1>       ; eax (ax) = cylinder, dx (dl) = head (max. FFh)
44539 0000F206 88D6                <1>       mov    dh, dl
44540 0000F208 6659                <1>       pop    cx                 ; AX=Cyl, DH=Head, CX=Sector
44541 0000F20A 8A5602              <1>       mov    dl, [esi+LD_PhyDrvNo]
44542                              <1>
44543 0000F20D 88C5                <1>       mov    ch, al             ; NOTE: max. 1023 cylinders !
44544 0000F20F C0CC02              <1>       ror    ah, 2              ; Rotate 2 bits right
44545 0000F212 08E1                <1>       or     cl, ah
44546                              <1>
44547                              <1>       ; 24/02/2016
44548                              <1>       ;pop   eax ; * (spt - sec) (example: 63 - 0 = 63)
44549                              <1>       ;cmp   eax, [sector_count]
44550                              <1>       ;jb    short chs_write_sectors
44551                              <1>       ;je    short chs_read_sectors
44552                              <1>       ;; (# of sectors to read is more than remaining sectors on the track)
44553                              <1>       ;mov   al, [sector_count]
44554                              <1> ;chs_read_sectors: ; read or write !
44555 0000F214 B001                <1>       mov    al, 1 ; 25/02/2016
44556 0000F216 8A25[D95B0100]      <1>       mov    ah, [disk_rw_op]  ; 02h = chs read, 03h = chs write
44557                              <1>       ;
44558 0000F21C E8E54FFFFF          <1>       call   int13h             ; BIOS Service func ( ah ) = 2
44559                              <1>                                 ; Read disk sectors
44560                              <1>                                 ; AL-sec num CH-track CL-sec
44561                              <1>                                 ; DH-head DL-drive ES:BX-buffer
44562                              <1>                                 ; CF-flag AH-stat AL-sec read
44563                              <1>                                ; If CF = 1 then (If AH > 0)
44564 0000F221 8825[DB5B0100]      <1>       mov    [disk_rw_err], ah
44565                              <1>
44566 0000F227 59                  <1>       pop    ecx
44567 0000F228 58                  <1>       pop    eax
44568 0000F229 7314                <1>       jnc    short chs_read_ok
44569                              <1>
44570 0000F22B 803D[DB5B0100]09    <1>       cmp    byte [disk_rw_err], 09h ; DMA crossed 64K segment boundary
44571 0000F232 7408                <1>       je     short chs_read_error_retn
44572                              <1>
44573 0000F234 FE0D[DA5B0100]      <1>       dec    byte [retry_count]
44574 0000F23A 75B0                <1>       jnz    short chs_read_retry
44575                              <1>
44576                              <1> chs_read_error_retn:
44577 0000F23C F9                  <1>       stc
44578                              <1>       ;retn
44579 0000F23D EB69                <1>       jmp    short update_drv_error_byte
44580                              <1>
44581                              <1> ;chs_write_sectors: ; read or write
44582                              <1>       ;; (# of sectors to read is less than remaining sectors on the track)
44583                              <1>       ;mov   [sector_count], al
44584                              <1>       ;jmp   short chs_read_sectors
44585                              <1>
44586                              <1> chs_read_ok:
44587                              <1>       ;; 23/02/2016
44588                              <1>       ;movzx edx, byte [sector_count] ; sector count (<= spt)
44589                              <1>         ;sub    ecx, edx  ; remaining sector count
44590                              <1>       ;jna   short update_drv_error_byte
44591                              <1>       ;add   eax, edx ; next disk sector
```

```
44592                                  <1>       ;shl   edx, 9 ; 512 * sector count
44593                                  <1>       ;add   ebx, edx ; next buffer byte address
44594                                  <1>         ;jmp     chs_read_next_sector
44595                                  <1>       ; 25/02/2016
44596 0000F23F 40                      <1>       inc    eax ; next sector
44597 0000F240 81C300020000            <1>       add    ebx, 512
44598 0000F246 E29D                    <1>       loop   chs_read_next_sector
44599 0000F248 EB5E                    <1>       jmp    short update_drv_error_byte
44600                                  <1>
44601                                  <1> lba_write:
44602                                  <1>       ; 23/02/2016
44603 0000F24A C605[D95B0100]1C        <1>       mov    byte [disk_rw_op], 1Ch ; LBA write
44604 0000F251 EB07                    <1>       jmp    short lba_rw
44605                                  <1>
44606                                  <1> lba_read:
44607                                  <1>       ; 23/02/2016
44608                                  <1>       ; 17/02/2016
44609                                  <1>       ; 14/02/2016
44610                                  <1>       ; 13/02/2016
44611                                  <1>       ; 31/01/2016 (TRDOS 386 =  TRDOS v2.0)
44612                                  <1>       ; 10/07/2015 (Retro UNIX 386 v1)
44613                                  <1>       ;
44614                                  <1>       ; INPUT -> EAX = Logical Block Address
44615                                  <1>       ;        ESI = Logical Dos Disk Table Offset (DRV)
44616                                  <1>       ;        ECX = Sector Count
44617                                  <1>       ;        EBX = Destination Buffer
44618                                  <1>       ; OUTPUT ->
44619                                  <1>       ;        cf = 0 or cf = 1
44620                                  <1>       ; (Modified registers: EAX, EBX, ECX, EDX)
44621                                  <1>
44622                                  <1>       ; LBA read/write (with private LBA function)
44623                                  <1>       ;((Retro UNIX 386 v1 - DISK I/O code by Erdogan Tan))
44624                                  <1>
44625                                  <1>
44626                                  <1>       ; 23/02/2016
44627 0000F253 C605[D95B0100]1B        <1>       mov    byte [disk_rw_op], 1Bh ; LBA read
44628                                  <1>
44629                                  <1> lba_rw:
44630                                  <1>       ; 17/02/2016
44631 0000F25A 57                      <1>       push   edi
44632                                  <1>
44633 0000F25B 890D[DC5B0100]          <1>       mov    [sector_count], ecx ; total sector (read) count
44634                                  <1>
44635 0000F261 8A5602                  <1>       mov    dl, [esi+LD_PhyDrvNo]
44636                                  <1>       ; dl = physical drive number (0,1, 80h, 81h, 82h, 83h)
44637                                  <1>
44638                                  <1> lba_read_next:
44639 0000F264 81F900010000            <1>       cmp    ecx, 256
44640 0000F26A 7605                    <1>       jna    short lba_read_rsc
44641 0000F26C B900010000              <1>       mov    ecx, 256 ; 17/02/2016
44642                                  <1> lba_read_rsc:
44643 0000F271 290D[DC5B0100]          <1>       sub    [sector_count], ecx ; remain sectors
44644                                  <1>
44645 0000F277 89CF                    <1>       mov    edi, ecx
44646 0000F279 89C1                    <1>       mov    ecx, eax ; sector number/address
44647                                  <1>
44648 0000F27B C605[DA5B0100]04        <1>       mov    byte [retry_count], 4
44649                                  <1> lba_read_retry:
44650 0000F282 89F8                    <1>       mov    eax, edi
44651                                  <1>       ;
44652                                  <1>       ; ecx = sector number
44653                                  <1>       ; al = sector count (0 - 255) /// (0 = 256)
44654                                  <1>       ; dl = drive number
44655                                  <1>       ; ebx = buffer offset
44656                                  <1>       ;
44657                                  <1>       ; Function 1Bh = LBA read, 1Ch = LBA write
44658                                  <1>       ; 23/02/2016
44659 0000F284 8A25[D95B0100]          <1>       mov    ah, [disk_rw_op] ; 1Bh = LBA read, 1Ch = LBA write
44660 0000F28A E8774FFFFF              <1>       call   int13h
44661                                  <1>       ; al = ? (changed)
44662                                  <1>       ; ah = error code
44663 0000F28F 8825[DB5B0100]          <1>       mov    [disk_rw_err], ah
44664 0000F295 7334                    <1>       jnc    short lba_read_ok
44665 0000F297 80FC80                  <1>       cmp    ah, 80h ; time out?
44666 0000F29A 740A                    <1>         je     short lba_read_stc_retn
44667 0000F29C FE0D[DA5B0100]          <1>       dec    byte [retry_count]
44668 0000F2A2 7FDE                    <1>       jg     short lba_read_retry
44669 0000F2A4 743A                    <1>       jz     short lba_read_reset
44670                                  <1>       ; sf =  1
44671                                  <1>
44672                                  <1> lba_read_stc_retn:
44673 0000F2A6 F9                      <1>       stc
44674                                  <1> lba_read_retn:
44675 0000F2A7 5F                      <1>       pop    edi
44676                                  <1>
44677                                  <1> update_drv_error_byte:
44678 0000F2A8 9C                      <1>       pushf
44679 0000F2A9 53                      <1>       push   ebx
44680 0000F2AA 6651                    <1>       push   cx
44681                                  <1>       ;or    ecx, ecx
44682                                  <1>       ;jz    short udrv_errb0
44683 0000F2AC 8A0D[DB5B0100]          <1>       mov    cl, [disk_rw_err]
44684                                  <1> udrv_errb0:
44685 0000F2B2 0FB65E02                <1>       movzx  ebx, byte [esi+LD_PhyDrvNo]
44686 0000F2B6 80FB02                  <1>       cmp    bl, 2
44687 0000F2B9 7203                    <1>         jb     short udrv_errb1
44688 0000F2BB 80EB7E                  <1>       sub    bl, 7Eh
44689                                  <1>       ;cmp   bl, 5
44690                                  <1>       ;ja    short udrv_errb2
44691                                  <1> udrv_errb1:
44692 0000F2BE 81C3[495D0000]          <1>         add    ebx, drv.error ; 13/02/2016
44693 0000F2C4 880B                    <1>       mov    [ebx], cl ; error code
44694                                  <1> udrv_errb2:
```

```
44695 0000F2C6 6659              <1>       pop    cx
44696 0000F2C8 5B                <1>       pop    ebx
44697 0000F2C9 9D                <1>       popf
44698 0000F2CA C3                <1>       retn
44699                            <1>
44700                            <1> lba_read_ok:
44701 0000F2CB 89C8              <1>       mov    eax, ecx ; sector number
44702 0000F2CD 01F8              <1>       add    eax, edi ; sector number (next)
44703 0000F2CF C1E709            <1>       shl    edi, 9 ; sector count * 512
44704 0000F2D2 01FB              <1>       add    ebx, edi ; next buffer offset
44705                            <1>
44706 0000F2D4 8B0D[DC5B0100]    <1>       mov    ecx, [sector_count] ; remaining sectors
44707 0000F2DA 09C9              <1>       or     ecx, ecx
44708 0000F2DC 7586              <1>       jnz    short lba_read_next
44709 0000F2DE EBC7              <1>       jmp    short lba_read_retn
44710                            <1>
44711                            <1> lba_read_reset:
44712 0000F2E0 B40D              <1>       mov    ah, 0Dh ; Alternate reset
44713 0000F2E2 E81F4FFFFF        <1>         call int13h
44714                            <1>       ; al = ? (changed)
44715                            <1>       ; ah = error code
44716 0000F2E7 7399              <1>       jnc    short lba_read_retry
44717 0000F2E9 EBBC              <1>       jmp    short lba_read_retn
44718                                %include 'trdosk8.s' ; 24/01/2016
44719                            <1> ; ****************************************************************************
44720                            <1> ; TRDOS386.ASM (TRDOS 386 Kernel - v2.0.0) - MAIN PROGRAM : trdosk8.s
44721                            <1> ; ----------------------------------------------------------------------------
44722                            <1> ; Last Update: 12/10/2017
44723                            <1> ; ----------------------------------------------------------------------------
44724                            <1> ; Beginning: 24/01/2016
44725                            <1> ; ----------------------------------------------------------------------------
44726                            <1> ; Assembler: NASM version 2.11 (trdos386.s)
44727                            <1> ; ----------------------------------------------------------------------------
44728                            <1> ; Derived from 'Retro UNIX 386 Kernel - v0.2.1.0' source code by Erdogan Tan
44729                            <1> ; u0.s (20/11/2015), u4.s (14/10/2015)
44730                            <1> ; ****************************************************************************
44731                            <1> ; Derived from TRDOS Operating System v1.0 (8086) source code by Erdogan Tan
44732                            <1> ; TRDOS2.ASM (09/11/2011)
44733                            <1> ; ----------------------------------------------------------------------------
44734                            <1> ; DIR.ASM (c) 2004-2011 Erdogan TAN  [07/01/2004] Last Update: 09/10/2011
44735                            <1>
44736                            <1> set_run_sequence:
44737                            <1>       ; 23/12/2016
44738                            <1>       ; 10/06/2016
44739                            <1>       ; 22/05/2016
44740                            <1>       ; 20/05/2016
44741                            <1>       ; 19/05/2016 - TRDOS 386 (TRDOS v2.0)
44742                            <1>       ; TRDOS 386 feature only !
44743                            <1>       ;
44744                            <1>       ; INPUT ->
44745                            <1>       ;     AL = process number (next process)
44746                            <1>       ;
44747                            <1>       ;     this process must be added to run sequence
44748                            <1>       ;
44749                            <1>       ;     [u.pri] = priority of present process
44750                            <1>       ;
44751                            <1>       ;     DL = priority (queue)
44752                            <1>       ;          0 = background (low) ; run on background
44753                            <1>       ;          1 = regular (normal) ; run as regular
44754                            <1>       ;          2 = event (high)     ; run for event
44755                            <1>       ;
44756                            <1>       ;     1) If the requested process is already running:
44757                            <1>       ;        a) If present priority is high ([u.pri]=2)
44758                            <1>       ;           and requested priority is also high,
44759                            <1>       ;           there is nothing to do! Because it has been
44760                            <1>       ;           done already (before this attempt).
44761                            <1>       ;        b) If present priority is high ([u.pri]=2)
44762                            <1>       ;           and requested priority is not high, there is
44763                            <1>       ;           nothing to do! Because, it's current
44764                            <1>       ;           run queue is unspecified, here. (It may be in
44765                            <1>       ;           a waiting list or in a run queue; if the new
44766                            <1>       ;           priority would be used to add it to relavant
44767                            <1>       ;              run queue, this would be wrong, unnecessary
44768                            <1>       ;           and destabilizing duplication!)
44769                            <1>       ;        c) If present priority is not high ([u.pri]<2)
44770                            <1>       ;              and requested priority is high (event),
44771                            <1>       ;           process will be added to present priority's
44772                            <1>       ;           run queue and then, priority will be changed
44773                            <1>       ;           to high ([u.pri]=2).
44774                            <1>       ;        d) If present priority is not high ([u.pri]<2)
44775                            <1>       ;           and requested priority is not high, [u.pri]
44776                            <1>       ;           value will be changed. There is nothing to do
44777                            <1>       ;           in addition. (The new priority value will be
44778                            <1>       ;           used by 'tswap/tswitch' procedure at 'sysret'
44779                            <1>       ;           or 'sysrele' stage.)
44780                            <1>       ;
44781                            <1>       ;     2) If the requested process is not running:
44782                            <1>       ;        a) If requested priority of the requested
44783                            <1>       ;           (next) process is high (event) and priority
44784                            <1>       ;           of present process is not high, the requested
44785                            <1>       ;           process will be added to ('runq_event') high
44786                            <1>       ;           priority run queue and then present (running)
44787                            <1>       ;           process will be stopped (swapped/switched out)
44788                            <1>       ;           immediately if it is in user mode, or it's
44789                            <1>       ;           [u.quant] value will be reset to 0 and (then)
44790                            <1>       ;           it will be stopped at 'sysret' stage.
44791                            <1>       ;        b) If requested priority of the requested
44792                            <1>       ;           (next) process is high (event) and priority
44793                            <1>       ;           of present process is also high, the requested
44794                            <1>       ;           process will be added to ('runq_event') high
44795                            <1>       ;           priority run queue and present (running)
44796                            <1>       ;           process will be allowed to run until it's
44797                            <1>       ;           time quantum will be elapsed ([u.quant]=0).
```

```
                                        ;        c) If requested priority of the requested
                                        ;           (next) process is not high ('run for event'),
                                        ;           there is nothing to do. Because, it's current
                                        ;           run queue is unspecified, here. (It may be in
                                        ;           a waiting list or in a run queue; if the new
                                        ;           priority would be used to add it to relavant
                                        ;               run queue, this would be wrong, unnecessary
                                        ;           and destabilizing duplication!)
                                        ;
                                        ; OUTPUT ->
                                        ;     none
                                        ;
                                        ;     [u.pri] = priority of present process
                                        ;
                                        ;     cf = 1, if the request could not be fulfilled.
                                        ;
                                        ;     NOTE:
                                        ;          * Processes in 'run as regular' queue can run
                                        ;            if there is no process in 'run for event' queue
                                        ;            ('run for event' processes have higher priority)
                                        ;        * When [u.quant] time quantum of a process is
                                        ;            elapsed, it's high priority ('run for event')
                                        ;            status will be disabled, it can be run in sequence
                                        ;            of it's actual run queue.
                                        ;        * A 'run on background' process will always be
                                        ;            sequenced in 'run on background' (low priority)
                                        ;            queue, it can run only when other priority queues
                                        ;            are empty. (idle time processes, e.g. printing)
                                        ;
                                        ; Modified registers: eax, ebx, edx
                                        ;
                                        srunseq_0:
0000F2EB 3A05[B3030300]                 <1>         cmp     al, [u.uno]     ; same process ?
0000F2F1 750C                           <1>       jne   short srunseq_2 ; no
                                        <1>
0000F2F3 8A25[A9030300]                 <1>       mov   ah, [u.pri]  ; present/current priority
0000F2F9 80FC02                         <1>       cmp   ah, 2        ; 'run for event' priority level
0000F2FC 7221                           <1>       jb    short srunseq_6 ; no
                                        <1>
                                        srunseq_1:
                                        <1>       ; there is nothing to do!
0000F2FE C3                             <1>       retn
                                        <1>
                                        srunseq_2:
                                        <1>       ;;this not necessary ! 23/12/2016
                                        <1>         ;;cmp      al, nproc    ; number of processes = 16
                                        <1>       ;;jnb  short srunseq_5    ; error ! invalid process number
                                        <1>
                                        <1>       ; dl = priority
0000F2FF 80FA02                         <1>       cmp   dl, 2        ; event queue
0000F302 72FA                           <1>       jb    short srunseq_1 ; requested process is not present
                                        <1>                             ; process and priority of requested
                                        <1>                             ; process is not high (event),
                                        <1>                             ; there is nothing to do!
                                        <1>
                                        <1>       ; requested process is not present process
                                        <1>       ; & priority of requested process is high
0000F304 3A15[A9030300]                 <1>       cmp   dl, [u.pri]  ; priority of present process
0000F30A 7606                           <1>       jna   short srunseq_3 ; is high, also
                                        <1>       ;
                                        <1>       ; present process will be swapped/switched out
0000F30C FE05[B55F0100]                 <1>       inc   byte [p_change] ; 1
                                        <1>
                                        srunseq_3:
                                        <1>       ; add process to 'runq_event' queue for new event
0000F312 BB[52030300]                   <1>       mov   ebx, runq_event ; high priority run queue
                                        <1>
                                        srunseq_4:
                                        <1>       ; al = process number
                                        <1>       ; ebx = run queue
0000F317 E8A6F4FFFF                     <1>       call  putlu
0000F31C C3                             <1>       retn
                                        <1>
                                        srunseq_5:
0000F31D F5                             <1>       cmc
0000F31E C3                             <1>       retn
                                        <1>
                                        srunseq_6:
                                        <1>       ; present priority of the process is not high
                                        <1>
0000F31F 8815[A9030300]                 <1>       mov   [u.pri], dl ; new priority
                                        <1>                         ; (will be used by 'tswap')
                                        <1>
0000F325 80FA02                         <1>       cmp   dl, 2        ; high priority ?
0000F328 72F3                           <1>       jb    short srunseq_5 ; no, there is nothing to do
                                        <1>                             ; in addition
                                        <1>
                                        <1>       ; process must be added to relevant run queue, here!
                                        <1>       ; (new priority is high/event priority and process
                                        <1>       ; will not be added to a run queue by 'tswap')
                                        <1>
0000F32A BB[54030300]                   <1>       mov   ebx, runq_normal ; 'run as regular' queue
                                        <1>
0000F32F 20E4                           <1>       and   ah, ah  ;  previous value of [u.pri]
0000F331 75E4                           <1>       jnz   short srunseq_4
                                        <1>
0000F333 43                             <1>       inc   ebx
0000F334 43                             <1>       inc   ebx
                                        <1>       ; ebx = runq_background ; 'run on backgroud' queue
                                        <1>
0000F335 EBE0                           <1>       jmp   short srunseq_4
                                        <1> clock:
```

```
                                <1>       ; 23/05/2016
                                <1>       ; 22/05/2016
                                <1>       ; 20/05/2016
                                <1>       ; 19/05/2016 - TRDOS 386 (TRDOS v2.0)
                                <1>       ; 14/05/2015 - 14/10/2015 (Retro UNIX 386 v1)
                                <1>       ; 07/12/2013 - 10/04/2014 (Retro UNIX 8086 v1)
                                <1>
0000F337 803D[A8030300]00       <1>       cmp    byte [u.quant], 0
0000F33E 772C                   <1>       ja     short clk_1
                                <1>       ;
0000F340 803D[B3030300]01       <1>       cmp     byte [u.uno], 1 ; /etc/init ? (for Retro UNIX 8086 & 386 v1)
                                <1>                       ; MainProg (Kernel's Command Interpreter)
                                <1>                       ; for TRDOS 386.
0000F347 7623                   <1>       jna    short clk_1 ; yes, do not swap out
                                <1>       ;
0000F349 803D[5B030300]FF       <1>       cmp     byte [sysflg], 0FFh ; user or system space ?
0000F350 7520                   <1>       jne    short clk_2      ; system space (sysflg <> 0FFh)
                                <1>       ;
0000F352 66833D[AA030300]00     <1>       cmp    word [u.intr], 0
0000F35A 7616                   <1>       jna    short clk_2
                                <1>       ;
                                <1>       ; 23/05/2016
0000F35C 803D[B65F0100]00       <1>       cmp    byte [multi_tasking], 0
0000F363 760D                   <1>       jna    short clk_2
                                <1>       ;
0000F365 FE05[B55F0100]         <1>       inc    byte [p_change] ; it is time to change running process
0000F36B C3                     <1>       retn
                                <1> clk_1:
0000F36C FE0D[A8030300]         <1>       dec    byte [u.quant]
                                <1> clk_2:
0000F372 C3                     <1>       retn   ; return to (hardware) timer interrupt routine
                                <1>
                                <1> ; 12/10/2017
                                <1> ; 15/01/2017
                                <1> ; 14/01/2017
                                <1> ; 07/01/2017
                                <1> ; 02/01/2017
                                <1> ; 17/08/2016
                                <1> ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
                                <1> int34h:  ; #IOCTL# (I/O port access support for ring 3)
                                <1> ; 23/05/2016
                                <1>       ; 20/06/2016
                                <1>       ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
                                <1>       ;
                                <1>       ; INPUT ->
                                <1>       ;     AH = 0 -> read port  (physical IO port) -byte-
                                <1>       ;     AH = 1 -> write port (physical IO port) -byte-
                                <1>       ;           AL = data byte
                                <1>       ;     AH = 2 -> read port  (physical IO port) -word-
                                <1>       ;     AH = 3 -> write port (physical IO port) -word-
                                <1>       ;           BX = data word
                                <1>       ;     AH = 4 -> read port  (physical IO port) -dword-
                                <1>       ;     AH = 5 -> write port (physical IO port) -dword-
                                <1>       ;           EBX = data dword
                                <1>       ;     ; 12/10/2017
                                <1>       ;     AH = 6 -> read port (physical IO port) twice -byte-
                                <1>       ;     AH = 7 -> write port (physical IO port) twice -byte-
                                <1>       ;           BX = data word
                                <1>       ;
                                <1>       ;     DX = Port number (<= 0FFFFh)
                                <1>       ;
                                <1>       ; OUTPUT ->
                                <1>       ;     AL = data byte (in al, dx)
                                <1>       ;     AX = data word (in ax, dx)
                                <1>       ;     EAX = data dword (in eax, dx)
                                <1>       ;
                                <1>       ;     (ECX = actual TRANSFER COUNT for string functions)
                                <1>       ;
                                <1>       ;
                                <1>       ; Modified registers: EAX
                                <1>       ;
                                <1>
                                <1>       ;cmp   ah, 5
                                <1>       ;ja    short int34h_5 ; invalid function !
                                <1>
                                <1>       ; 12/10/2017
0000F373 80FC07                 <1>       cmp    ah, 7
0000F376 7743                   <1>       ja     short int34h_5 ; invalid function !
                                <1>
                                <1>       ;; 15/01/2017
                                <1>       ; 14/01/2017
                                <1>       ; 02/01/2017
                                <1>       ;;mov  byte [ss:intflg], 34h     ; IOCTL interrupt
0000F378 FB                     <1>       sti
                                <1>
                                <1>       ;sti   ; enable interrupts
0000F379 80642408FE             <1>       and    byte [esp+8], 11111110b   ; clear carry bit of eflags register
                                <1>
0000F37E 80FC01                 <1>       cmp    ah, 1
0000F381 7205                   <1>       jb     short int34h_0
0000F383 7705                   <1>       ja     short int34h_1
                                <1>
0000F385 EE                     <1>       out    dx, al
                                <1>       ;iretd
0000F386 EB01                   <1>       jmp short int34h_iret
                                <1>
                                <1> int34h_0:
0000F388 EC                     <1>       in     al, dx
                                <1>       ;iretd
                                <1> int34h_iret:
                                <1>       ;cli   ; 07/01/2017
                                <1>       ;; 15/01/2017
                                <1>       ;;mov  byte [ss:intflg], 0 ; reset
```

```
45004 0000F389 CF                <1>       iretd
45005                            <1>
45006                            <1> int34h_1:
45007 0000F38A F6C401            <1>       test   ah, 1
45008 0000F38D 7516              <1>       jnz    short int34h_3 ; out
45009                            <1>
45010                            <1>       ; in
45011 0000F38F 80FC02            <1>       cmp    ah, 2
45012 0000F392 7707              <1>       ja     short int34h_2
45013                            <1>
45014 0000F394 6689D8            <1>       mov    ax, bx
45015 0000F397 66ED              <1>       in     ax, dx
45016                            <1>       ;iretd
45017 0000F399 EBEE              <1>       jmp    short int34h_iret
45018                            <1>
45019                            <1> int34h_2:
45020 0000F39B 80FC04            <1>       cmp    ah, 4
45021 0000F39E 772C              <1>       ja     short int34h_7      ; 12/10/2017
45022                            <1>       ; ah = 4
45023 0000F3A0 89D8              <1>       mov    eax, ebx
45024 0000F3A2 ED                <1>       in     eax, dx
45025                            <1>       ;iretd
45026 0000F3A3 EBE4              <1>       jmp    short int34h_iret
45027                            <1>
45028                            <1> int34h_3:
45029 0000F3A5 80FC03            <1>       cmp    ah, 3
45030 0000F3A8 7707              <1>       ja     short int34h_4
45031                            <1>
45032 0000F3AA 6689D8            <1>       mov    ax, bx
45033 0000F3AD 66EF              <1>       out    dx, ax
45034                            <1>       ;iretd
45035 0000F3AF EBD8              <1>       jmp    short int34h_iret
45036                            <1>
45037                            <1> int34h_4:
45038 0000F3B1 80FC05            <1>       cmp    ah, 5
45039 0000F3B4 770B              <1>       ja     short int34h_6      ; 12/10/2017
45040                            <1>       ; ah = 5
45041 0000F3B6 89D8              <1>       mov    eax, ebx
45042 0000F3B8 EF                <1>       out    dx, eax
45043                            <1>       ;iretd
45044 0000F3B9 EBCE              <1>       jmp    short int34h_iret
45045                            <1>
45046                            <1> int34h_5:
45047 0000F3BB 804C240801        <1>       or     byte [esp+8], 1     ; set carry bit of eflags register
45048 0000F3C0 CF                <1>       iretd
45049                            <1>
45050                            <1>       ; 12/10/2017
45051                            <1> int34h_6:
45052 0000F3C1 6689D8            <1>       mov    ax, bx
45053 0000F3C4 EE                <1>       out    dx, al
45054 0000F3C5 EB00              <1>       jmp    short $+2
45055 0000F3C7 86E0              <1>       xchg   ah, al
45056 0000F3C9 EE                <1>       out    dx, al
45057                            <1>       ;xchg  al, ah
45058                            <1>       ;iretd
45059 0000F3CA EB06              <1>       jmp    short int34h_8
45060                            <1> int34h_7:
45061 0000F3CC EC                <1>       in     al, dx
45062 0000F3CD EB00              <1>       jmp    short $+2
45063 0000F3CF 88C4              <1>       mov    ah, al
45064 0000F3D1 EC                <1>       in     al, dx
45065                            <1> int34h_8:
45066 0000F3D2 86C4              <1>       xchg   al, ah
45067 0000F3D4 CF                <1>       iretd
45068                            <1>
45069                            <1>
45070                            <1> INT4Ah:
45071                            <1>       ; 24/01/2016
45072                            <1>       ; this procedure will be called by 'RTC_INT' (in 'timer.s')
45073 0000F3D5 C3                <1>       retn
45074                            <1>
45075                            <1> ; u0.s
45076                            <1> ; Retro UNIX 386 v1 Kernel (v0.2) - SYS0.INC
45077                            <1> ; Last Modification: 20/11/2015
45078                            <1>
45079                            <1> com2_int:
45080                            <1>       ; 07/11/2015
45081                            <1>       ; 24/10/2015
45082                            <1>       ; 23/10/2015
45083                            <1>       ; 14/03/2015 (Retro UNIX 386 v1 - Beginning)
45084                            <1>       ; 28/07/2014 (Retro UNIX 8086 v1)
45085                            <1>       ; < serial port 2 interrupt handler >
45086                            <1>       ;
45087 0000F3D6 890424            <1>       mov    [esp], eax ; overwrite call return address
45088                            <1>       ;push  eax
45089 0000F3D9 66B80900          <1>       mov    ax, 9
45090 0000F3DD EB07              <1>       jmp    short comm_int
45091                            <1> com1_int:
45092                            <1>       ; 07/11/2015
45093                            <1>       ; 24/10/2015
45094 0000F3DF 890424            <1>       mov    [esp], eax ; overwrite call return address
45095                            <1>       ; 23/10/2015
45096                            <1>       ;push  eax
45097 0000F3E2 66B80800          <1>       mov    ax, 8
45098                            <1> comm_int:
45099                            <1>       ; 20/11/2015
45100                            <1>       ; 18/11/2015
45101                            <1>       ; 17/11/2015
45102                            <1>       ; 16/11/2015
45103                            <1>       ; 09/11/2015
45104                            <1>       ; 08/11/2015
45105                            <1>       ; 07/11/2015
45106                            <1>       ; 06/11/2015 (serial4.asm, 'serial')
```

```
                                <1>       ; 01/11/2015
                                <1>       ; 26/10/2015
                                <1>       ; 23/10/2015
0000F3E6 53                     <1>       push   ebx
0000F3E7 56                     <1>       push   esi
0000F3E8 57                     <1>       push   edi
0000F3E9 1E                     <1>       push   ds
0000F3EA 06                     <1>       push   es
                                <1>       ; 18/11/2015
0000F3EB 0F20DB                 <1>       mov    ebx, cr3
0000F3EE 53                     <1>       push   ebx ; ****
                                <1>       ;
0000F3EF 51                     <1>       push   ecx ; ***
0000F3F0 52                     <1>       push   edx ; **
                                <1>       ;
0000F3F1 BB10000000             <1>       mov    ebx, KDATA
0000F3F6 8EDB                   <1>       mov    ds, bx
0000F3F8 8EC3                   <1>       mov    es, bx
                                <1>       ;
0000F3FA 8B0D[20520100]         <1>       mov    ecx, [k_page_dir]
0000F400 0F22D9                 <1>       mov    cr3, ecx
                                <1>       ; 20/11/2015
                                <1>       ; Interrupt identification register
0000F403 66BAFA02               <1>       mov    dx, 2FAh ; COM2
                                <1>       ;
0000F407 3C08                   <1>       cmp    al, 8
0000F409 7702                   <1>       ja     short com_i0
                                <1>       ;
                                <1>       ; 20/11/2015
                                <1>       ; 17/11/2015
                                <1>       ; 16/11/2015
                                <1>       ; 15/11/2015
                                <1>       ; 24/10/2015
                                <1>       ; 14/03/2015 (Retro UNIX 386 v1 - Beginning)
                                <1>       ; 28/07/2014 (Retro UNIX 8086 v1)
                                <1>       ; < serial port 1 interrupt handler >
                                <1>       ;
0000F40B FEC6                   <1>       inc    dh ; 3FAh ; COM1 Interrupt id. register
                                <1> com_i0:
                                <1>       ;push  eax ; *
                                <1>       ; 07/11/2015
0000F40D A2[8A520100]           <1>       mov    byte [ccomport], al
                                <1>       ; 09/11/2015
0000F412 0FB7D8                 <1>       movzx  ebx, ax ; 8 or 9
                                <1>       ; 17/11/2015
                                <1>       ; reset request for response status
0000F415 88A3[80520100]         <1>       mov    [ebx+req_resp-8], ah ; 0
                                <1>       ;
                                <1>       ; 20/11/2015
0000F41B EC                     <1>       in     al, dx       ; read interrupt id. register
0000F41C EB00                   <1>       JMP    $+2          ; I/O DELAY
0000F41E 2404                   <1>       and    al, 4        ; received data available?
0000F420 7470                   <1>       jz     short com_eoi; (transmit. holding reg. empty)
                                <1>       ;
                                <1>       ; 20/11/2015
0000F422 80EA02                 <1>       sub    dl, 3FAh-3F8h; data register (3F8h, 2F8h)
0000F425 EC                     <1>       in     al, dx       ; read character
                                <1>       ;JMP   $+2          ; I/O DELAY
                                <1>       ; 08/11/2015
                                <1>       ; 07/11/2015
0000F426 89DE                   <1>       mov    esi, ebx
0000F428 89DF                   <1>       mov    edi, ebx
0000F42A 81C6[84520100]         <1>       add    esi, rchar - 8 ; points to last received char
0000F430 81C7[86520100]         <1>       add    edi, schar - 8 ; points to last sent char
0000F436 8806                   <1>       mov    [esi], al ; received char (current char)
                                <1>       ; query
0000F438 20C0                   <1>       and    al, al
0000F43A 7527                   <1>       jnz    short com_i2
                                <1>       ; response
                                <1>       ; 17/11/2015
                                <1>       ; set request for response status
0000F43C FE83[80520100]         <1>         inc     byte [ebx+req_resp-8] ; 1
                                <1>       ;
0000F442 6683C205               <1>       add    dx, 3FDh-3F8h; (3FDh, 2FDh)
0000F446 EC                     <1>       in     al, dx       ; read line status register
0000F447 EB00                   <1>       JMP    $+2          ; I/O DELAY
0000F449 2420                   <1>       and    al, 20h            ; transmitter holding reg. empty?
0000F44B 7445                   <1>       jz     short com_eoi      ; no
0000F44D B0FF                   <1>       mov    al, 0FFh     ; response
0000F44F 6683EA05               <1>       sub    dx, 3FDh-3F8h      ; data port (3F8h, 2F8h)
0000F453 EE                     <1>       out    dx, al       ; send on serial port
                                <1>       ; 17/11/2015
0000F454 803F00                 <1>       cmp    byte [edi], 0   ; query ? (schar)
0000F457 7502                   <1>       jne    short com_i1    ; no
0000F459 8807                   <1>       mov    [edi], al    ; 0FFh (responded)
                                <1> com_i1:
                                <1>       ; 17/11/2015
                                <1>       ; reset request for response status (again)
0000F45B FE8B[80520100]         <1>         dec     byte [ebx+req_resp-8] ; 0
0000F461 EB2F                   <1>       jmp    short com_eoi
                                <1> com_i2:
                                <1>       ; 08/11/2015
0000F463 3CFF                   <1>       cmp    al, 0FFh     ; (response ?)
0000F465 7417                   <1>       je     short com_i3 ; (check for response signal)
                                <1>       ; 07/11/2015
0000F467 3C04                   <1>       cmp    al, 04h      ; EOT
0000F469 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)
0000F46B 861D[4E520100]         <1>       xchg   bl, [ptty]  ; tty number (8 or 9)
0000F471 E8606FFFFF             <1>       call   ctrlbrk
```

```
45210 0000F476 861D[4E520100]      <1>       xchg   [ptty], bl ; (restore ptty value and BL value)
45211                              <1>       ;mov   al, 04h ; EOT
45212                              <1>       ; 08/11/2015
45213 0000F47C EB09                <1>       jmp    short com_i4
45214                              <1> com_i3:
45215                              <1>       ; 08/11/2015
45216                              <1>       ; If 0FFh has been received just after a query
45217                              <1>       ; (schar, ZERO), it is a response signal.
45218                              <1>       ; 17/11/2015
45219 0000F47E 803F00              <1>         cmp     byte [edi], 0 ; query ? (schar)
45220 0000F481 7704                <1>       ja     short com_i4 ; no
45221                              <1>       ; reset query status (schar)
45222 0000F483 8807                <1>       mov    [edi], al ; 0FFh
45223 0000F485 FEC0                <1>       inc    al ; 0
45224                              <1> com_i4:
45225                              <1>       ; 27/07/2014
45226                              <1>       ; 09/07/2014
45227 0000F487 D0E3                <1>       shl    bl, 1
45228 0000F489 81C3[50520100]      <1>       add    ebx, ttychr
45229                              <1>       ; 23/07/2014 (always overwrite)
45230                              <1>       ;;cmp  word [ebx], 0
45231                              <1>       ;;ja   short com_eoi
45232                              <1>       ;
45233 0000F48F 668903              <1>       mov    [ebx], ax   ; Save ascii code
45234                              <1>                          ; scan code = 0
45235                              <1> com_eoi:
45236                              <1>       ;mov   al, 20h
45237                              <1>       ;out   20h, al         ; end of interrupt
45238                              <1>       ;
45239                              <1>       ; 07/11/2015
45240                              <1>             ;pop   eax ; *
45241 0000F492 A0[8A520100]        <1>       mov    al, byte [ccomport] ; current COM port
45242                              <1>        ; al = tty number (8 or 9)
45243 0000F497 E85E010000          <1>         call wakeup
45244                              <1> com_iret:
45245                              <1>       ; 23/10/2015
45246 0000F49C 5A                  <1>       pop    edx ; **
45247 0000F49D 59                  <1>       pop    ecx ; ***
45248                              <1>       ; 18/11/2015
45249                              <1>       ;pop   eax ; ****
45250                              <1>       ;mov   cr3, eax
45251                              <1>       ;jmp   iiret
45252 0000F49E E93D16FFFF          <1>       jmp    iiretp
45253                              <1>
45254                              <1> ;iiretp: ; 01/09/2015
45255                              <1> ;     ; 28/08/2015
45256                              <1> ;     pop    eax ; (*) page directory
45257                              <1> ;     mov    cr3, eax
45258                              <1> ;iiret:
45259                              <1> ;     ; 22/08/2014
45260                              <1> ;     mov    al, 20h ; END OF INTERRUPT COMMAND TO 8259
45261                              <1> ;     out    20h, al      ; 8259 PORT
45262                              <1> ;     ;
45263                              <1> ;     pop    es
45264                              <1> ;     pop    ds
45265                              <1> ;     pop    edi
45266                              <1> ;     pop    esi
45267                              <1> ;     pop    ebx ; 29/08/2014
45268                              <1> ;     pop    eax
45269                              <1> ;     iretd
45270                              <1>
45271                              <1> sp_init:
45272                              <1>       ; 07/11/2015
45273                              <1>       ; 29/10/2015
45274                              <1>       ; 26/10/2015
45275                              <1>       ; 23/10/2015
45276                              <1>       ; 29/06/2015
45277                              <1>       ; 14/03/2015 (Retro UNIX 386 v1 - 115200 baud)
45278                              <1>       ; 28/07/2014 (Retro UNIX 8086 v1 - 9600 baud)
45279                              <1>       ; Initialization of Serial Port Communication Parameters
45280                              <1>       ; (COM1 base port address = 3F8h, COM1 Interrupt = IRQ 4)
45281                              <1>       ; (COM2 base port address = 2F8h, COM1 Interrupt = IRQ 3)
45282                              <1>       ;
45283                              <1>       ; ((Modified registers: EAX, ECX, EDX, EBX))
45284                              <1>       ;
45285                              <1>       ; INPUT:  (29/06/2015)
45286                              <1>       ;     AL = 0 for COM1
45287                              <1>       ;          1 for COM2
45288                              <1>       ;     AH = Communication parameters
45289                              <1>       ;
45290                              <1>       ;  (*) Communication parameters (except BAUD RATE):
45291                              <1>       ;     Bit    4      3      2      1      0
45292                              <1>       ;            -PARITY--   STOP BIT  -WORD LENGTH-
45293                              <1>       ;  this one -->     00 = none    0 = 1 bit  11 = 8 bits
45294                              <1>       ;            01 = odd     1 = 2 bits   10 = 7 bits
45295                              <1>       ;            11 = even
45296                              <1>       ;  Baud rate setting bits: (29/06/2015)
45297                              <1>       ;            Retro UNIX 386 v1 feature only !
45298                              <1>       ;     Bit    7    6    5  | Baud rate
45299                              <1>       ;            ------------------------
45300                              <1>       ;     value  0    0    0  | Default (Divisor = 1)
45301                              <1>       ;            0    0    1  | 9600 (12)
45302                              <1>       ;            0    1    0  | 19200 (6)
45303                              <1>       ;            0    1   1  | 38400 (3)
45304                              <1>       ;            1    0   0  | 14400 (8)
45305                              <1>       ;            1    0   1  | 28800 (4)
45306                              <1>       ;            1    1    0  | 57600 (2)
45307                              <1>       ;            1    1    1  | 115200 (1)
45308                              <1>
45309                              <1>       ; References:
45310                              <1>       ; (1) IBM PC-XT Model 286 BIOS Source Code
45311                              <1>       ;     RS232.ASM --- 10/06/1985 COMMUNICATIONS BIOS (RS232)
45312                              <1>       ; (2) Award BIOS 1999 - ATORGS.ASM
```

```
45313                                  <1>       ; (3) http://wiki.osdev.org/Serial_Ports
45314                                  <1>       ;
45315                                  <1>       ; Set communication parameters for COM1 (= 03h)
45316                                  <1>       ;
45317 0000F4A3 BB[86520100]            <1>       mov    ebx, com1p          ; COM1 parameters
45318 0000F4A8 66BAF803                <1>       mov    dx, 3F8h            ; COM1
45319                                  <1>        ; 29/10/2015
45320 0000F4AC 66B90103                <1>       mov    cx, 301h  ; divisor = 1 (115200 baud)
45321 0000F4B0 E86F000000              <1>       call   sp_i3  ; call A4
45322 0000F4B5 A880                    <1>       test   al, 80h
45323 0000F4B7 7410                    <1>       jz     short sp_i0 ; OK..
45324                                  <1>              ; Error !
45325                                  <1>       ;mov   dx, 3F8h
45326 0000F4B9 80EA05                  <1>       sub    dl, 5 ; 3FDh -> 3F8h
45327 0000F4BC 66B90E03                <1>       mov    cx, 30Eh  ; divisor = 12 (9600 baud)
45328 0000F4C0 E85F000000              <1>       call   sp_i3  ; call A4
45329 0000F4C5 A880                    <1>       test   al, 80h
45330 0000F4C7 7508                    <1>       jnz    short sp_i1
45331                                  <1> sp_i0:
45332                                  <1>         ; (Note: Serial port interrupts will be disabled here...)
45333                                  <1>         ; (INT 14h initialization code disables interrupts.)
45334                                  <1>       ;
45335 0000F4C9 C603E3                  <1>       mov    byte [ebx], 0E3h ; 11100011b
45336 0000F4CC E8DC000000              <1>       call   sp_i5 ; 29/06/2015
45337                                  <1> sp_i1:
45338 0000F4D1 43                      <1>       inc    ebx
45339 0000F4D2 66BAF802                <1>       mov    dx, 2F8h            ; COM2
45340                                  <1>        ; 29/10/2015
45341 0000F4D6 66B90103                <1>       mov    cx, 301h  ; divisor = 1 (115200 baud)
45342 0000F4DA E845000000              <1>       call   sp_i3  ; call A4
45343 0000F4DF A880                    <1>       test   al, 80h
45344 0000F4E1 7410                    <1>       jz     short sp_i2 ; OK..
45345                                  <1>              ; Error !
45346                                  <1>       ;mov   dx, 2F8h
45347 0000F4E3 80EA05                  <1>       sub    dl, 5 ; 2FDh -> 2F8h
45348 0000F4E6 66B90E03                <1>       mov    cx, 30Eh  ; divisor = 12 (9600 baud)
45349 0000F4EA E835000000              <1>       call   sp_i3  ; call A4
45350 0000F4EF A880                    <1>       test   al, 80h
45351 0000F4F1 7530                    <1>       jnz    short sp_i7
45352                                  <1> sp_i2:
45353 0000F4F3 C603E3                  <1>       mov    byte [ebx], 0E3h ; 11100011b
45354                                  <1> sp_i6:
45355                                  <1>       ;; COM2 - enabling IRQ 3
45356                                  <1>       ; 07/11/2015
45357                                  <1>       ; 26/10/2015
45358 0000F4F6 9C                      <1>       pushf
45359 0000F4F7 FA                      <1>       cli
45360                                  <1>       ;
45361 0000F4F8 66BAFC02                <1>       mov    dx, 2FCh            ; modem control register
45362 0000F4FC EC                      <1>       in     al, dx                    ; read register
45363 0000F4FD EB00                    <1>       JMP    $+2                 ; I/O DELAY
45364 0000F4FF 0C08                    <1>       or     al, 8               ; enable bit 3 (OUT2)
45365 0000F501 EE                      <1>       out    dx, al              ; write back to register
45366 0000F502 EB00                    <1>       JMP    $+2                 ; I/O DELAY
45367 0000F504 66BAF902                <1>       mov    dx, 2F9h            ; interrupt enable register
45368 0000F508 EC                      <1>       in     al, dx              ; read register
45369 0000F509 EB00                    <1>       JMP    $+2                 ; I/O DELAY
45370                                  <1>       ;or    al, 1               ; receiver data interrupt enable and
45371 0000F50B 0C03                    <1>       or     al, 3               ; transmitter empty interrupt enable
45372 0000F50D EE                      <1>       out    dx, al                    ; write back to register
45373 0000F50E EB00                    <1>       JMP    $+2                 ; I/O DELAY
45374 0000F510 E421                    <1>       in     al, 21h             ; read interrupt mask register
45375 0000F512 EB00                    <1>       JMP    $+2                 ; I/O DELAY
45376 0000F514 24F7                    <1>       and    al, 0F7h            ; enable IRQ 3 (COM2)
45377 0000F516 E621                    <1>       out    21h, al             ; write back to register
45378                                  <1>       ;
45379                                  <1>       ; 23/10/2015
45380 0000F518 B8[D6F30000]            <1>       mov    eax, com2_int
45381 0000F51D A3[F5F50000]            <1>       mov    [com2_irq3], eax
45382                                  <1>       ; 26/10/2015
45383 0000F522 9D                      <1>       popf
45384                                  <1> sp_i7:
45385 0000F523 C3                      <1>       retn
45386                                  <1>
45387                                  <1> sp_i3:
45388                                  <1> ;A4:        ;----- INITIALIZE THE COMMUNICATIONS PORT
45389                                  <1>       ; 28/10/2015
45390 0000F524 FEC2                    <1>       inc    dl     ; 3F9h (2F9h); 3F9h, COM1 Interrupt enable register
45391 0000F526 B000                    <1>       mov    al, 0
45392 0000F528 EE                      <1>       out    dx, al              ; disable serial port interrupt
45393 0000F529 EB00                    <1>       JMP    $+2                 ; I/O DELAY
45394 0000F52B 80C202                  <1>       add    dl, 2  ; 3FBh (2FBh); COM1 Line control register (3FBh)
45395 0000F52E B080                    <1>       mov    al, 80h
45396 0000F530 EE                      <1>       out    dx, al              ; SET DLAB=1 ; divisor latch access bit
45397                                  <1>       ;----- SET BAUD RATE DIVISOR
45398                                  <1>       ; 26/10/2015
45399 0000F531 80EA03                  <1>       sub    dl, 3   ; 3F8h (2F8h)     ; register for least significant byte
45400                                  <1>                                  ; of the divisor value
45401 0000F534 88C8                    <1>       mov    al, cl ; 1
45402 0000F536 EE                      <1>       out    dx, al              ; 1 = 115200 baud (Retro UNIX 386 v1)
45403                                  <1>                                  ; 2 = 57600 baud
45404                                  <1>                                  ; 3 = 38400 baud
45405                                  <1>                                  ; 6 = 19200 baud
45406                                  <1>                                  ; 12 = 9600 baud (Retro UNIX 8086 v1)
45407 0000F537 EB00                    <1>       JMP    $+2                 ; I/O DELAY
45408 0000F539 28C0                    <1>       sub    al, al
45409 0000F53B FEC2                    <1>       inc    dl      ; 3F9h (2F9h)     ; register for most significant byte
45410                                  <1>                                  ; of the divisor value
45411 0000F53D EE                      <1>       out    dx, al ; 0
45412 0000F53E EB00                    <1>       JMP    $+2                 ; I/O DELAY
45413                                  <1>       ;
45414 0000F540 88E8                    <1>       mov    al, ch ; 3          ; 8 data bits, 1 stop bit, no parity
45415                                  <1>       ;and   al, 1Fh ; Bits 0,1,2,3,4
```

```
45416 0000F542 80C202              <1>       add    dl, 2  ; 3FBh (2FBh); Line control register
45417 0000F545 EE                  <1>       out    dx, al
45418 0000F546 EB00                <1>       JMP    $+2                ; I/O DELAY
45419                              <1>       ; 29/10/2015
45420 0000F548 FECA                <1>       dec    dl     ; 3FAh (2FAh); FIFO Control register (16550/16750)
45421 0000F54A 30C0                <1>       xor    al, al             ; 0
45422 0000F54C EE                  <1>       out    dx, al             ; Disable FIFOs (reset to 8250 mode)
45423 0000F54D EB00                <1>       JMP    $+2
45424                              <1> sp_i4:
45425                              <1> ;A18: ;----- COMM PORT STATUS ROUTINE
45426                              <1>       ; 29/06/2015 (line status after modem status)
45427 0000F54F 80C204              <1>       add    dl, 4  ; 3FEh (2FEh); Modem status register
45428                              <1> sp_i4s:
45429 0000F552 EC                  <1>       in     al, dx             ; GET MODEM CONTROL STATUS
45430 0000F553 EB00                <1>       JMP    $+2                ; I/O DELAY
45431 0000F555 88C4                <1>       mov    ah, al             ; PUT IN (AH) FOR RETURN
45432 0000F557 FECA                <1>       dec    dl     ; 3FDh (2FDh); POINT TO LINE STATUS REGISTER
45433                              <1>                                  ; dx = 3FDh for COM1, 2FDh for COM2
45434 0000F559 EC                  <1>       in     al, dx             ; GET LINE CONTROL STATUS
45435                              <1>       ; AL = Line status, AH = Modem status
45436 0000F55A C3                  <1>       retn
45437                              <1>
45438                              <1> sp_status:
45439                              <1>       ; 29/06/2015
45440                              <1>       ; 27/06/2015 (Retro UNIX 386 v1)
45441                              <1>       ; Get serial port status
45442 0000F55B 66BAFE03            <1>       mov    dx, 3FEh           ; Modem status register (COM1)
45443 0000F55F 28C6                <1>       sub    dh, al             ; dh = 2 for COM2 (al = 1)
45444                              <1>                                  ; dx = 2FEh for COM2
45445 0000F561 EBEF                <1>       jmp    short sp_i4s
45446                              <1>
45447                              <1> sp_setp: ; Set serial port communication parameters
45448                              <1>       ; 07/11/2015
45449                              <1>       ; 29/10/2015
45450                              <1>       ; 29/06/2015
45451                              <1>       ; Retro UNIX 386 v1 feature only !
45452                              <1>       ;
45453                              <1>       ; INPUT:
45454                              <1>       ;     AL = 0 for COM1
45455                              <1>       ;          1 for COM2
45456                              <1>       ;     AH = Communication parameters (*)
45457                              <1>       ; OUTPUT:
45458                              <1>       ;     CL = Line status
45459                              <1>       ;     CH = Modem status
45460                              <1>       ;   If cf = 1 -> Error code in [u.error]
45461                              <1>       ;             'invalid parameter !'
45462                              <1>       ;                   or
45463                              <1>       ;             'device not ready !' error
45464                              <1>       ;
45465                              <1>       ;  (*) Communication parameters (except BAUD RATE):
45466                              <1>       ;     Bit    4     3     2     1     0
45467                              <1>       ;            -PARITY--   STOP BIT  -WORD LENGTH-
45468                              <1>       ;  this one -->    00 = none    0 = 1 bit  11 = 8 bits
45469                              <1>       ;            01 = odd     1 = 2 bits   10 = 7 bits
45470                              <1>       ;            11 = even
45471                              <1>       ;  Baud rate setting bits: (29/06/2015)
45472                              <1>       ;            Retro UNIX 386 v1 feature only !
45473                              <1>       ;     Bit    7    6    5  | Baud rate
45474                              <1>       ;            ------------------------
45475                              <1>       ;     value  0    0    0  | Default (Divisor = 1)
45476                              <1>       ;            0    0    1  | 9600 (12)
45477                              <1>       ;            0    1    0  | 19200 (6)
45478                              <1>       ;            0    1   1  | 38400 (3)
45479                              <1>       ;            1    0   0  | 14400 (8)
45480                              <1>       ;            1    0   1  | 28800 (4)
45481                              <1>       ;            1    1    0  | 57600 (2)
45482                              <1>       ;            1    1    1  | 115200 (1)
45483                              <1>       ;
45484                              <1>       ; (COM1 base port address = 3F8h, COM1 Interrupt = IRQ 4)
45485                              <1>       ; (COM2 base port address = 2F8h, COM1 Interrupt = IRQ 3)
45486                              <1>       ;
45487                              <1>       ; ((Modified registers: EAX, ECX, EDX, EBX))
45488                              <1>       ;
45489 0000F563 66BAF803            <1>       mov    dx, 3F8h
45490 0000F567 BB[86520100]        <1>       mov    ebx, com1p ; COM1 control byte offset
45491 0000F56C 3C01                <1>       cmp    al, 1
45492 0000F56E 776B                <1>       ja     short sp_invp_err
45493 0000F570 7203                <1>       jb     short sp_setp1 ;  COM1 (AL = 0)
45494 0000F572 FECE                <1>       dec    dh ; 2F8h
45495 0000F574 43                  <1>       inc    ebx ; COM2 control byte offset
45496                              <1> sp_setp1:
45497                              <1>       ; 29/10/2015
45498 0000F575 8823                <1>       mov    [ebx], ah
45499 0000F577 0FB6CC              <1>       movzx  ecx, ah
45500 0000F57A C0E905              <1>       shr    cl, 5 ; -> baud rate index
45501 0000F57D 80E41F              <1>       and    ah, 1Fh ; communication parameters except baud rate
45502 0000F580 8A81[EAF50000]      <1>       mov    al, [ecx+b_div_tbl]
45503 0000F586 6689C1              <1>       mov    cx, ax
45504 0000F589 E896FFFFFF          <1>       call   sp_i3
45505 0000F58E 6689C1              <1>       mov    cx, ax ; CL = Line status, CH = Modem status
45506 0000F591 A880                <1>       test   al, 80h
45507 0000F593 740F                <1>       jz     short sp_setp2
45508 0000F595 C603E3              <1>         mov     byte [ebx], 0E3h ; Reset to initial value (11100011b)
45509                              <1> stp_dnr_err:
45510 0000F598 C705[C8030300]0F00- <1>       mov    dword [u.error], ERR_DEV_NOT_RDY ; 'device not ready !'
45511 0000F5A0 0000                <1>
45512                              <1>       ; CL = Line status, CH = Modem status
45513 0000F5A2 F9                  <1>       stc
45514 0000F5A3 C3                  <1>       retn
45515                              <1> sp_setp2:
45516 0000F5A4 80FE02              <1>       cmp    dh, 2 ; COM2 (2F?h)
45517 0000F5A7 0F8649FFFFFF        <1>         jna     sp_i6
45518                              <1>                   ; COM1 (3F?h)
```

```
45519                                  <1> sp_i5:
45520                                  <1>       ; 07/11/2015
45521                                  <1>       ; 26/10/2015
45522                                  <1>       ; 29/06/2015
45523                                  <1>       ;
45524                                  <1>       ;; COM1 - enabling IRQ 4
45525 0000F5AD 9C                      <1>       pushf
45526 0000F5AE FA                      <1>       cli
45527 0000F5AF 66BAFC03                <1>       mov    dx, 3FCh           ; modem control register
45528 0000F5B3 EC                      <1>       in     al, dx                   ; read register
45529 0000F5B4 EB00                    <1>       JMP    $+2                ; I/O DELAY
45530 0000F5B6 0C08                    <1>       or     al, 8              ; enable bit 3 (OUT2)
45531 0000F5B8 EE                      <1>       out    dx, al             ; write back to register
45532 0000F5B9 EB00                    <1>       JMP    $+2                ; I/O DELAY
45533 0000F5BB 66BAF903                <1>       mov    dx, 3F9h           ; interrupt enable register
45534 0000F5BF EC                      <1>       in     al, dx             ; read register
45535 0000F5C0 EB00                    <1>       JMP    $+2                ; I/O DELAY
45536                                  <1>       ;or    al, 1              ; receiver data interrupt enable and
45537 0000F5C2 0C03                    <1>       or     al, 3              ; transmitter empty interrupt enable
45538 0000F5C4 EE                      <1>       out    dx, al                   ; write back to register
45539 0000F5C5 EB00                    <1>       JMP    $+2                ; I/O DELAY
45540 0000F5C7 E421                    <1>       in     al, 21h            ; read interrupt mask register
45541 0000F5C9 EB00                    <1>       JMP    $+2                ; I/O DELAY
45542 0000F5CB 24EF                    <1>       and    al, 0EFh           ; enable IRQ 4 (COM1)
45543 0000F5CD E621                    <1>       out    21h, al            ; write back to register
45544                                  <1>       ;
45545                                  <1>       ; 23/10/2015
45546 0000F5CF B8[DFF30000]            <1>       mov    eax, com1_int
45547 0000F5D4 A3[F1F50000]            <1>       mov    [com1_irq4], eax
45548                                  <1>       ; 26/10/2015
45549 0000F5D9 9D                      <1>       popf
45550 0000F5DA C3                      <1>       retn
45551                                  <1>
45552                                  <1> sp_invp_err:
45553 0000F5DB C705[C8030300]1700-      <1>       mov    dword [u.error], ERR_INV_PARAMETER ; 'invalid parameter !'
45554 0000F5E3 0000                    <1>
45555 0000F5E5 31C9                    <1>       xor    ecx, ecx
45556 0000F5E7 49                      <1>       dec    ecx ; 0FFFFh
45557 0000F5E8 F9                      <1>       stc
45558 0000F5E9 C3                      <1>       retn
45559                                  <1>
45560                                  <1> ; 29/10/2015
45561                                  <1> b_div_tbl: ; Baud rate divisor table (115200/divisor)
45562 0000F5EA 010C0603080401          <1>       db 1, 12, 6, 3, 8, 4, 1
45563                                  <1>
45564                                  <1>
45565                                  <1> ; 23/10/2015
45566                                  <1> com1_irq4:
45567 0000F5F1 [F9F50000]              <1>       dd dummy_retn
45568                                  <1> com2_irq3:
45569 0000F5F5 [F9F50000]              <1>       dd dummy_retn
45570                                  <1>
45571                                  <1> dummy_retn:
45572 0000F5F9 C3                      <1>       retn
45573                                  <1>
45574                                  <1> wakeup:
45575                                  <1>       ; 24/01/2016
45576 0000F5FA C3                      <1>       retn
45577                                  <1>
45578                                  <1> set_working_path_x:
45579                                  <1>             ; 17/10/2016 (TRDOS 386 - FFF & FNF)
45580 0000F5FB 66B80100                <1>             mov    ax, 1
45581                                  <1>                    ; File name is needed/forced (AL=1)
45582                                  <1>                    ; Change directory as temporary (AH=0)
45583                                  <1>
45584                                  <1>             ; This is needed for preventing wrong Find Next File
45585                                  <1>             ; system call after sysopen, syscreate, sysmkdir etc.
45586                                  <1>             ; Find Next File must immmediate follow Find First file)
45587                                  <1>
45588 0000F5FF 8825[D85F0100]          <1>             mov    [FFF_Valid], ah ; 0 ; reset ; 17/10/2016
45589                                  <1>
45590                                  <1> set_working_path:
45591                                  <1>             ; 16/10/2016
45592                                  <1>             ; 12/10/2016
45593                                  <1>             ; 10/10/2016
45594                                  <1>             ; 05/10/2016 - TRDOS 386 (TRDOS v2.0)
45595                                  <1>             ;
45596                                  <1>             ; TRDOS v1.0 (DIR.ASM, "proc_set_working_path")
45597                                  <1>                ; 27/01/2011 - 08/02/2011
45598                                  <1>             ; Set/Changes current drive, directory and file
45599                                  <1>             ; depending on command tail
45600                                  <1>             ; (procedure is derivated from CMD_INTR.ASM
45601                                  <1>             ; file or dir locating code of internal commands)
45602                                  <1>             ; (This procedure is prepared for INT 21H file/dir
45603                                  <1>             ; functions and also to get compact code for
45604                                  <1>             ; internal mainprog -command interpreter- commands)
45605                                  <1>             ;
45606                                  <1>             ; INPUT: DS:SI -> Command tail (ASCIIZ string)
45607                                  <1>             ; AL= 0 -> any, AL > 0 -> file name is forced
45608                                  <1>             ; AH= CD -> Change directory permanently
45609                                  <1>             ; AH <> CD -> Change directory as temporary
45610                                  <1>             ;
45611                                  <1>             ; OUTPUT: ES=DS, FindFile structure has been set
45612                                  <1>             ;       RUN_CDRV points previous current drive
45613                                  <1>             ;       DS:SI = FindFile structure address
45614                                  <1>             ;       (DS=CS)
45615                                  <1>             ;       AX, BX, CX, DX, DI will be changed
45616                                  <1>             ;   cf = 1 -> Error code in AX (AL)
45617                                  <1>             ;       stc & AX = 0 -> Bad command or path name
45618                                  <1>             ; ------------------------------------------------
45619                                  <1>             ;
45620                                  <1>             ; TRDOS 386 (05/10/2016)
45621                                  <1>             ; INPUT:
```

```
45622                                  <1>             ;     ESI = File/Directory Path (ASCIIZ string)
45623                                  <1>             ;           address in user's memory space
45624                                  <1>             ;       Al = 0 -> any
45625                                  <1>             ;       AL > 0 -> file name is forced
45626                                  <1>             ;       AH = CD -> change directory as permanent
45627                                  <1>             ;       AH <> CD -> change directory as temporary
45628                                  <1>             ;
45629                                  <1>             ; OUTPUT:
45630                                  <1>             ;     FindFile structure has been set
45631                                  <1>             ;       RUN_CDRV points previous current drive
45632                                  <1>             ;       ESI = FindFile_Name address ; 12/10/2016
45633                                  <1>             ;
45634                                  <1>             ;       cf = 1 -> Error code in EAX (AL)
45635                                  <1>             ;       stc & EAX = 0 -> Bad command or path name
45636                                  <1>             ;
45637                                  <1>             ; Modified registers: EAX, EBX, ECX, EDX, ESI, EDI
45638                                  <1>
45639 0000F605 66A3[DC5F0100]          <1>             mov    [SWP_Mode], ax
45640 0000F60B A0[E6520100]            <1>             mov    al, [Current_Drv]
45641 0000F610 30E4                    <1>             xor    ah, ah
45642 0000F612 66A3[DE5F0100]          <1>             mov    word [SWP_DRV], ax
45643                                  <1>
45644                                  <1>             ; TRDOS 386 ring 3 (user's page directory)
45645                                  <1>             ; to ring 0 (kernel's page directory)
45646                                  <1>             ; transfer modifications (05/10/2016).
45647                                  <1>
45648 0000F618 55                      <1>             push   ebp
45649 0000F619 89E5                    <1>             mov    ebp, esp
45650                                  <1>
45651 0000F61B B980000000              <1>             mov    ecx, 128 ; maximum path length = 128 bytes
45652 0000F620 29CC                    <1>             sub    esp, ecx ; reserve 128 bytes (buffer) on stack
45653 0000F622 89E7                    <1>             mov    edi, esp ; destination address (kernel space)
45654                                  <1>             ; esi = source address (virtual, in user's memory space)
45655 0000F624 E8CEF2FFFF              <1>             call   transfer_from_user_buffer
45656 0000F629 720A                    <1>             jc     short loc_swp_xor_retn
45657                                  <1>
45658 0000F62B 89E6                    <1>             mov    esi, esp ; temporary buffer (the path) on stack
45659                                  <1> loc_swp_fchar:
45660 0000F62D 8A06                    <1>             mov    al, [esi]
45661 0000F62F 3C20                    <1>             cmp    al, 20h
45662 0000F631 7711                    <1>             ja     short loc_swp_parse_path_name
45663 0000F633 740C                    <1>             je     short loc_swp_fchar_next
45664                                  <1>
45665                                  <1> loc_swp_xor_retn:
45666 0000F635 31C0                    <1>             xor    eax, eax
45667 0000F637 F9                      <1>             stc
45668                                  <1> loc_swp_retn:
45669 0000F638 89EC                    <1>             mov    esp, ebp
45670 0000F63A 5D                      <1>             pop    ebp
45671                                  <1>
45672                                  <1>             ;mov   esi, FindFile_Drv
45673 0000F63B BE[CC5C0100]            <1>             mov    esi, FindFile_Name ; 12/10/2016
45674 0000F640 C3                      <1>             retn
45675                                  <1>
45676                                  <1> loc_swp_fchar_next:
45677 0000F641 46                      <1>             inc    esi
45678 0000F642 EBE9                    <1>             jmp    short loc_swp_fchar
45679                                  <1>
45680                                  <1> loc_swp_parse_path_name:
45681 0000F644 BF[8A5C0100]            <1>             mov    edi, FindFile_Drv
45682 0000F649 E8E3ABFFFF              <1>             call   parse_path_name
45683 0000F64E 72E8                    <1>             jc     short loc_swp_retn
45684                                  <1>
45685                                  <1> loc_swp_checkfile_name:
45686 0000F650 803D[DC5F0100]00        <1>             cmp    byte [SWP_Mode], 0
45687 0000F657 761E                    <1>             jna    short loc_swp_drv
45688                                  <1>
45689                                  <1>             ; 10/10/2016 (valid file name checking)
45690 0000F659 BE[CC5C0100]            <1>             mov    esi, FindFile_Name
45691 0000F65E 803E20                  <1>             cmp    byte [esi], 20h
45692 0000F661 76D2                    <1>             jna    short loc_swp_xor_retn
45693                                  <1>
45694                                  <1>             ; 16/10/2016
45695 0000F663 C605[DB5F0100]00        <1>             mov    byte [SWP_inv_fname], 0 ; reset
45696                                  <1>             ; esi = file name address (ASCIIZ)
45697 0000F66A E85C8DFFFF              <1>             call   check_filename
45698 0000F66F 7306                    <1>             jnc    short loc_swp_drv
45699                                  <1>
45700 0000F671 FE05[DB5F0100]          <1>             inc    byte [SWP_inv_fname] ; set
45701                                  <1> loc_swp_drv:
45702 0000F677 8A35[E6520100]          <1>             mov    dh, [Current_Drv]
45703                                  <1>                ;mov       [RUN_CDRV], dh
45704                                  <1>
45705 0000F67D 8A15[8A5C0100]          <1>             mov    dl, [FindFile_Drv]
45706                                  <1>                ;cmp       dl, dh
45707 0000F683 3A15[E6520100]          <1>             cmp    dl, [Current_Drv]
45708 0000F689 740D                    <1>             je     short loc_swp_change_directory
45709                                  <1>
45710 0000F68B FE05[DF5F0100]          <1>             inc    byte [SWP_DRV_chg]
45711 0000F691 E8DA75FFFF              <1>             call   change_current_drive
45712 0000F696 72A0                    <1>             jc     short loc_swp_retn ; eax = error code
45713                                  <1>             ; eax = 0
45714                                  <1>
45715                                  <1> loc_swp_change_directory:
45716 0000F698 803D[8B5C0100]21        <1>             cmp    byte [FindFile_Directory], 21h
45717 0000F69F F5                      <1>             cmc
45718 0000F6A0 7396                    <1>             jnc    short loc_swp_retn
45719                                  <1>
45720 0000F6A2 FE05[DF5F0100]          <1>             inc    byte [SWP_DRV_chg]
45721 0000F6A8 FE05[D3060100]          <1>             inc    byte [Restore_CDIR]
45722 0000F6AE BE[8B5C0100]            <1>             mov    esi, FindFile_Directory
45723 0000F6B3 8A25[DD5F0100]          <1>             mov    ah, [SWP_Mode+1]
45724 0000F6B9 E85DA5FFFF              <1>             call   change_current_directory
```

```
45725 0000F6BE 0F8274FFFFFF        <1>             jc     loc_swp_retn ; eax = error code
45726                              <1>
45727                              <1> loc_swp_change_prompt_dir_string:
45728                              <1>             ; esi = PATH_Array
45729                              <1>             ; eax = Current Directory First Cluster
45730                              <1>             ; edi = Logical DOS Drive Description Table
45731 0000F6C4 E877A4FFFF          <1>             call   change_prompt_dir_str
45732 0000F6C9 29C0                <1>             sub    eax, eax ; 0
45733 0000F6CB E968FFFFFF          <1>             jmp    loc_swp_retn
45734                              <1>
45735                              <1> reset_working_path:
45736                              <1>             ; 06/10/2016 - TRDOS 386 (TRDOS v2.0)
45737                              <1>             ;
45738                              <1>             ; TRDOS v1.0 (DIR.ASM, "proc_reset_working_path")
45739                              <1>             ; 05/02/2011 - 08/02/2011
45740                              <1>             ;
45741                              <1>             ; Restores current drive and directory
45742                              <1>             ;
45743                              <1>             ; INPUT: none
45744                              <1>             ; OUTPUT: DL = SWP_DRV, EAX = 0 -> OK
45745                              <1>             ;
45746                              <1>             ;    AX = 0 -> ESI = Logical Dos Drv Desc. Table
45747                              <1>             ;
45748                              <1>             ;    EAX, EBX, ECX, EDX, ESI, EDI will be changed
45749                              <1>             ;
45750                              <1>
45751                              <1>
45752 0000F6D0 31C0                <1>             xor    eax, eax
45753 0000F6D2 48                  <1>             dec    eax
45754                              <1>
45755 0000F6D3 668B15[DE5F0100]    <1>             mov    dx, [SWP_DRV]
45756 0000F6DA 08F6                <1>             or     dh, dh
45757 0000F6DC 742E                <1>             jz     short loc_rwp_return
45758                              <1>
45759 0000F6DE 3A15[E6520100]      <1>             cmp    dl, [Current_Drv]
45760 0000F6E4 7407                <1>             je     short loc_rwp_restore_cdir
45761                              <1> loc_rwp_restore_cdrv:
45762 0000F6E6 E88575FFFF          <1>             call   change_current_drive
45763 0000F6EB EB10                <1>             jmp    short loc_rwp_restore_ok
45764                              <1> loc_rwp_restore_cdir:
45765 0000F6ED 31DB                <1>             xor    ebx, ebx
45766 0000F6EF 88D7                <1>             mov    bh, dl
45767 0000F6F1 BE00010900          <1>             mov    esi, Logical_DOSDisks
45768 0000F6F6 01DE                <1>             add    esi, ebx
45769                              <1>
45770 0000F6F8 E82A76FFFF          <1>             call   restore_current_directory
45771                              <1>
45772                              <1> loc_rwp_restore_ok:
45773 0000F6FD 668B15[DE5F0100]    <1>             mov    dx, [SWP_DRV]
45774 0000F704 31C0                <1>             xor    eax, eax
45775 0000F706 66A3[DF5F0100]      <1>             mov    [SWP_DRV_chg], ax
45776                              <1> loc_rwp_return:
45777 0000F70C C3                  <1>             retn
45778                              <1>
45779                              <1> get_file_name:
45780                              <1>             ; 15/10/2016 - TRDOS 386 (TRDOS v2.0)
45781                              <1>             ; Convert file name
45782                              <1>             ;      from directory entry format
45783                              <1>                ;  to (8.3) dot file name format
45784                              <1>             ;
45785                              <1>             ; TRDOS v1.0 (DIR.ASM, "get_file_name")
45786                              <1>                ; 2005 - 09/10/2011
45787                              <1>             ; INPUT:
45788                              <1>             ;      DS:SI -> Directory Entry Format File Name
45789                              <1>             ;       ES:DI -> DOS Dot File Name Address
45790                              <1>             ; OUTPUT:
45791                              <1>             ;      DS:SI -> DOS Dot File Name Address
45792                              <1>                ;  ES:DI -> Directory Entry Format File Name
45793                              <1>             ;
45794                              <1>             ; TRDOS 386 (15/10/2016)
45795                              <1>             ; INPUT:
45796                              <1>             ;      ESI = File name addr in dir entry format
45797                              <1>             ;      EDI = Dot file name address (destination)
45798                              <1>             ; OUTPUT:
45799                              <1>             ;      File name is converted and moved
45800                              <1>             ;      to destination (as 8.3 dot filename)
45801                              <1>             ;
45802                              <1>             ; Modified registers: EAX, ECX
45803                              <1>
45804                              <1>                ; 2005 (TRDOS 8086) - 2016 (TRDOS 386)
45805                              <1>
45806 0000F70D 57                  <1>             push   edi
45807 0000F70E 56                  <1>             push   esi
45808 0000F70F AC                  <1>             lodsb
45809 0000F710 3C20                <1>             cmp    al, 20h
45810 0000F712 762A                <1>             jna    short pass_gfn_ext
45811 0000F714 56                  <1>             push   esi
45812 0000F715 AA                  <1>             stosb
45813 0000F716 B907000000          <1>             mov    ecx, 7
45814                              <1> loc_gfn_next_char:
45815 0000F71B AC                  <1>             lodsb
45816 0000F71C 3C20                <1>             cmp    al, 20h
45817 0000F71E 7603                <1>             jna    short pass_gfn_fn
45818 0000F720 AA                  <1>             stosb
45819 0000F721 E2F8                <1>             loop   loc_gfn_next_char
45820                              <1> pass_gfn_fn:
45821 0000F723 5E                  <1>             pop    esi
45822 0000F724 83C607              <1>             add    esi, 7
45823 0000F727 AC                  <1>             lodsb
45824 0000F728 3C20                <1>             cmp    al, 20h
45825 0000F72A 7612                <1>             jna    short pass_gfn_ext
45826 0000F72C B42E                <1>             mov    ah, '.'
45827 0000F72E 86E0                <1>             xchg   ah, al
```

```
45828 0000F730 66AB              <1>             stosw
45829 0000F732 AC                <1>             lodsb
45830 0000F733 3C20              <1>             cmp    al, 20h
45831 0000F735 7607              <1>             jna    short pass_gfn_ext
45832 0000F737 AA                <1>             stosb
45833 0000F738 AC                <1>             lodsb
45834 0000F739 3C20              <1>             cmp    al, 20h
45835 0000F73B 7601              <1>             jna    short pass_gfn_ext
45836 0000F73D AA                <1>             stosb
45837                            <1> pass_gfn_ext:
45838 0000F73E 30C0              <1>             xor    al, al
45839 0000F740 AA                <1>             stosb
45840 0000F741 5E                <1>             pop    esi
45841 0000F742 5F                <1>             pop    edi
45842 0000F743 C3                <1>             retn
45843                            <1>
45844                            <1> set_hardware_int_vector:
45845                            <1>             ; 18/03/2017
45846                            <1>             ; 03/03/2017
45847                            <1>             ; 28/02/2017 - TRDOS 386 (TRDOS v2.0)
45848                            <1>             ;
45849                            <1>             ; SET/RESET HARDWARE INTERRUPT GATE
45850                            <1>             ;
45851                            <1>             ; Changes interrupt gate descriptor table
45852                            <1>             ; (without changing default interrupt list)
45853                            <1>             ;
45854                            <1>             ; INPUT:
45855                            <1>             ;     AL = IRQ number (0 to 15)
45856                            <1>             ;     AH > 0 -> set
45857                            <1>             ;     AH = 0 -> reset
45858                            <1>             ;
45859                            <1>             ; Modified registers: eax, ebx, edx, edi
45860                            <1>             ;
45861                            <1>
45862 0000F744 C0E002            <1>             shl    al, 2 ; IRQ number * 4
45863 0000F747 0FB6D8            <1>             movzx  ebx, al
45864                            <1>
45865 0000F74A 08E4              <1>             or     ah, ah
45866 0000F74C 7508              <1>             jnz    short shintv_1 ; set (for user call service)
45867                            <1>
45868                            <1>             ; 18/03/2017
45869 0000F74E 81C3[D0100100]    <1>             add    ebx, IRQ_list ; reset to default interrupt list
45870 0000F754 EB06              <1>             jmp    short shintv_2
45871                            <1> shintv_1:
45872 0000F756 81C3[7DF70000]    <1>             add    ebx, IRQ_u_list
45873                            <1> shintv_2:
45874 0000F75C 8B13              <1>             mov    edx, [ebx] ; IRQ handler address
45875                            <1>
45876                            <1>             ; 03/03/2017
45877 0000F75E D0E0              <1>             shl    al, 1 ; IRQ number * 8
45878                            <1>             ; 18/03/2017
45879 0000F760 0FB6F8            <1>             movzx  edi, al
45880 0000F763 81C7[38500100]    <1>             add    edi, idt + (8*32) ; IRQ 0 offset = idt + 256
45881                            <1>
45882 0000F769 89D0              <1>             mov    eax, edx ; IRQ handler address
45883 0000F76B BB00000800        <1>             mov    ebx, 80000h
45884                            <1>
45885                            <1>             ;mov   edx, eax
45886 0000F770 66BA008E          <1>             mov    dx, 8E00h
45887 0000F774 6689C3            <1>             mov    bx, ax
45888 0000F777 89D8              <1>             mov    eax, ebx ; /* selector = 0x0008 = cs */
45889                            <1>                                     ; /* interrupt gate - dpl=0, present */
45890 0000F779 AB                <1>             stosd  ; selector & offset bits 0-15
45891 0000F77A 8917              <1>             mov    [edi], edx ; attributes & offset bits 16-23
45892                            <1>
45893 0000F77C C3                <1>             retn
45894                            <1> IRQ_u_list:
45895                            <1>             ; 28/02/2017
45896 0000F77D [8B060000]        <1>             dd     timer_int
45897 0000F781 [FF0D0000]        <1>             dd     kb_int
45898 0000F785 [6D080000]        <1>             dd     irq2
45899 0000F789 [BDF70000]        <1>             dd     IRQ_service3
45900 0000F78D [C7F70000]        <1>             dd     IRQ_service4
45901 0000F791 [D1F70000]        <1>             dd     IRQ_service5
45902 0000F795 [B0410000]        <1>             dd     fdc_int
45903 0000F799 [DBF70000]        <1>             dd     IRQ_service7
45904 0000F79D [F6070000]        <1>             dd     rtc_int
45905 0000F7A1 [E5F70000]        <1>             dd     IRQ_service9
45906 0000F7A5 [EFF70000]        <1>             dd     IRQ_service10
45907 0000F7A9 [F9F70000]        <1>             dd     IRQ_service11
45908 0000F7AD [03F80000]        <1>             dd     IRQ_service12
45909 0000F7B1 [0DF80000]        <1>             dd     IRQ_service13
45910 0000F7B5 [2D4B0000]        <1>             dd     hdc1_int
45911 0000F7B9 [544B0000]        <1>             dd     hdc2_int
45912                            <1>
45913                            <1>             ; 03/03/2017
45914                            <1>             ; 27/02/2017
45915                            <1> IRQ_service3:
45916 0000F7BD 36C605[A2650100]03 <1>             mov    byte [ss:IRQnum], 3
45917 0000F7C5 EB4E              <1>             jmp    short IRQ_service
45918                            <1> IRQ_service4:
45919 0000F7C7 36C605[A2650100]04 <1>             mov    byte [ss:IRQnum], 4
45920 0000F7CF EB44              <1>             jmp    short IRQ_service
45921                            <1> IRQ_service5:
45922 0000F7D1 36C605[A2650100]05 <1>             mov    byte [ss:IRQnum], 5
45923 0000F7D9 EB3A              <1>             jmp    short IRQ_service
45924                            <1> IRQ_service7:
45925 0000F7DB 36C605[A2650100]07 <1>             mov    byte [ss:IRQnum], 7
45926 0000F7E3 EB30              <1>             jmp    short IRQ_service
45927                            <1> IRQ_service9:
45928 0000F7E5 36C605[A2650100]09 <1>             mov    byte [ss:IRQnum], 9
45929 0000F7ED EB26              <1>             jmp    short IRQ_service
45930                            <1> IRQ_service10:
```

```
45931 0000F7EF 36C605[A2650100]0A  <1>             mov    byte [ss:IRQnum], 10
45932 0000F7F7 EB1C                <1>             jmp    short IRQ_service
45933                              <1> IRQ_service11:
45934 0000F7F9 36C605[A2650100]0B  <1>             mov    byte [ss:IRQnum], 11
45935 0000F801 EB12                <1>             jmp    short IRQ_service
45936                              <1> IRQ_service12:
45937 0000F803 36C605[A2650100]0C  <1>             mov    byte [ss:IRQnum], 12
45938 0000F80B EB08                <1>             jmp    short IRQ_service
45939                              <1> IRQ_service13:
45940 0000F80D 36C605[A2650100]0D  <1>             mov    byte [ss:IRQnum], 13
45941                              <1>             ;jmp   short IRQ_service
45942                              <1> IRQ_service:
45943                              <1>             ; 13/06/2017
45944                              <1>             ; 11/06/2017
45945                              <1>             ; 10/06/2017
45946                              <1>             ; 01/03/2017, 04/03/2017
45947                              <1>             ; 27/02/2017, 28/02/2017
45948 0000F815 1E                  <1>             push   ds
45949 0000F816 06                  <1>             push   es
45950 0000F817 0FA0                <1>             push   fs
45951 0000F819 0FA8                <1>             push   gs
45952                              <1>
45953 0000F81B 60                  <1>             pushad ; eax,ecx,edx,ebx,esp,ebp,esi,edi
45954 0000F81C 66B91000            <1>             mov     cx, KDATA
45955 0000F820 8ED9                <1>             mov     ds, cx
45956 0000F822 8EC1                <1>             mov     es, cx
45957 0000F824 8EE1                <1>             mov     fs, cx
45958 0000F826 8EE9                <1>             mov     gs, cx
45959                              <1>
45960 0000F828 0F20D8              <1>             mov    eax, cr3
45961 0000F82B A3[9E650100]        <1>             mov    [IRQ_cr3], eax
45962                              <1>
45963 0000F830 A1[20520100]        <1>             mov    eax, [k_page_dir]
45964 0000F835 0F22D8              <1>             mov    cr3, eax
45965                              <1>
45966 0000F838 A0[A2650100]        <1>             mov    al, [IRQnum]
45967                              <1>
45968                              <1>             ;mov   cl, [sysflg]
45969                              <1>             ;mov   [u.r_mode], cl  ; system (0) or user mode (FFh)
45970                              <1> IRQsrv_0:
45971 0000F83D 0FB6D8              <1>             movzx  ebx, al
45972 0000F840 8A9B[08100100]      <1>             mov    bl, [ebx+IRQenum] ; IRQ (available) index number + 1
45973                              <1>             ; 01/03/2017
45974 0000F846 FECB                <1>             dec    bl  ; IRQ index number, 0 to 8
45975 0000F848 0F8807010000        <1>             js     IRQsrv_5 ; not available to use here!?
45976                              <1>             ;
45977 0000F84E 80BB[68650100]80    <1>             cmp    byte [ebx+IRQ.method], 80h ; using by a dev or kernel?
45978 0000F855 7205                <1>             jb     short IRQsrv_1 ; no
45979                              <1>
45980                              <1>             ; If the IRQ service is already owned by TRDOS 386 kernel
45981                              <1>             ;      or a Device driver
45982                              <1>             ; we need to call 'dev_IRQ_service'
45983                              <1>
45984                              <1>             ; IRQ number in AL
45985 0000F857 E868020000          <1>             call   dev_IRQ_service     ; IRQ service for device drivers
45986                              <1>             ; IRQ number in AL
45987                              <1> IRQsrv_1:
45988                              <1>             ; check user callback service status
45989                              <1>             ; AL = IRQ number
45990                              <1>             ; EBX = IRQ (Available) Index number
45991                              <1>
45992 0000F85C A2[D7030300]        <1>             mov    [u.irqwait], al ; set waiting IRQ flag
45993                              <1>
45994 0000F861 8A83[56650100]      <1>             mov    al, [ebx+IRQ.owner]
45995 0000F867 20C0                <1>             and    al, al
45996 0000F869 0F84E6000000        <1>             jz     IRQsrv_5 ; it is not owned by a user/proc
45997                              <1>
45998                              <1>             ; 03/03/2017
45999 0000F86F 89DA                <1>             mov    edx, ebx
46000 0000F871 C0E202              <1>             shl    dl, 2
46001 0000F874 8B92[7A650100]      <1>             mov    edx, [edx+IRQ.addr] ; S.R.B. or Callback service addr
46002                              <1>
46003 0000F87A 8AA3[68650100]      <1>             mov    ah, [ebx+IRQ.method]
46004 0000F880 F6C401              <1>             test   ah, 1
46005 0000F883 7534                <1>             jnz    short IRQsrv_4 ; Callback service method
46006                              <1>
46007                              <1>             ; Signal Response Byte method
46008                              <1>             ;mov   edx, [edx+IRQ.addr] ; Signal Response Byte address
46009                              <1>             ;                     ; (Physical address, non-swappable)
46010 0000F885 80E402              <1>             and    ah, 2 ; bit 1, (S.R.B.) counter (auto increment) method
46011 0000F888 8AA3[71650100]      <1>             mov    ah, [ebx+IRQ.srb] ; Signal Response Byte value
46012 0000F88E 7408                <1>             jz     short IRQsrv_2 ; fixed S.R.B. value
46013                              <1>             ; counter method (auto increment)
46014 0000F890 FEC4                <1>             inc    ah
46015 0000F892 88A3[71650100]      <1>             mov    [ebx+IRQ.srb], ah ; next (count) number
46016                              <1> IRQsrv_2:
46017 0000F898 8822                <1>             mov    [edx], ah ; put S.R.B. val to the user's S.R.B. addr
46018 0000F89A C605[D7030300]00    <1>             mov    byte [u.irqwait], 0 ; clear waiting IRQ flag
46019                              <1>
46020 0000F8A1 3A05[B3030300]      <1>             cmp    al, [u.uno]
46021 0000F8A7 0F84A8000000        <1>             je     IRQsrv_5 ; the owner is current user/process
46022                              <1> IRQsrv_3:
46023                              <1>             ; the owner is not current user/process
46024                              <1>             ; AL = process number
46025 0000F8AD B202                <1>             mov    dl, 2 ; priority, 2 = event (high)
46026 0000F8AF E837FAFFFF          <1>             call   set_run_sequence
46027                              <1>
46028                              <1>             ; [u.irqwait] = waiting IRQ number for callback service
46029                              <1>
46030 0000F8B4 E99C000000          <1>             jmp    IRQsrv_5
46031                              <1> IRQsrv_4:
46032 0000F8B9 3A05[B3030300]      <1>             cmp    al, [u.uno]  ; is the owner is current user/process?
46033 0000F8BF 75EC                <1>             jne    short IRQsrv_3 ; no !
```

```
46034                                  <1>
46035                                  <1>             ; Check if an IRQ callback service already in progress
46036 0000F8C1 803D[D8030300]00        <1>             cmp    byte [u.r_lock], 0
46037 0000F8C8 0F8787000000            <1>             ja     IRQsrv_5 ; nothing to do !
46038                                  <1>                             ; (we need to complete prev callback)
46039 0000F8CE 803D[D4030300]00        <1>             cmp    byte [u.t_lock], 0
46040 0000F8D5 777E                    <1>             ja     short IRQsrv_5 ; nothing to do !
46041                                  <1>                             ; (we need to complete timer callback)
46042                                  <1>
46043                                  <1>             ; 04/03/2017
46044 0000F8D7 C605[D7030300]00        <1>             mov    byte [u.irqwait], 0 ; reset/clear waiting IRQ flag
46045                                  <1>
46046 0000F8DE FE05[D8030300]          <1>             inc    byte [u.r_lock] ; 'IRQ callback service in progress' flag
46047                                  <1>
46048 0000F8E4 8A0D[5B030300]          <1>             mov    cl, [sysflg]   ; (system call) mode flag (kernel/user)
46049 0000F8EA 880D[D9030300]          <1>             mov    [u.r_mode], cl ; system mode (0) or user mode (FFh)
46050                                  <1>
46051                                  <1>             ;
46052 0000F8F0 8B2D[BC510100]          <1>             mov    ebp, [tss.esp0] ; kernel stack address (for ring 0)
46053 0000F8F6 83ED14                  <1>             sub    ebp, 20             ; eip, cs, eflags, esp, ss
46054 0000F8F9 892D[5C030300]          <1>             mov    [u.sp], ebp
46055 0000F8FF 8925[60030300]          <1>             mov    [u.usp], esp
46056                                  <1>
46057                                  <1>             ;or    word [ebp+8], 200h ; 22/01/2017, force enabling interrupts
46058                                  <1>
46059 0000F905 8B44241C                <1>             mov    eax, [esp+28] ; pushed eax
46060 0000F909 A3[64030300]            <1>             mov    [u.r0], eax
46061                                  <1>
46062 0000F90E E83FEEFFFF              <1>             call   wswap ; save user's registers & status
46063                                  <1>
46064                                  <1>             ; software int is in ring 0 but IRQ handler must return to ring 3
46065                                  <1>             ; so, ring 3 return address and stack registers
46066                                  <1>             ; (eip, cs, eflags, esp, ss)
46067                                  <1>             ; must be copied to IRQ handler return
46068                                  <1>             ; eip will be replaced by callback service routine address
46069                                  <1>
46070 0000F913 C605[5B030300]FF        <1>             mov    byte [sysflg], 0FFh ; user mode
46071                                  <1>
46072                                  <1>             ; system mode (system call)
46073                                  <1>             ;mov   ebp, [u.sp] ; EIP (u), CS (UCODE), EFLAGS (u),
46074                                  <1>                                ; ESP (u), SS (UDATA)
46075                                  <1>
46076 0000F91A 8B4510                  <1>             mov    eax, [ebp+16]; SS (UDATA)
46077 0000F91D 89E6                    <1>             mov    esi, esp
46078 0000F91F 50                      <1>             push   eax
46079 0000F920 50                      <1>             push   eax
46080 0000F921 89E7                    <1>             mov    edi, esp
46081 0000F923 893D[60030300]          <1>             mov    [u.usp], edi
46082 0000F929 B908000000              <1>             mov    ecx, ((ESPACE/4) - 4) ; except DS, ES, FS, GS
46083 0000F92E F3A5                    <1>             rep    movsd
46084 0000F930 B104                    <1>             mov    cl, 4
46085 0000F932 F3AB                    <1>             rep    stosd
46086 0000F934 893D[5C030300]          <1>             mov    [u.sp], edi
46087 0000F93A 89EE                    <1>             mov    esi, ebp
46088 0000F93C B105                    <1>             mov    cl, 5 ; EIP (u), CS (UCODE), EFLAGS (u), ESP (u), SS (UDATA)
46089 0000F93E F3A5                    <1>             rep    movsd
46090                                  <1>             ;
46091                                  <1>
46092 0000F940 8B0D[B8030300]          <1>             mov    ecx, [u.pgdir]
46093 0000F946 890D[9E650100]          <1>             mov    [IRQ_cr3], ecx
46094                                  <1>
46095                                  <1> set_IRQ_callback_addr:
46096                                  <1>             ;
46097                                  <1>             ; This routine sets return address
46098                                  <1>             ; to start of user's interrupt
46099                                  <1>             ; service (callback) address
46100                                  <1>             ;
46101                                  <1>             ; INPUT:
46102                                  <1>             ;     EDX = callback routine/service address
46103                                  <1>             ;           (virtual, not physical address!)
46104                                  <1>             ;     [u.sp] = kernel stack, points to
46105                                  <1>             ;              user's EIP,CS,EFLAGS,ESP,SS
46106                                  <1>             ;              registers.
46107                                  <1>             ; OUTPUT:
46108                                  <1>             ;     EIP (user) = callback (service) address
46109                                  <1>             ;     CS (user) = UCODE
46110                                  <1>             ;     EFLAGS (user) = flags before callback
46111                                  <1>             ;      ESP (user) = ESP-4 (user, before callback)
46112                                  <1>             ;     [ESP](user) = EIP (user) before callback
46113                                  <1>             ;
46114                                  <1>             ; Note: If CPU was in user mode while entering
46115                                  <1>             ;     the timer interrupt service routine,
46116                                  <1>             ;     'IRET' will get return to callback routine
46117                                  <1>             ;     immediately. If CPU was in system/kernel mode
46118                                  <1>             ;     'iret' will get return to system call and
46119                                  <1>             ;     then, callback routine will be return address
46120                                  <1>             ;     from system call. (User's callback/service code
46121                                  <1>             ;     will be able to return to normal return address
46122                                  <1>             ;     via a 'sysrele' system call at the end.)
46123                                  <1>             ;
46124                                  <1>             ; Note: User's IRQ callback service code must be ended
46125                                  <1>             ;     with a 'sysrele' system call !
46126                                  <1>             ;
46127                                  <1>             ;     For example:
46128                                  <1>             ;
46129                                  <1>             ;     audio_IRQ_callback:
46130                                  <1>             ;         ...
46131                                  <1>             ;         <load DMA buffer with audio data>
46132                                  <1>             ;         ...
46133                                  <1>             ;         mov eax, 39 ; 'sysrele'
46134                                  <1>             ;         int 40h ; TRDOS 386 system call (interrupt)
46135                                  <1>             ;
46136                                  <1>
```

```
46137                                   <1>             ;mov   edx, [edx+IRQ.addr] ; Callback service address
46138                                   <1>             ;                        ; (Virtual address)
46139                                   <1>
46140 0000F94C 8B2D[5C030300]           <1>             mov    ebp, [u.sp]; kernel's stack, points to EIP (user)
46141 0000F952 895500                   <1>             mov    [ebp], edx
46142                                   <1> IRQsrv_5:
46143                                   <1>             ; EOI & return
46144                                   <1>             ; 11/06/2017
46145                                   <1>             ; 10/06/2017
46146 0000F955 A0[A2650100]             <1>             mov    al, [IRQnum]
46147 0000F95A FA                       <1>             cli
46148 0000F95B 3C07                     <1>             cmp    al, 7
46149 0000F95D 7604                     <1>             jna    short IRQsrv_6
46150                                   <1>             ;
46151                                   <1>             ;mov   al, EOI      ; end of interrupt
46152 0000F95F B020                     <1>             mov    al, 20h
46153                                   <1>             ;cli         ; disable interrupts till stack cleared
46154                                   <1>             ;out   INTB00, al ; For controll2 #2
46155 0000F961 E6A0                     <1>             out    0A0h, al
46156                                   <1> IRQsrv_6:
46157                                   <1>             ;mov   byte [IRQnum], 0 ; reset
46158                                   <1>             ;mov   al, EOI      ; end of interrupt
46159 0000F963 B020                     <1>             mov    al, 20h
46160                                   <1>             ;cli         ; disable interrupts till stack cleared
46161                                   <1>             ;out   INTA00, al ; end of interrupt to 8259 - 1
46162 0000F965 E620                     <1>             out    20h, al
46163                                   <1> IRQsrv_7:
46164                                   <1>             ;; 13/06/2017
46165                                   <1>             ;or    word [ebp+8], 200h ; force enabling interrupts
46166                                   <1>             ;
46167 0000F967 8B0D[9E650100]           <1>             mov    ecx, [IRQ_cr3]      ; previous content of cr3 register
46168 0000F96D 0F22D9                   <1>             mov    cr3, ecx     ; restore cr3 register content
46169                                   <1>             ;
46170 0000F970 61                       <1>             popad ; edi,esi,ebp,(icrement esp by 4), ebx,edx,ecx,eax
46171                                   <1>             ;
46172 0000F971 0FA9                     <1>             pop    gs
46173 0000F973 0FA1                     <1>             pop    fs
46174 0000F975 07                       <1>             pop    es
46175 0000F976 1F                       <1>             pop    ds
46176                                   <1>             ;
46177 0000F977 CF                       <1>             iretd  ; return from interrupt
46178                                   <1>
46179                                   <1> get_device_number:
46180                                   <1>             ; 08/10/2016
46181                                   <1>             ; 07/10/2016 - TRDOS 386 (TRDOS v2.0)
46182                                   <1>             ;
46183                                   <1>             ; This procedure compares name of requested
46184                                   <1>             ; device with kernel device names and
46185                                   <1>             ; installable device names. If names match,
46186                                   <1>             ; the relevant device index (entry) number
46187                                   <1>             ; will be returned the caller (sysopen)
46188                                   <1>             ; for the requested device.
46189                                   <1>             ;
46190                                   <1>             ; NOTE: Installable device drivers must
46191                                   <1>             ; be loaded before using 'sysopen'
46192                                   <1>             ; (opendev) system call.
46193                                   <1>             ;
46194                                   <1>             ; INPUT:
46195                                   <1>             ;    ESI = device name address (ASCIIZ)
46196                                   <1>             ;         (in kernel's memory space)
46197                                   <1>             ;    max name length = 8 without '/dev/')
46198                                   <1>             ;    Device name will be capitalized
46199                                   <1>             ;    and if there is, '/dev/' will be
46200                                   <1>             ;    removed from name before comparising)
46201                                   <1>             ;
46202                                   <1>             ; OUTPUT:
46203                                   <1>             ;    cf = 0 ->
46204                                   <1>             ;      EAX (AL) = device entry/index number
46205                                   <1>             ;    cf = 1 -> device not found (installed)
46206                                   <1>             ;             or invalid device name
46207                                   <1>             ;             (AL=0)
46208                                   <1>             ;    device_name = device name address (asciiz)
46209                                   <1>             ;      ;
46210                                   <1>             ; Modified registers: EAX, EBX, ESI, EDI
46211                                   <1>
46212 0000F978 BF[E15F0100]             <1>             mov    edi, device_name
46213 0000F97D E805010000               <1>             call   lodsb_capitalize
46214 0000F982 88C4                     <1>             mov    ah, al
46215 0000F984 3C2F                     <1>             cmp    al, '/'
46216 0000F986 750E                     <1>             jne    short gdn_1
46217 0000F988 BF[E15F0100]             <1>             mov    edi, device_name
46218 0000F98D E8F5000000               <1>             call   lodsb_capitalize
46219                                   <1> gdn_0:
46220 0000F992 20C0                     <1>             and    al, al ; 0 ?
46221 0000F994 7420                     <1>             jz     short gdn_err ; null name after '/'
46222                                   <1> gdn_1:
46223 0000F996 3C44                     <1>             cmp    al, 'D'
46224 0000F998 7517                     <1>             jne    short gdn_2
46225 0000F99A E8E8000000               <1>             call   lodsb_capitalize
46226 0000F99F 3C45                     <1>             cmp    al, 'E'
46227 0000F9A1 750E                     <1>             jne    short gdn_2
46228 0000F9A3 E8DF000000               <1>             call   lodsb_capitalize
46229 0000F9A8 3C56                     <1>             cmp    al, 'V'
46230 0000F9AA 7505                     <1>             jne    short gdn_2
46231 0000F9AC AC                       <1>             lodsb
46232 0000F9AD 3C2F                     <1>             cmp    al, '/'
46233 0000F9AF 740D                     <1>             je     short gdn_4
46234                                   <1> gdn_2:
46235 0000F9B1 80FC2F                   <1>             cmp    ah, '/'
46236 0000F9B4 750F                     <1>             jne    short gdn_5
46237                                   <1> gdn_err:
46238                                   <1>             ; invalid device name or device not found
46239 0000F9B6 31C0                     <1>             xor    eax, eax ; 0
```

```
46240 0000F9B8 F9                <1>             stc
46241 0000F9B9 C3                <1>             retn
46242                            <1> gdn_3:
46243 0000F9BA 3C2F              <1>             cmp    al, '/'
46244 0000F9BC 7507              <1>             jne    short gdn_5
46245                            <1> gdn_4:
46246 0000F9BE BF[E15F0100]      <1>             mov    edi, device_name
46247 0000F9C3 EB04              <1>             jmp    short gdn_6
46248                            <1> gdn_5:
46249 0000F9C5 3C00              <1>             cmp    al, 0
46250 0000F9C7 7419              <1>             je     short gdn_7
46251                            <1> gdn_6:
46252 0000F9C9 E8B9000000        <1>             call lodsb_capitalize
46253 0000F9CE 81FF[E95F0100]    <1>             cmp    edi, device_name + 8
46254 0000F9D4 72E4              <1>             jb     short gdn_3
46255 0000F9D6 3C00              <1>             cmp    al, 0
46256 0000F9D8 75DC              <1>             jne    short gdn_err
46257 0000F9DA 81FF[E25F0100]    <1>             cmp    edi, device_name + 1
46258 0000F9E0 76D4              <1>             jna    short gdn_err ; null name after '/'
46259                            <1> gdn_7:
46260 0000F9E2 AA                <1>             stosb
46261                            <1>             ; zero padding ("NAME",0,0,0,0)
46262 0000F9E3 81FF[E95F0100]    <1>             cmp    edi, device_name + 8
46263 0000F9E9 72F7              <1>             jb     short gdn_7
46264                            <1> gdn_8:
46265                            <1>             ; search for kernel device names
46266 0000F9EB BE[E15F0100]      <1>             mov    esi, device_name
46267 0000F9F0 BF[EE0D0100]      <1>             mov    edi, KDEV_NAME
46268 0000F9F5 31C0              <1>             xor    eax, eax
46269                            <1> gdn_9:
46270 0000F9F7 A7                <1>             cmpsd
46271 0000F9F8 7505              <1>             jne    short gdn_10
46272 0000F9FA A7                <1>             cmpsd
46273 0000F9FB 7503              <1>             jne    short gdn_11
46274 0000F9FD EB2B              <1>             jmp    short gdn_17 ; match
46275                            <1> gdn_10:
46276 0000F9FF A7                <1>             cmpsd  ; add esi, 4 & add edi, 4
46277                            <1> gdn_11:
46278 0000FA00 BE[E15F0100]      <1>             mov    esi, device_name
46279 0000FA05 FEC0              <1>             inc    al
46280 0000FA07 3C16              <1>             cmp    al, NumOfKernelDevNames
46281 0000FA09 72EC              <1>             jb     short gdn_9
46282                            <1> gdn_12:
46283                            <1>             ; search for installable device names
46284                            <1>             ; esi = offset device_name
46285 0000FA0B BF[0C600100]      <1>             mov    edi, IDEV_NAME
46286 0000FA10 28C0              <1>             sub    al, al ; 0
46287                            <1> gdn_13:
46288 0000FA12 A7                <1>             cmpsd
46289 0000FA13 7505              <1>             jne    short gdn_14
46290 0000FA15 A7                <1>             cmpsd
46291 0000FA16 7503              <1>             jne    short gdn_15
46292 0000FA18 EB3F              <1>             jmp    short gdn_19 ; match
46293                            <1> gdn_14:
46294 0000FA1A A7                <1>             cmpsd  ; add esi, 4 & add edi, 4
46295                            <1> gdn_15:
46296 0000FA1B BE[E15F0100]      <1>             mov    esi, device_name
46297 0000FA20 FEC0              <1>             inc    al
46298 0000FA22 3C08              <1>             cmp    al, NumOfInstallableDevices
46299 0000FA24 72EC              <1>             jb     short gdn_13
46300                            <1>
46301                            <1> gdn_16:     ; error: invalid device name (not found) !
46302 0000FA26 30C0              <1>             xor    al, al
46303 0000FA28 F9                <1>             stc
46304 0000FA29 C3                <1>             retn
46305                            <1>
46306                            <1> gdn_17:             ; name match (with one of kernel device names)
46307                            <1>             ;
46308                            <1>             ; convert KDEV_NAME index to
46309                            <1>             ; KDEV_NUMBER index
46310                            <1>             ; (different names are used for same devices)
46311                            <1>             ; (example: "COM1" & "TTY8" = device number 18)
46312 0000FA2A 89C3              <1>             mov    ebx, eax ; < 256
46313 0000FA2C 8A83[9E0E0100]    <1>             mov    al, [KDEV_NUMBER+ebx]
46314                            <1>
46315                            <1>             ; check if empty dev entry in the list
46316 0000FA32 80B8[90610100]00  <1>             cmp    byte [DEV_OPENMODE+eax], 0
46317 0000FA39 771B              <1>             ja     short gdn_18 ; it must be already set
46318                            <1>
46319                            <1>             ; (re)set device name and access flags
46320                            <1>             ; (remain open work will be easy after that)
46321                            <1>             ; (NOTE: here, data will be copied to bss section)
46322 0000FA3B 88C3              <1>             mov    bl, al
46323 0000FA3D 83EF08            <1>             sub    edi, 8 ; kernel device name address (data)
46324 0000FA40 66C1E302          <1>             shl    bx, 2
46325 0000FA44 89BB[AE610100]    <1>             mov    [DEV_NAME_PTR+ebx], edi ; (all) device names
46326 0000FA4A 8A98[F40F0100]    <1>             mov    bl, [KDEV_ACCESS+eax] ; kernel dev list (data)
46327 0000FA50 8898[DC600100]    <1>             mov    [DEV_ACCESS+eax], bl ; (all) device list (bss)
46328                            <1> gdn_18:
46329 0000FA56 FEC0              <1>             inc    al ; 1 to NumOfKernelDevNames (<=7Fh)
46330                            <1>             ; eax = device index/entry number
46331 0000FA58 C3                <1>             retn
46332                            <1>
46333                            <1> gdn_19:             ; name match (with one of installable device names)
46334                            <1>             ;
46335                            <1>             ; al = 0 to NumOfInstallableDevices - 1 (<=7Fh)
46336                            <1>
46337 0000FA59 89C3              <1>             mov    ebx, eax
46338 0000FA5B 80C316            <1>             add    bl, NumOfKernelDevices    ; < NUMOFDEVICES
46339                            <1>
46340                            <1>             ; check if empty dev entry in the list
46341 0000FA5E 80BB[90610100]00  <1>             cmp    byte [DEV_OPENMODE+ebx], 0
46342 0000FA65 771D              <1>             ja     short gdn_20 ; it must be already set
```

```
46343                                  <1>
46344                                  <1>             ; (re)set device name and access flags
46345                                  <1>             ; (remain open work will be easy after that)
46346 0000FA67 83EF08                  <1>             sub    edi, 8 ; installable device name address
46347 0000FA6A 66C1E302                <1>             shl    bx, 2 ;*4
46348 0000FA6E 89BB[AE610100]          <1>             mov    [DEV_NAME_PTR+ebx], edi ; (all) device names
46349 0000FA74 66C1EB02                <1>             shr    bx, 2
46350 0000FA78 8A80[54600100]          <1>             mov    al, [IDEV_FLAGS+eax] ; installable dev list
46351 0000FA7E 8883[DC600100]          <1>             mov    [DEV_ACCESS+ebx], al ; (all) device list
46352                                  <1> gdn_20:
46353 0000FA84 88D8                    <1>             mov    al, bl
46354                                  <1>             ; eax = device index/entry number ; < NUMOFDEVICES
46355 0000FA86 C3                      <1>             retn
46356                                  <1>
46357                                  <1> lodsb_capitalize:
46358                                  <1>       ; 07/10/2016 - TRDOS 386 (TRDOS v2.0)
46359                                  <1>       ; INPUT -> [esi] = character
46360                                  <1>       ;          edi = destination
46361                                  <1>       ; OUTPUT -> AL contains capitalized character
46362                                  <1>       ;        esi = esi+1
46363                                  <1>       ;        edi = edi+1
46364                                  <1>       ;
46365 0000FA87 AC                      <1>       lodsb
46366 0000FA88 3C61                    <1>       cmp al, 61h
46367 0000FA8A 7206                    <1>       jb  short lodsb_cap_retn
46368 0000FA8C 3C7A                    <1>       cmp al, 7Ah
46369 0000FA8E 7702                    <1>       ja  short lodsb_cap_retn
46370 0000FA90 24DF                    <1>       and al, 0DFh
46371                                  <1> lodsb_cap_retn:
46372 0000FA92 AA                      <1>       stosb
46373 0000FA93 C3                      <1>       retn
46374                                  <1>
46375                                  <1> device_open:
46376                                  <1>       ; 08/10/2016 - TRDOS 386 (TRDOS v2.0)
46377                                  <1>       ; Complete device opening work for sysopen (device)
46378                                  <1>       ;
46379                                  <1>       ; INPUT ->
46380                                  <1>       ;     EAX = Device Number (AL)
46381                                  <1>       ;       CL = Open mode (1 = read, 2 = write)
46382                                  <1>       ;      CH = Device access byte (bit 0 = 0)
46383                                  <1>       ; OUTPUT ->
46384                                  <1>       ;     EAX = Device Number
46385                                  <1>       ;     CF = 0 -> device has been opened
46386                                  <1>       ;     CF = 1 -> device could not be opened
46387                                  <1>       ;
46388                                  <1>       ;  Modified registers: ebx, (edx, ecx, esi, edi, ebp)
46389                                  <1>       ;
46390                                  <1>
46391 0000FA94 89C3                    <1>       mov    ebx, eax
46392 0000FA96 66C1E302                <1>       shl    bx, 2 ; *4
46393                                  <1>
46394 0000FA9A F6C580                  <1>       test   ch, 80h ; bit 7, installable device driver flag
46395 0000FA9D 7406                    <1>       jz     short d_open_2 ; Kernel device
46396                                  <1>       ; installable device
46397                                  <1> d_open_1:
46398 0000FA9F FFA3[58600100]          <1>         jmp  dword [ebx+IDEV_OADDR-4]
46399                                  <1> d_open_2:
46400 0000FAA5 FFA3[B00E0100]          <1>       jmp    dword [ebx+KDEV_OADDR-4]
46401                                  <1>
46402                                  <1> device_close:
46403                                  <1>       ; 08/10/2016 - TRDOS 386 (TRDOS v2.0)
46404                                  <1>       ; Complete device closing work for sysclose (device)
46405                                  <1>       ;
46406                                  <1>       ; INPUT ->
46407                                  <1>       ;     EAX = Device Number (AL)
46408                                  <1>       ;       CL = Open mode (1 = read, 2 = write)
46409                                  <1>       ;      CH = Device access byte (bit 0 = 0)
46410                                  <1>       ; OUTPUT ->
46411                                  <1>       ;     EAX = Device Number
46412                                  <1>       ;     CF = 0 -> device has been closed
46413                                  <1>       ;     CF = 1 -> device could not be closed
46414                                  <1>       ;
46415                                  <1>       ; Modified registers: ebx, (edx, ecx, esi, edi, ebp)
46416                                  <1>       ;
46417                                  <1>
46418 0000FAAB 89C3                    <1>       mov    ebx, eax
46419 0000FAAD 66C1E302                <1>       shl    bx, 2 ; *4
46420                                  <1>
46421 0000FAB1 F6C580                  <1>       test   ch, 80h ; bit 7, installable device driver flag
46422 0000FAB4 7406                    <1>       jz     short d_close_2 ; Kernel device
46423                                  <1>       ; installable device
46424                                  <1> d_close_1:
46425 0000FAB6 FFA3[78600100]          <1>         jmp  dword [ebx+IDEV_CADDR-4]
46426                                  <1> d_close_2:
46427 0000FABC FFA3[000F0100]          <1>       jmp    dword [ebx+KDEV_CADDR-4]
46428                                  <1>
46429                                  <1> rnull:
46430                                  <1>       ; 07/10/2016 - TRDOS 386 (TRDOS v2.0)
46431                                  <1>       ; read null (read from null device)
46432 0000FAC2 C3                      <1>       retn
46433                                  <1>
46434                                  <1> wnull:
46435                                  <1>       ; 07/10/2016 - TRDOS 386 (TRDOS v2.0)
46436                                  <1>       ; write null (write to null device)
46437 0000FAC3 C3                      <1>       retn
46438                                  <1>
46439                                  <1> dev_IRQ_service:
46440                                  <1>       ; 12/05/2017
46441                                  <1>       ; 13/04/2017
46442                                  <1>       ; 27/02/2017 - TRDOS 386 (TRDOS v2.0)
46443                                  <1>       ; INPUT ->
46444                                  <1>       ;     AL = IRQ Number (0 to 15)
46445                                  <1>       ;
```

```
46446 0000FAC4 53                <1>       push   ebx
46447 0000FAC5 0FB6D8            <1>       movzx  ebx, al
46448 0000FAC8 C0E302            <1>       shl    bl, 2 ; * 4
46449 0000FACB 8B9B[16650100]    <1>       mov    ebx, [ebx+DEV_INT_HNDLR]
46450 0000FAD1 21DB              <1>       and    ebx, ebx
46451 0000FAD3 7404              <1>         jz   short dIRQ_s_retn
46452 0000FAD5 50                <1>       push   eax
46453                            <1>
46454 0000FAD6 FFD3              <1>       call   ebx
46455                            <1>
46456 0000FAD8 58                <1>       pop    eax
46457                            <1> dIRQ_s_retn:
46458 0000FAD9 5B                <1>       pop    ebx
46459 0000FADA C3                <1>       retn
46460                            <1>
46461                            <1>
46462                            <1> set_dev_IRQ_service:
46463                            <1>       ; 13/04/2017 - TRDOS 386 (TRDOS v2.0)
46464                            <1>       ;
46465                            <1>       ; Set Device Interrupt Service
46466                            <1>       ;
46467                            <1>       ; INPUT ->
46468                            <1>       ;     AL = IRQ Number
46469                            <1>       ;     EBX = Hardware Interrupt Service Address
46470                            <1>       ;
46471                            <1>       ; Note: There is not a validation check here
46472                            <1>       ;     because this procedure is called by
46473                            <1>       ;     TRDOS 386 kernel !
46474                            <1>       ;      (Even if a device driver does not exist
46475                            <1>       ;     this setting may be used by sysaudio
46476                            <1>       ;     and other system calls for hardware
46477                            <1>       ;     components which use IRQ method for I/O.)
46478                            <1>       ;
46479                            <1>       ;push  esi
46480 0000FADB 0FB6F0            <1>       movzx  esi, al
46481 0000FADE 66C1E602          <1>       shl    si, 2 ; * 4
46482 0000FAE2 899E[16650100]    <1>       mov    [esi+DEV_INT_HNDLR], ebx
46483                            <1>       ;pop   esi
46484 0000FAE8 C3                <1>       retn
46485                            <1>
46486                            <1>
46487                            <1> sysaudio: ; AUDIO FUNCTIONS
46488                            <1>       ; 10/10/2017
46489                            <1>       ; 22/06/2017
46490                            <1>       ; 28/05/2017, 04/06/2017, 05/06/2017, 10/06/2017
46491                            <1>       ; 01/05/2017, 12/05/2017, 15/05/2017, 20/05/2017
46492                            <1>       ; 21/04/2017, 22/04/2017, 23/04/2017, 24/04/2017
46493                            <1>       ; 10/04/2017, 13/04/2017, 14/04/2017, 16/04/2017
46494                            <1>       ; 03/04/2017 (VIA VT8237R)
46495                            <1>       ; 01/04/2016 (trdosk6.s -> tdosk8.s)
46496                            <1>       ; 16/05/2016 - TRDOS 386 (TRDOS v2.0)
46497                            <1>       ;
46498                            <1>       ; Inputs:
46499                            <1>       ;
46500                            <1>       ;     BH = 0 -> Beep (PC Speaker)
46501                            <1>       ;          BL = Duration Counter (1 for 1/64 second)
46502                            <1>       ;          CX = Frequency Divisor (1193180/Frequency)
46503                            <1>       ;             (1331 for 886 Hz)
46504                            <1>       ;
46505                            <1>       ;     01/04/2017
46506                            <1>       ;
46507                            <1>       ;     BH = 1 -> DETECT (& ENABLE) AUDIO DEVICE
46508                            <1>       ;          BL = 0 : PC SPEAKER
46509                            <1>       ;               1 : SOUND BLASTER 16
46510                            <1>       ;               2 : INTEL AC'97
46511                            <1>       ;               3 : VIA VT8237R (VT8233)
46512                            <1>       ;               4 : INTEL HDA
46513                            <1>       ;            5-FEh : unknown/invalid
46514                            <1>       ;              ; 04/06/2017
46515                            <1>       ;             FFh : Get current audio device id
46516                            <1>       ;
46517                            <1>       ;     BH = 2 -> ALLOCATE AUDIO BUFFER (for user)
46518                            <1>       ;           ECX = Audio Buffer Size (must be equal to
46519                            <1>       ;                 the half of DMA buffer size)
46520                            <1>       ;           EDX = Virtual Address of the buffer
46521                            <1>       ;                 (This is not DMA buffer!)
46522                            <1>       ;
46523                            <1>       ;     BH = 3 -> INITIALIZE AUDIO DEVICE
46524                            <1>       ;          BL = 0,2 -> for Signal Response Byte
46525                            <1>       ;           CL = Signal Response Byte Value (fixed)
46526                            <1>       ;                        if BL = 0
46527                            <1>       ;                 auto increment of S.R.B. value
46528                            <1>       ;                        if BL = 2
46529                            <1>       ;             EDX = Signal Response (Return) Byte Address
46530                            <1>       ;
46531                            <1>       ;          BL = 1 for CallBack Method
46532                            <1>       ;           EDX = CallBack Service Address (Virtual)
46533                            <1>       ;
46534                            <1>       ;          BL > 2 -> invalid function
46535                            <1>       ;
46536                            <1>       ;         (Audio buffer must be allocated before
46537                            <1>       ;          initialization.)
46538                            <1>       ;
46539                            <1>       ;     BH = 4 -> START TO PLAY
46540                            <1>       ;          BL = Mode
46541                            <1>       ;              Bit 0 = mono/stereo (1 = stereo)
46542                            <1>       ;              Bit 1 = 8 bit / 16 bit (1 = 16 bit)
46543                            <1>       ;          CX = Sampling Rate (Hz)
46544                            <1>       ;
46545                            <1>       ;     BH = 5 -> PAUSE
46546                            <1>       ;          BL = Any
46547                            <1>       ;
46548                            <1>       ;     BH = 6 -> CONTINUE TO PLAY
```

```
46549 <1>       ;           BL = Any
46550 <1>       ;
46551 <1>       ;      BH = 7 -> STOP
46552 <1>       ;           BL = Any
46553 <1>       ;
46554 <1>       ;      BH = 8 -> RESET
46555 <1>       ;           BL = Any
46556 <1>       ;
46557 <1>       ;      BH = 9 -> CANCEL (CALLBACK or S.R.B. SERVICE)
46558 <1>       ;           BL = Any
46559 <1>       ;
46560 <1>       ;      BH = 10 -> DEALLOCATE AUDIO BUFFER (for user)
46561 <1>       ;           BL = Any
46562 <1>       ;
46563 <1>       ;      BH = 11 -> SET VOLUME LEVEL
46564 <1>       ;           BL: (Bit 0 to 6)
46565 <1>       ;               0 = Master (Playback, Lineout) volume
46566 <1>       ;           CL = Left Channel Volume
46567 <1>       ;           CH = Right Channel Volume
46568 <1>       ;
46569 <1>       ;           Note: If BL >= 80h (Bit 7 of BL is set),
46570 <1>       ;           volume level will be set for next playing
46571 <1>       ;           (actual volume level will not be changed
46572 <1>       ;           immediately)
46573 <1>       ;
46574 <1>       ;      BH = 12 -> DISABLE AUDIO DEVICE
46575 <1>       ;           (reset audio device and unlink dma buffer)
46576 <1>       ;           BL = Any
46577 <1>       ;
46578 <1>       ;      12/05/2017
46579 <1>       ;      BH = 13 -> MAP DMA BUFFER TO USER
46580 <1>       ;          (for direct access to system's dma buffer)
46581 <1>       ;
46582 <1>       ;           ECX = map size in bytes
46583 <1>       ;               (will be rounded up to page borders)
46584 <1>       ;           EDX = Virtual Address of the buffer
46585 <1>       ;               (Will be rounded up to page borders)
46586 <1>       ;
46587 <1>       ;      05/06/2017
46588 <1>       ;      04/06/2017
46589 <1>       ;      BH = 14 -> GET AUDIO DEVICE INFO
46590 <1>       ;           BL: 0 = Audio Controller Info
46591 <1>       ;             > 0 = Invalid for now!
46592 <1>       ;
46593 <1>       ;      22/06/2017
46594 <1>       ;      BH = 15 -> GET CURRENT SOUND DATA (for graphics)
46595 <1>       ;           BL: 0 -> PCM OUT data
46596 <1>       ;             > 0 -> Invalid for now!
46597 <1>       ;           ECX = 0 -> Get DMA Buffer Pointer
46598 <1>       ;              EDX = Not Used
46599 <1>       ;           ECX > 0 -> Byte count for buffer (EDX)
46600 <1>       ;               EDX = Buffer Address (Virtual)
46601 <1>       ;
46602 <1>       ;      10/10/2017
46603 <1>       ;      BH = 16 -> UPDATE DMA BUFFER DATA
46604 <1>       ;                (by using the Audio Buffer content)
46605 <1>       ;           BL = 0 : Update dma half buffer in sequence
46606 <1>       ;                   (automatic destination)
46607 <1>       ;               1 : Update 1st half of the dma buffer
46608 <1>       ;               2 : Update 2nd half of the dma buffer
46609 <1>       ;               3-FEh: Invalid!
46610 <1>       ;               FFh = Get current flag value
46611 <1>       ;                    (Half buffer number -1)
46612 <1>       ;
46613 <1>       ;
46614 <1>       ; Outputs:
46615 <1>       ;
46616 <1>       ;      For BH = 0 -> Beep
46617 <1>       ;          None
46618 <1>       ;
46619 <1>       ;      01/04/2017
46620 <1>       ;
46621 <1>       ;      For BH = 1 -> DETECT (& ENABLE) AUDIO DEVICE
46622 <1>       ;          AH = 0 : PC SPEAKER
46623 <1>       ;               1 : SOUND BLASTER 16
46624 <1>       ;               2 : INTEL AC'97
46625 <1>       ;               3 : VIA VT8237R (VT8233)
46626 <1>       ;               4 : INTEL HDA
46627 <1>       ;             5-FFh : unknown/invalid
46628 <1>       ;          AL = mode status
46629 <1>       ;               bit 0 = mono /stereo (1 = stereo)
46630 <1>       ;               bit 1 = 8 bit / 16 bit ( 1 = 16 bit)
46631 <1>       ;          04/06/2017
46632 <1>       ;          EBX = PCI DEVICE/VENDOR ID (if >0)
46633 <1>       ;              (BX = VENDOR ID)
46634 <1>       ;          (if CF = 1 -> Error code in EAX)
46635 <1>       ;
46636 <1>       ;      For BH = 2 -> ALLOCATE AUDIO BUFFER (for user)
46637 <1>       ;          EAX = Physical Address of the buffer
46638 <1>       ;          (if CF = 1 -> Error code in EAX)
46639 <1>       ;
46640 <1>       ;      For BH = 3 -> INITIALIZE AUDIO DEVICE
46641 <1>       ;          (if CF = 1 -> Error code in EAX)
46642 <1>       ;
46643 <1>       ;      For BH = 4 -> START TO PLAY
46644 <1>       ;          none (if CF = 1 -> Error code in EAX)
46645 <1>       ;
46646 <1>       ;      For BH = 5 -> PAUSE
46647 <1>       ;          none (if CF = 1 -> Error code in EAX)
46648 <1>       ;
46649 <1>       ;      For BH = 6 -> CONTINUE TO PLAY
46650 <1>       ;          none (if CF = 1 -> Error code in EAX)
46651 <1>       ;
```

```
46652                                 <1>       ;     For BH = 7 -> STOP
46653                                 <1>       ;         none (if CF = 1 -> Error code in EAX)
46654                                 <1>       ;
46655                                 <1>       ;     For BH = 8 -> RESET
46656                                 <1>       ;         none (if CF = 1 -> Error code in EAX)
46657                                 <1>       ;
46658                                 <1>       ;     For BH = 9 -> CANCEL (CALLBACK or S.R.B. SERVICE)
46659                                 <1>       ;         none (if CF = 1 -> Error code in EAX)
46660                                 <1>       ;
46661                                 <1>       ;     For BH = 10 -> DEALLOCATE AUDIO BUFFER (for user)
46662                                 <1>       ;         none (if CF = 1 -> Error code in EAX)
46663                                 <1>       ;
46664                                 <1>       ;     For BH = 11 -> SET VOLUME LEVEL
46665                                 <1>       ;         none (if CF = 1 -> Error code in EAX)
46666                                 <1>       ;
46667                                 <1>       ;     For BH = 12 -> DISABLE AUDIO DEVICE
46668                                 <1>       ;         none (if CF = 1 -> Error code in EAX)
46669                                 <1>       ;
46670                                 <1>       ;     12/05/2017
46671                                 <1>       ;     For BH = 13 -> MAP DMA BUFFER TO USER
46672                                 <1>       ;         EAX = Physical Address of the buffer
46673                                 <1>       ;         (if CF = 1 -> Error code in EAX)
46674                                 <1>       ;
46675                                 <1>       ;     04/06/2017
46676                                 <1>       ;     For BH = 14 -> GET AUDIO DEVICE INFO
46677                                 <1>       ;     (for BL = 0) ; 05/06/2017
46678                                 <1>       ;         EAX = IRQ Number in AL
46679                                 <1>       ;               Audio Device Number in AH
46680                                 <1>       ;         EBX = DEV/VENDOR ID
46681                                 <1>       ;             (DDDDDDDDDDDDDDDDVVVVVVVVVVVVVVVV)
46682                                 <1>       ;         ECX = BUS/DEV/FN
46683                                 <1>       ;             (00000000BBBBBBBBDDDDDFFF00000000)
46684                                 <1>       ;         EDX = NABMBAR/NAMBAR (for AC97)
46685                                 <1>       ;             (Low word, DX = NAMBAR address)
46686                                 <1>       ;         EDX = Base IO Addr (DX) for SB16 & VT8233
46687                                 <1>       ;         (if CF = 1 -> Error code in EAX)
46688                                 <1>       ;                     (ERR_DEV_NOT_RDY = 15)
46689                                 <1>       ;
46690                                 <1>       ;     22/06/2017
46691                                 <1>       ;     For BH = 15 -> GET CURRENT SOUND DATA
46692                                 <1>       ;                    (for graphics)
46693                                 <1>       ;     (for BL = 0)
46694                                 <1>       ;      If ECX input is 0
46695                                 <1>       ;         EAX = DMA Buffer Current Position (Offset)
46696                                 <1>       ;      If ECX input >  0
46697                                 <1>       ;         EAX = Actual transfer count
46698                                 <1>       ;         (Sound samples will be copied from
46699                                 <1>       ;          Current DMA Buffer Position to EDX
46700                                 <1>       ;          virtual address as EAX bytes.)
46701                                 <1>       ;       ((If CF = 1 ->  Error code in EAX))
46702                                 <1>       ;
46703                                 <1>       ;
46704                                 <1>       ;     10/10/2017
46705                                 <1>       ;     For BH = 16 -> UPDATE DMA BUFFER DATA
46706                                 <1>       ;         EAX = 0, if the updated (or current)
46707                                 <1>       ;                 half buffer is DMA half buffer 1
46708                                 <1>       ;         EAX = 1, if the updated (or current)
46709                                 <1>       ;                 half buffer is DMA half buffer 2
46710                                 <1>       ;         (If CF = 1 -> Error code in EAX)
46711                                 <1>       ;
46712                                 <1>
46713 0000FAE9 80FF11                <1>       cmp   bh, AUDIO1L/4
46714 0000FAEC 0F83EBC9FFFF          <1>       jnb   sysret
46715                                 <1>
46716 0000FAF2 C0E702                <1>       shl   bh, 2 ; *4
46717 0000FAF5 0FB6F7                <1>       movzx esi, bh
46718                                 <1>
46719                                 <1>       ; 22/04/2017
46720 0000FAF8 31C0                  <1>       xor   eax, eax
46721 0000FAFA A3[64030300]          <1>       mov   [u.r0], eax  ; 0
46722                                 <1>
46723 0000FAFF FF96[0AFB0000]        <1>       call  dword [esi+AUDIO1]
46724                                 <1>       ;jc   error
46725 0000FB05 E9D3C9FFFF            <1>       jmp   sysret
46726                                 <1>
46727 0000FB0A [A11D0000]            <1> AUDIO1:     dd    beep ; FUNCTION = 0 (bl = Duration Counter
46728                                 <1>                 ;                     cx = Frequency Divisor
46729 0000FB0E [4EFB0000]            <1>       dd    soundc_detect
46730 0000FB12 [EAFB0000]            <1>       dd    sound_alloc
46731 0000FB16 [A1FC0000]            <1>       dd    soundc_init
46732 0000FB1A [59FE0000]            <1>       dd    sound_play
46733 0000FB1E [EFFE0000]            <1>       dd    sound_pause
46734 0000FB22 [19FF0000]            <1>       dd    sound_continue
46735 0000FB26 [43FF0000]            <1>       dd    sound_stop
46736 0000FB2A [6CFF0000]            <1>       dd    soundc_reset
46737 0000FB2E [9DFF0000]            <1>       dd    soundc_cancel
46738 0000FB32 [C3FF0000]            <1>       dd    sound_dalloc
46739 0000FB36 [EEFF0000]            <1>       dd    sound_volume
46740 0000FB3A [40000100]            <1>       dd    soundc_disable
46741 0000FB3E [B2000100]            <1>       dd    sound_dma_map
46742 0000FB42 [21010100]            <1>       dd    soundc_info
46743 0000FB46 [80010100]            <1>       dd    sound_data
46744 0000FB4A [2D020100]            <1>       dd    sound_update
46745                                 <1>
46746                                 <1> AUDIO1L     EQU   $ - AUDIO1
46747                                 <1>
46748                                 <1> soundc_detect:
46749                                 <1>       ; FUNCTION = 1
46750                                 <1>       ; bl = Audio device type number
46751                                 <1>       ; (0= pc speaker, 1 = sound blaster 16, 2 = intel ac97
46752                                 <1>       ;  3= via vt823x, 4 = intel HDA, 0FFh= any)
46753                                 <1>
46754                                 <1>       ; 04/06/2017
```

```
46755 0000FB4E 8A25[A5650100]      <1>       mov    ah, [audio_device]
46756 0000FB54 80FBFF              <1>       cmp    bl, 0FFh ; get current audio device id
46757 0000FB57 7408                <1>       je     short sysaudio0
46758                              <1>
46759 0000FB59 20E4                <1>       and    ah, ah
46760 0000FB5B 741E                <1>       jz     short soundc_get_dev
46761                              <1>
46762 0000FB5D 38DC                <1>       cmp    ah, bl
46763 0000FB5F 7567                <1>       jne    short soundc_dev_err
46764                              <1>
46765                              <1> sysaudio0:
46766 0000FB61 A0[A6650100]        <1>       mov    al, [audio_mode]
46767                              <1> sysaudio1:
46768 0000FB66 A3[64030300]        <1>       mov    [u.r0], eax
46769 0000FB6B 8B1D[B0650100]      <1>       mov    ebx, [audio_vendor] ; (DEVICE/VENDOR ID)
46770 0000FB71 8B2D[60030300]      <1>       mov    ebp, [u.usp]
46771 0000FB77 895D10              <1>       mov    [ebp+16], ebx  ; ebx
46772 0000FB7A C3                  <1>       retn
46773                              <1>
46774                              <1> soundc_get_dev:
46775                              <1>       ; 28/05/2017
46776                              <1>       ; 03/04/2017, 24/04/2017
46777 0000FB7B C605[A4650100]00    <1>       mov    byte [audio_pci], 0
46778 0000FB82 80FB03              <1>       cmp    bl, 3 ; VIA VT8233 (VT8237R) Audio Controller & AC97 Codec
46779                              <1>       ;jne   short soundc_get_dev_sb
46780                              <1>       ; 28/05/2017
46781 0000FB85 7220                <1>       jb     short soundc_get_dev_sb
46782 0000FB87 773F                <1>       ja     short soundc_dev_err  ; temporary (28/05/2017)
46783                              <1>       ;
46784 0000FB89 E83A180000          <1>       call   DetectVT8233
46785 0000FB8E 7238                <1>       jc     short soundc_dev_err
46786                              <1>       ; eax = 0
46787                              <1>
46788                              <1>       ;mov   ebx, [audio_vendor]
46789                              <1>       ; ebx = DEVICE/VENDOR ID
46790                              <1>       ;      DDDDDDDDDDDDDDDDVVVVVVVVVVVVVVVV
46791                              <1>
46792 0000FB90 B003                <1>       mov    al, 3  ; VIA VT8237R (VT3233) Audio Controller
46793 0000FB92 88C4                <1>       mov    ah, al
46794                              <1>
46795                              <1> soundc_get_pci_dev_ok: ; 28/05/2017
46796 0000FB94 FE05[A4650100]      <1>       inc    byte [audio_pci] ; = 1
46797                              <1> soundc_get_dev_ok:
46798                              <1>
46799                              <1> soundc_get_dev_sb16_ok:
46800 0000FB9A A2[A5650100]        <1>       mov    [audio_device], al
46801 0000FB9F 8825[A6650100]      <1>       mov    [audio_mode], ah ; stereo (bit0), 16 bit (bit1) capability
46802 0000FBA5 EBBF                <1>       jmp    short sysaudio1
46803                              <1>
46804                              <1> soundc_get_dev_sb:
46805                              <1>       ; 24/04/2017
46806 0000FBA7 80FB01              <1>       cmp    bl, 1 ; Sound Blaster 16
46807 0000FBAA 750E                <1>       jne    short soundc_get_dev_ich ; 28/05/2017
46808                              <1>       ;
46809 0000FBAC E8451D0000          <1>       call   DetectSB
46810 0000FBB1 7215                <1>       jc     short soundc_dev_err
46811 0000FBB3 B801030000          <1>       mov    eax, 0301h ; Sound Blaster 16
46812 0000FBB8 EBE0                <1>       jmp    short soundc_get_dev_sb16_ok
46813                              <1>
46814                              <1> soundc_get_dev_ich:
46815                              <1>       ; 28/05/2017
46816                              <1>       ;cmp   bl, 2 ; Intel AC'97 Audio Controller (ICH)
46817                              <1>       ;jne   short soundc_dev_err ; Temporary  (28/05/2017)
46818                              <1>       ;                      ; (Here will be modified just after
46819                              <1>       ;                      ; new sound card code will be ready!)
46820 0000FBBA E8FC170000          <1>       call   DetectICH
46821 0000FBBF 7207                <1>       jc     short soundc_dev_err
46822                              <1>       ;
46823 0000FBC1 B802030000          <1>       mov    eax, 0302h ; AC'97 (ICH)
46824 0000FBC6 EBCC                <1>       jmp    short soundc_get_pci_dev_ok
46825                              <1>
46826                              <1> soundc_dev_err:
46827 0000FBC8 B80F000000          <1>       mov    eax, ERR_DEV_NOT_RDY ; Device not ready !
46828 0000FBCD EB0C                <1>       jmp    short sysaudio_err
46829                              <1>
46830                              <1> sound_buff_error:
46831 0000FBCF B82E000000          <1>       mov    eax, ERR_BUFFER ; Buffer error !
46832 0000FBD4 EB05                <1>       jmp    short sysaudio_err
46833                              <1>
46834                              <1> soundc_respond_err:
46835                              <1>       ; ERR_TIME_OUT ; 'time out !' error
46836 0000FBD6 B819000000          <1>       mov    eax, ERR_DEV_NOT_RESP ; 'device not responding !' error
46837                              <1> sysaudio_err:
46838 0000FBDB A3[64030300]        <1>       mov    [u.r0], eax
46839 0000FBE0 A3[C8030300]        <1>       mov    [u.error], eax
46840 0000FBE5 E9D3C8FFFF          <1>       jmp    error
46841                              <1>
46842                              <1> sound_alloc:
46843                              <1>       ; FUNCTION =  2
46844                              <1>       ; ecx = audio buffer size (in bytes)
46845                              <1>       ; edx = audio buffer address (virtual)
46846                              <1>       ; 28/05/2017
46847                              <1>       ; 01/05/2017, 15/05/2017
46848                              <1>       ; 21/04/2017, 24/04/2017
46849 0000FBEA 803D[A4650100]00    <1>       cmp    byte [audio_pci], 0
46850 0000FBF1 7708                <1>       ja     short snd_alloc_0
46851                              <1>       ; Max. 64KB DMA buffer !!!
46852 0000FBF3 81F900800000        <1>       cmp    ecx, 32768
46853 0000FBF9 77D4                <1>       ja     short sound_buff_error
46854                              <1> snd_alloc_0:
46855                              <1>       ; 15/05/2017
46856 0000FBFB 81F900100000        <1>       cmp    ecx, 4096 ; PAGE_SIZE
46857 0000FC01 72CC                <1>       jb     short sound_buff_error
```

```
                                <1>       ;
0000FC03 A1[B8650100]           <1>       mov    eax, [audio_buffer] ; audio buffer address (current)
0000FC08 09C0                   <1>       or     eax, eax
0000FC0A 7445                   <1>       jz     short snd_alloc_2
                                <1>       ; audio buffer exists !
0000FC0C 8A1D[B3030300]         <1>       mov    bl, [u.uno]
0000FC12 3A1D[CD650100]         <1>       cmp    bl, [audio_user]
0000FC18 0F85F5000000           <1>       jne    sndc_owner_error ; not owner !
0000FC1E 39D0                   <1>       cmp    eax, edx ; same virtual buffer address ?
0000FC20 7508                   <1>       jne    short snd_alloc_1
0000FC22 3B0D[C0650100]         <1>       cmp    ecx, [audio_buff_size]
0000FC28 746C                   <1>       je     short snd_alloc_3 ; Nothing to do !
                                <1>                                ; Buffer has been set already!
                                <1> snd_alloc_1:
0000FC2A 51                     <1>       push   ecx
0000FC2B 52                     <1>       push   edx
0000FC2C 89C3                   <1>       mov    ebx, eax ; audio buffer address (current)
0000FC2E 8B0D[C0650100]         <1>       mov    ecx, [audio_buff_size]
0000FC34 E8465BFFFF             <1>       call   deallocate_user_pages
0000FC39 5A                     <1>       pop    edx
0000FC3A 59                     <1>       pop    ecx
0000FC3B 31C0                   <1>       xor    eax, eax ; 0
0000FC3D A3[B8650100]           <1>       mov    [audio_buffer], eax  ; 0
0000FC42 A3[BC650100]           <1>       mov    [audio_p_buffer], eax  ; 0
0000FC47 A3[C0650100]           <1>       mov    [audio_buff_size], eax
0000FC4C A2[CD650100]           <1>       mov    [audio_user], al ; 0
                                <1> snd_alloc_2:
0000FC51 89D3                   <1>       mov    ebx, edx
                                <1>       ; 01/05/2017
0000FC53 BA00F0FFFF             <1>       mov    edx, ~PAGE_OFF ; truncating page offsets
                                <1>                             ; for aligning to page borders
                                <1>       ;and   eax, edx
0000FC58 21D3                   <1>       and    ebx, edx
0000FC5A 21D1                   <1>       and    ecx, edx
                                <1>       ; 15/05/2017
                                <1>       ; EAX = Beginning address (physical)
                                <1>       ; EAX = 0 -> Allocate mem block from the 1st proper aperture
                                <1>       ; ECX = Number of bytes to be allocated
0000FC5C E8C357FFFF             <1>       call   allocate_memory_block
0000FC61 0F8268FFFFFF           <1>       jc     sound_buff_error
                                <1>       ; EAX = Physical address of the allocated memory block
                                <1>       ; ECX = Allocated bytes (as truncated to page border)
                                <1>       ; EBX = Virtual address (as truncated to page border)
0000FC67 50                     <1>       push   eax
0000FC68 53                     <1>       push   ebx
0000FC69 51                     <1>       push   ecx
0000FC6A E8055CFFFF             <1>       call   allocate_user_pages
0000FC6F 59                     <1>       pop    ecx
0000FC70 5B                     <1>       pop    ebx
0000FC71 58                     <1>       pop    eax
0000FC72 7223                   <1>       jc     short snd_alloc_4  ; insufficient memory, buff error
                                <1>       ; eax = physical address of the user's audio buffer
                                <1>       ; ebx = virtual address of the user's audio buffer
                                <1>       ; ecx = buffer size (in bytes)
0000FC74 A3[BC650100]           <1>       mov    [audio_p_buffer], eax
0000FC79 891D[B8650100]         <1>       mov    [audio_buffer], ebx
0000FC7F 890D[C0650100]         <1>       mov    [audio_buff_size], ecx
0000FC85 8A15[B3030300]         <1>       mov    dl, [u.uno]
0000FC8B 8815[CD650100]         <1>       mov    [audio_user], dl
0000FC91 A3[64030300]           <1>       mov    [u.r0], eax
                                <1> snd_alloc_3:
0000FC96 C3                     <1>       retn
                                <1> snd_alloc_4:
                                <1>       ; 15/05/2017
                                <1>       ; EAX = Beginning address (physical)
                                <1>       ; ECX = Number of bytes to be deallocated
0000FC97 E89559FFFF             <1>       call   deallocate_memory_block
0000FC9C E92EFFFFFF             <1>       jmp    sound_buff_error  ; insufficient memory, buff error
                                <1>
                                <1> soundc_init:
                                <1>       ; FUNCTION = 3
                                <1>       ; bl = method (0= s.r.b., 1= callback, 2= auto incr s.r.b.)
                                <1>       ; cl = signal response byte (initial or fixed) value
                                <1>       ; edx = signal response byte or callback address
                                <1>       ; 28/05/2017
                                <1>       ; 12/05/2017, 20/05/2017
                                <1>       ; 22/04/2017, 23/04/2017, 24/04/2017
                                <1>       ; 13/04/2017, 14/04/2017, 16/04/2017, 21/04/2017
                                <1>       ; 03/04/2017, 10/04/2017
                                <1>
0000FCA1 A0[A5650100]           <1>       mov    al, [audio_device]
0000FCA6 20C0                   <1>       and    al, al
0000FCA8 7549                   <1>       jnz    short sndc_init_6
                                <1>       ;
0000FCAA C605[A4650100]00       <1>       mov    byte [audio_pci], 0
0000FCB1 52                     <1>       push   edx
0000FCB2 53                     <1>       push   ebx
0000FCB3 51                     <1>       push   ecx
0000FCB4 E83D1C0000             <1>       call   DetectSB
0000FCB9 7213                   <1>       jc     short sndc_init_8
0000FCBB 66B80103               <1>       mov    ax, 0301h ; Sound Blaster 16
0000FCBF EB1E                   <1>       jmp    short sndc_init_7
                                <1>
                                <1> sndc_init_11:
                                <1>       ; 28/05/2017
0000FCC1 E8F5160000             <1>       call   DetectICH ; Detect AC'97 (ICH) Audio Controller
0000FCC6 7217                   <1>       jc     short sndc_init_7
0000FCC8 66B80203               <1>       mov    ax, 0302h ; Intel AC'97 Audio Device
0000FCCC EB0B                   <1>       jmp    short sndc_init_12 ; (PCI device)
                                <1>
                                <1> sndc_init_8:
0000FCCE E8F5160000             <1>       call   DetectVT8233
                                <1>       ;jc    short sndc_init_7
```

```
46961 0000FCD3 72EC               <1>       jc     sndc_init_11 ; 28/05/2017
46962                             <1>       ; eax = 0
46963 0000FCD5 B003               <1>       mov    al, 3  ; VIA VT8237R (VT3233) Audio Controller
46964 0000FCD7 88C4               <1>       mov    ah, al
46965                             <1>
46966                             <1> sndc_init_12:
46967 0000FCD9 FE05[A4650100]     <1>       inc    byte [audio_pci] ; = 1
46968                             <1> sndc_init_7:
46969 0000FCDF 59                 <1>       pop    ecx
46970 0000FCE0 5B                 <1>       pop    ebx
46971 0000FCE1 5A                 <1>       pop    edx
46972 0000FCE2 0F82E0FEFFFF       <1>       jc     soundc_dev_err
46973                             <1>       ;
46974 0000FCE8 A2[A5650100]       <1>       mov    [audio_device], al
46975 0000FCED 8825[A6650100]     <1>       mov    [audio_mode], ah ; stereo (bit0), 16 bit (bit1) capability
46976                             <1>
46977                             <1> sndc_init_6:
46978 0000FCF3 833D[B8650100]00   <1>       cmp    dword [audio_buffer], 0
46979 0000FCFA 0F86CFFEFFFF       <1>       jna    sound_buff_error
46980                             <1>
46981 0000FD00 A0[B3030300]       <1>       mov    al, [u.uno]
46982 0000FD05 8A25[CD650100]     <1>       mov    ah, [audio_user]
46983 0000FD0B 08E4               <1>       or     ah, ah
46984 0000FD0D 7418               <1>       jz     short sndc_init0
46985 0000FD0F 38E0               <1>       cmp    al, ah
46986 0000FD11 7419               <1>       je     short sndc_init1
46987                             <1>
46988                             <1> sndc_owner_error:
46989 0000FD13 B80B000000         <1>       mov    eax, ERR_NOT_OWNER ; 'permission denied !' error
46990                             <1> sndc_perm_error:
46991 0000FD18 A3[64030300]       <1>       mov    [u.r0], eax
46992 0000FD1D A3[C8030300]       <1>       mov    [u.error], eax
46993 0000FD22 E996C7FFFF         <1>       jmp    error
46994                             <1> sndc_init0:
46995 0000FD27 A2[CD650100]       <1>       mov    [audio_user], al
46996                             <1> sndc_init1:
46997 0000FD2C 8915[D0650100]     <1>       mov    [audio_cb_addr], edx
46998 0000FD32 881D[CE650100]     <1>       mov    [audio_cb_mode], bl
46999 0000FD38 880D[CF650100]     <1>       mov    [audio_srb], cl
47000                             <1>
47001                             <1>       ; 24/04/2017
47002 0000FD3E 803D[A5650100]03   <1>       cmp    byte [audio_device], 3 ; VT8233 (VT8237R)
47003 0000FD45 7438               <1>       je     short sndc_init_9
47004                             <1>       ;ja    short soundc_respond_err ; temporary (28/05/2017)
47005 0000FD47 803D[A5650100]01   <1>       cmp    byte [audio_device], 1 ; SB 16
47006 0000FD4E 7510               <1>       jne    short sndc_init_13
47007 0000FD50 BB[1B1B0100]       <1>       mov    ebx, sb16_int_handler
47008                             <1>       ; Note: 'SbInit' is at 'Start to Play' stage
47009                             <1>       ; 20/05/2017
47010 0000FD55 66C705[DA650100]08- <1>      mov    word [audio_master_volume], 0808h ; 2/8
47011 0000FD5D 08                 <1>
47012 0000FD5E EB3F               <1>       jmp    short sndc_init_10
47013                             <1> sndc_init_13:
47014                             <1>       ; 28/05/2017
47015 0000FD60 803D[A5650100]02   <1>       cmp    byte [audio_device], 2 ; AC 97 (ICH)
47016 0000FD67 0F8569FEFFFF       <1>       jne    soundc_respond_err ; temporary (28/05/2017)
47017                             <1>
47018 0000FD6D E8FE1E0000         <1>       call   ac97_codec_config
47019 0000FD72 0F825EFEFFFF       <1>       jc     soundc_respond_err ; codec error !
47020                             <1>
47021 0000FD78 BB[571E0100]       <1>       mov    ebx, ac97_int_handler
47022 0000FD7D EB20               <1>       jmp    short sndc_init_10
47023                             <1>
47024                             <1> sndc_init_9:
47025                             <1>       ;call  reset_codec
47026                             <1>       ;; eax = 1
47027                             <1>       ;call  codec_io_w16 ; w32
47028 0000FD7F E8BB170000         <1>       call   init_codec ; 28/05/2017
47029 0000FD84 0F824CFEFFFF       <1>       jc     soundc_respond_err ; codec error !
47030                             <1>
47031 0000FD8A E8EC190000         <1>       call   channel_reset
47032                             <1>
47033                             <1>       ; setup the Codec (actually mixer registers)
47034 0000FD8F E8F6180000         <1>         call    codec_config  ; unmute codec, set rates.
47035 0000FD94 0F823CFEFFFF       <1>       jc     soundc_respond_err ; codec error !
47036                             <1>
47037 0000FD9A BB[F7160100]       <1>       mov    ebx, vt8233_int_handler
47038                             <1> sndc_init_10:
47039                             <1>       ; 13/04/2017
47040 0000FD9F A0[A7650100]       <1>       mov    al, [audio_intr] ; IRQ number
47041 0000FDA4 E832FDFFFF         <1>       call   set_dev_IRQ_service
47042                             <1>
47043                             <1>       ; SETUP (audio) INTERRUPT CALLBACK SERVICE
47044 0000FDA9 8A1D[A7650100]     <1>       mov    bl, [audio_intr] ; IRQ number
47045 0000FDAF 8A3D[CE650100]     <1>       mov    bh, [audio_cb_mode]
47046 0000FDB5 FEC7               <1>       inc    bh  ; 1 = Signal Response Byte method (fixed value)
47047                             <1>                   ; 2 = Callback service method
47048                             <1>                   ; 3 = Auto Increment S.R.B. method
47049 0000FDB7 8A0D[CF650100]     <1>       mov    cl, [audio_srb]
47050 0000FDBD 8B15[D0650100]     <1>       mov    edx, [audio_cb_addr]
47051 0000FDC3 A0[CD650100]       <1>       mov    al, [audio_user]
47052                             <1>       ; 14/04/2017
47053 0000FDC8 E8DB040000         <1>       call   set_irq_callback_service
47054                             <1>       ; 16/04/2017
47055 0000FDCD A3[64030300]       <1>       mov    [u.r0], eax
47056                             <1>       ;jnc   sysret
47057 0000FDD2 7316               <1>       jnc    short sndc_init2 ; 21/04/2017
47058                             <1>       ;
47059 0000FDD4 A3[C8030300]       <1>       mov    dword [u.error], eax
47060                             <1>
47061 0000FDD9 A0[A7650100]       <1>       mov    al, [audio_intr] ; IRQ number
47062 0000FDDE 31DB               <1>       xor    ebx, ebx ; reset IRQ handler address
47063 0000FDE0 E8F6FCFFFF         <1>       call   set_dev_IRQ_service
```

```
47064                                  <1>
47065 0000FDE5 E9D3C6FFFF              <1>       jmp    error
47066                                  <1>
47067                                  <1> sndc_init2:
47068                                  <1>       ; 21/04/2017
47069 0000FDEA 8B0D[C0650100]          <1>       mov    ecx, [audio_buff_size] ; audio buffer size
47070 0000FDF0 D1E1                    <1>       shl    ecx, 1 ; *2
47071 0000FDF2 A1[C4650100]            <1>       mov    eax, [audio_dma_buff]
47072 0000FDF7 21C0                    <1>       and    eax, eax
47073 0000FDF9 7415                    <1>       jz     short sndc_init3
47074                                  <1>
47075 0000FDFB 8B15[C8650100]          <1>       mov    edx, [audio_dmabuff_size] ; dma buffer size
47076 0000FE01 39D1                    <1>       cmp    ecx, edx
47077 0000FE03 744D                    <1>       je     short sndc_init5
47078                                  <1>
47079 0000FE05 87CA                    <1>       xchg   ecx, edx
47080 0000FE07 E82558FFFF              <1>       call   deallocate_memory_block
47081 0000FE0C 87D1                    <1>       xchg   edx, ecx
47082 0000FE0E 31C0                    <1>       xor    eax, eax
47083                                  <1> sndc_init3:
47084                                  <1>       ; 12/05/2017
47085 0000FE10 803D[A5650100]01        <1>       cmp    byte [audio_device], 1 ; SB 16
47086 0000FE17 7515                    <1>       jne    short sndc_init4
47087 0000FE19 C705[C4650100]-         <1>       mov    dword [audio_dma_buff], sb16_dma_buffer
47088 0000FE1F [00000200]              <1>
47089 0000FE23 C705[C8650100]0000-     <1>       mov    dword [audio_dmabuff_size], 65536
47090 0000FE2B 0100                    <1>
47091                                  <1>       ;xor   eax, eax
47092                                  <1>       ;mov   [u.r0], eax ; 0 = no error, successful
47093 0000FE2D C3                      <1>       retn
47094                                  <1>
47095                                  <1> sndc_init4:
47096                                  <1>       ; EAX = Beginning address (physical)
47097                                  <1>       ; EAX = 0 -> Allocate mem block from the 1st proper aperture
47098                                  <1>       ; ECX = Number of bytes to be allocated (>0)
47099 0000FE2E E8F155FFFF              <1>       call   allocate_memory_block
47100 0000FE33 0F8296FDFFFF            <1>       jc     sound_buff_error
47101                                  <1>
47102                                  <1>       ; set dma buffer address and size parameters
47103 0000FE39 A3[C4650100]            <1>       mov    [audio_dma_buff], eax ; dma buffer address
47104 0000FE3E 890D[C8650100]          <1>       mov    [audio_dmabuff_size], ecx ; dma buffer size
47105                                  <1> ;     ; EAX = Beginning (physical) addr of the allocated mem block
47106                                  <1> ;     ; ECX = Num of allocated bytes (rounded up to page borders)
47107                                  <1> ;     cmp    byte [audio_pci], 0 ; AC97 audio controller ?
47108                                  <1> ;     ja     short sndc_init4
47109                                  <1> ;
47110                                  <1> ;     ; Sound Blaster 16 uses classic DMA
47111                                  <1> ;     mov    edx, eax
47112                                  <1> ;     add    edx, ecx
47113                                  <1> ;     cmp    edx, 1000000h ; 1st 16 MB
47114                                  <1> ;     jna    short sndc_init4
47115                                  <1> ;
47116                                  <1> ;     ; error !
47117                                  <1> ;     ; restore Memory Allocation Table Content
47118                                  <1> ;     ; EAX = Beginning address (physical)
47119                                  <1> ;     ; ECX = Number of bytes to be deallocated
47120                                  <1> ;     call   deallocate_memory_block
47121                                  <1> ;     ; reset dma buffer address and size parameters
47122                                  <1> ;     xor    eax, eax ; 0
47123                                  <1> ;     mov    [audio_dma_buff], eax ; 0
47124                                  <1> ;     mov    [audio_dmabuff_size], ecx ; 0
47125                                  <1> ;     jmp    sound_buff_error
47126                                  <1> ;
47127                                  <1> ;sndc_init4:
47128 0000FE44 803D[A5650100]03        <1>       cmp    byte [audio_device], 3
47129                                  <1>       ;jne   short sndc_init5
47130 0000FE4B 7506                    <1>       jne    short sndc_init14 ; 28/05/2017
47131 0000FE4D E86A190000              <1>       call   set_vt8233_bdl
47132                                  <1> sndc_init5:
47133                                  <1>       ;sub   eax, eax ; 0
47134                                  <1>       ;mov   [u.r0], eax ; 0 = no error, successful
47135 0000FE52 C3                      <1>       retn
47136                                  <1> sndc_init14:
47137 0000FE53 E8311F0000              <1>       call   set_ac97_bdl
47138                                  <1>       ;jmp   short sndc_init5
47139 0000FE58 C3                      <1>       retn
47140                                  <1>
47141                                  <1> sound_play:
47142                                  <1>       ; FUNCTION = 4
47143                                  <1>       ; bl = Mode
47144                                  <1>       ;      bit 0 = mono/stereo (1 = stereo)
47145                                  <1>       ;      bit 1 = 8 bit / 16 bit (1 = 16 bit)
47146                                  <1>       ; cx = Sampling Rate (Hz)
47147                                  <1>
47148                                  <1>       ; 13/06/2017
47149                                  <1>       ; Note: Even if Mode bits are not 11b,
47150                                  <1>       ;      AC'97 Audio Controller (&Codec)
47151                                  <1>       ;      will play audio samples as 16 bit, stereo
47152                                  <1>       ;      samples.
47153                                  <1>       ;      (Program must fill the audio buffer
47154                                  <1>       ;      as required; 8 bit samples must be converted
47155                                  <1>       ;      to 16 bit samples and mono samples must be
47156                                  <1>       ;      converted to stereo samples...)
47157                                  <1>       ;
47158                                  <1>       ; 28/05/2017
47159                                  <1>       ; 15/05/2017, 20/05/2017
47160                                  <1>       ; 21/04/2017, 24/04/2017
47161                                  <1>       ; ... device check at first
47162 0000FE59 A0[A5650100]            <1>       mov    al, [audio_device]
47163 0000FE5E 08C0                    <1>       or     al, al ; 0 ; pc speaker or invalid
47164 0000FE60 0F84351FFFFF            <1>       jz     beeper_gfx ; 'video.s' ; temporary !
47165                                  <1> ;     cmp    al, 3 ; VIA VT 8237R (vt8233)
47166                                  <1> ;     je     short snd_play_1
```

```
47167                                  <1> ;     cmp   al, 1 ; SB 16
47168                                  <1> ;     jne   soundc_dev_err ; temporary !
47169                                  <1> ;snd_play_0:
47170                                  <1>       ; ... buffer & (buffer) owner check at second
47171 0000FE66 833D[B8650100]00        <1>       cmp   dword [audio_buffer], 0
47172 0000FE6D 0F865CFDFFFF            <1>       jna   sound_buff_error
47173 0000FE73 A0[B3030300]            <1>       mov   al, [u.uno]
47174 0000FE78 3A05[CD650100]          <1>       cmp   al, [audio_user]
47175 0000FE7E 0F858FFEFFFF            <1>       jne   sndc_owner_error
47176                                  <1>
47177 0000FE84 66890D[D6650100]        <1>       mov   [audio_freq], cx ; sample frequency (Hertz)
47178 0000FE8B 88D8                    <1>       mov   al, bl
47179 0000FE8D 2401                    <1>       and   al, 1 ; mono/stereo (1= stereo)
47180 0000FE8F FEC0                    <1>       inc   al ; channels
47181 0000FE91 A2[D5650100]            <1>       mov   [audio_stmo], al ; sound channels (1 or 2)
47182 0000FE96 B008                    <1>       mov   al, 8
47183 0000FE98 F6C302                  <1>       test  bl, 2 ; bits per sample (1= 16 bit)
47184 0000FE9B 7402                    <1>       jz    short snd_play_bps
47185 0000FE9D D0E0                    <1>       shl   al, 1
47186                                  <1> snd_play_bps:
47187 0000FE9F A2[D4650100]            <1>       mov   [audio_bps], al
47188                                  <1>       ; Transfer ring 3 (user's) audio buffer content to dma buffer
47189 0000FEA4 8B3D[C4650100]          <1>       mov   edi, [audio_dma_buff] ; dma buffer (ring 0)
47190 0000FEAA 09FF                    <1>       or    edi, edi
47191 0000FEAC 0F841DFDFFFF            <1>       jz    sound_buff_error
47192 0000FEB2 8B35[BC650100]          <1>       mov   esi, [audio_p_buffer] ; physical address (ring 3)
47193 0000FEB8 8B0D[C0650100]          <1>       mov   ecx, [audio_buff_size] ; 15/05/2017
47194                                  <1>       ;rep  movsb
47195 0000FEBE C1E902                  <1>       shr   ecx, 2
47196 0000FEC1 F3A5                    <1>       rep   movsd
47197                                  <1>       ; 20/05/2017
47198 0000FEC3 C605[CC650100]01        <1>       mov   byte [audio_flag], 1 ; next half (on next time)
47199                                  <1>
47200                                  <1>       ; 24/04/2017
47201 0000FECA A0[A5650100]            <1>       mov   al, [audio_device]
47202 0000FECF 3C03                    <1>       cmp   al, 3 ; VT8233 (VT8237R)
47203 0000FED1 7410                    <1>       je    short snd_play_1
47204 0000FED3 3C01                    <1>       cmp   al, 1 ; Sound Blaster 16
47205 0000FED5 7512                    <1>       jne   short snd_play_2  ; 28/05/2017
47206 0000FED7 E8E81A0000              <1>       call  SbInit_play
47207 0000FEDC 0F82F4FCFFFF            <1>       jc    soundc_respond_err
47208 0000FEE2 C3                      <1>       retn
47209                                  <1>
47210                                  <1> snd_play_1:
47211 0000FEE3 E804190000              <1>       call  vt8233_start_play
47212 0000FEE8 C3                      <1>       retn
47213                                  <1>
47214                                  <1> snd_play_2:
47215                                  <1>       ; 28/05/2017
47216                                  <1>       ;cmp  al, 2 ; AC'97
47217                                  <1>       ;jne  short snd_play_3
47218                                  <1>
47219 0000FEE9 E8CF1E0000              <1>       call  ac97_start_play
47220 0000FEEE C3                      <1>       retn
47221                                  <1>
47222                                  <1> ;snd_play_3:
47223                                  <1> ;     ;call hda_start_play
47224                                  <1> ;     retn
47225                                  <1>
47226                                  <1> sound_pause:
47227                                  <1>       ; FUNCTION = 5
47228                                  <1>       ; Pause
47229                                  <1>       ; 28/05/2017
47230                                  <1>       ; 24/04/2017
47231                                  <1>       ; 22/04/2017
47232 0000FEEF E814030000              <1>       call  snd_dev_check
47233 0000FEF4 7275                    <1>       jc    short snd_nothing ; temporary.
47234 0000FEF6 E81A030000              <1>       call  snd_buf_check
47235 0000FEFB 726E                    <1>       jc    short snd_nothing ; temporary.
47236 0000FEFD A0[A5650100]            <1>       mov   al, [audio_device]
47237 0000FF02 3C03                    <1>       cmp   al, 3 ; VIA VT 8237R (vt8233)
47238 0000FF04 7409                    <1>       je    short snd_pause_1
47239 0000FF06 3C01                    <1>       cmp   al, 1 ; Sound Blaster 16
47240 0000FF08 750A                    <1>       jne   short snd_pause_2 ; 28/05/2017
47241 0000FF0A E9931C0000              <1>       jmp   sb16_pause
47242                                  <1> snd_pause_1:
47243 0000FF0F E996190000              <1>       jmp   vt8233_pause
47244                                  <1> snd_pause_2:
47245                                  <1>       ; 28/05/2017
47246                                  <1>       ;cmp  al, 2 ; AC'97
47247                                  <1>       ;jne  short snd_nothing ; temporary.
47248 0000FF14 E932200000              <1>       jmp   ac97_pause
47249                                  <1>
47250                                  <1> sound_continue:
47251                                  <1>       ; FUNCTION = 6
47252                                  <1>       ; Continue to play
47253                                  <1>       ; 28/05/2017
47254                                  <1>       ; 22/04/2017
47255 0000FF19 E8EA020000              <1>       call  snd_dev_check
47256 0000FF1E 724B                    <1>       jc    short snd_nothing ; temporary.
47257 0000FF20 E8F0020000              <1>       call  snd_buf_check
47258 0000FF25 7244                    <1>       jc    short snd_nothing ; temporary.
47259 0000FF27 A0[A5650100]            <1>       mov   al, [audio_device]
47260 0000FF2C 3C03                    <1>       cmp   al, 3 ; VIA VT 8237R (vt8233)
47261 0000FF2E 7409                    <1>       je    short snd_cont_1
47262 0000FF30 3C01                    <1>       cmp   al, 1 ; Sound Blaster 16
47263 0000FF32 750A                    <1>       jne   short snd_cont_2 ; 28/05/2017
47264 0000FF34 E98C1C0000              <1>       jmp   sb16_continue
47265                                  <1> snd_cont_1:
47266 0000FF39 E919190000              <1>       jmp   vt8233_play
47267                                  <1> snd_cont_2:
47268                                  <1>       ; 28/05/2017
47269                                  <1>       ;cmp  al, 2 ; AC'97
```

```
47270                                   <1>       ;jne   short snd_nothing ; temporary.
47271 0000FF3E E9D01E0000               <1>       jmp    ac97_play
47272                                   <1>
47273                                   <1> sound_stop:
47274                                   <1>       ; FUNCTION = 7
47275                                   <1>       ; Stop playing
47276                                   <1>       ; 28/05/2017
47277                                   <1>       ; 24/05/2017
47278                                   <1>       ; 21/04/2017, 22/04/2017, 24/04/2017
47279 0000FF43 E8C0020000               <1>       call   snd_dev_check
47280 0000FF48 7221                     <1>       jc     short snd_nothing ; temporary.
47281                                   <1>       ;call  snd_buf_check
47282 0000FF4A E8CF020000               <1>       call   snd_user_check ; 24/05/2017
47283 0000FF4F 721A                     <1>       jc     short snd_nothing ; temporary.
47284                                   <1>
47285 0000FF51 A0[A5650100]             <1>       mov    al, [audio_device]
47286 0000FF56 3C03                     <1>       cmp    al, 3 ; VIA VT 8237R (vt8233)
47287 0000FF58 0F8455180000             <1>       je     vt8233_stop
47288                                   <1>       ; 28/05/2017
47289                                   <1>       ;ja    short snd_nothing
47290 0000FF5E 3C01                     <1>       cmp    al, 1 ; Sound Blaster 16
47291 0000FF60 0F84821C0000             <1>       je     sb16_stop
47292                                   <1>       ;cmp   al, 2
47293                                   <1>       ;je    short ac97_stop
47294 0000FF66 E9B21F0000               <1>       jmp    ac97_stop ; temporary.
47295                                   <1>       ;jmp   hda_stop
47296                                   <1>
47297                                   <1> snd_nothing:
47298                                   <1>       ; 21/04/2017
47299 0000FF6B C3                       <1>       retn
47300                                   <1>
47301                                   <1> soundc_reset:
47302                                   <1>       ; FUNCTION = 8
47303                                   <1>       ; Reset Audio Controller
47304                                   <1>       ; 28/05/2017
47305                                   <1>       ; 22/04/2017
47306 0000FF6C E897020000               <1>       call   snd_dev_check
47307 0000FF71 72F8                     <1>       jc     snd_nothing ; temporary.
47308 0000FF73 E89D020000               <1>       call   snd_buf_check
47309 0000FF78 72F1                     <1>       jc     snd_nothing ; temporary.
47310                                   <1>
47311 0000FF7A A0[A5650100]             <1>       mov    al, [audio_device]
47312 0000FF7F 3C03                     <1>       cmp    al, 3 ; VIA VT 8237R (vt8233)
47313 0000FF81 0F8431190000             <1>       je     vt8233_reset
47314 0000FF87 77E2                     <1>       ja     short snd_nothing ; temporary.
47315                                   <1>       ;ja    hda_reset
47316 0000FF89 3C01                     <1>       cmp    al, 1 ; Sound Blaster 16
47317 0000FF8B 0F850B200000             <1>       jne    ac97_reset
47318 0000FF91 E8A41C0000               <1>       call   sb16_reset
47319 0000FF96 0F823AFCFFFF             <1>       jc     soundc_respond_err
47320 0000FF9C C3                       <1>       retn
47321                                   <1>
47322                                   <1> soundc_cancel:
47323                                   <1>       ; FUNCTION = 9
47324                                   <1>       ; Cancel audio callback service
47325                                   <1>       ; 22/04/2017
47326 0000FF9D A0[CD650100]             <1>       mov    al, [audio_user]
47327 0000FFA2 3A05[B3030300]           <1>       cmp    al, [u.uno]
47328 0000FFA8 75C1                     <1>       jne    short snd_nothing
47329                                   <1>       ; RESET (audio) INTERRUPT CALLBACK SERVICE
47330 0000FFAA 8A1D[A7650100]           <1>       mov    bl, [audio_intr] ; IRQ number
47331 0000FFB0 A0[B3030300]             <1>       mov    al, [u.uno]
47332 0000FFB5 28FF                     <1>       sub    bh, bh ; 0 ; unlink IRQ from user service
47333 0000FFB7 E8EC020000               <1>       call   set_irq_callback_service
47334 0000FFBC 0F8256FDFFFF             <1>       jc     sndc_perm_error ; 'permission denied' error
47335 0000FFC2 C3                       <1>       retn
47336                                   <1>
47337                                   <1> sound_dalloc:
47338                                   <1>       ; FUNCTION = 10
47339                                   <1>       ; Deallocate (ring 3) audio buffer
47340                                   <1>       ; 22/04/2017
47341 0000FFC3 A0[CD650100]             <1>       mov    al, [audio_user]
47342 0000FFC8 3A05[B3030300]           <1>       cmp    al, [u.uno]
47343 0000FFCE 759B                     <1>       jne    short snd_nothing
47344 0000FFD0 8B1D[B8650100]           <1>       mov    ebx, [audio_buffer]
47345                                   <1>       ;or    ebx, ebx
47346                                   <1>       ;jz    short snd_nothing
47347 0000FFD6 8B0D[C0650100]           <1>       mov    ecx, [audio_buff_size]
47348 0000FFDC E89E57FFFF               <1>       call   deallocate_user_pages
47349 0000FFE1 31C0                     <1>       xor    eax, eax
47350 0000FFE3 A3[B8650100]             <1>       mov    [audio_buffer], eax ; 0
47351 0000FFE8 A2[CD650100]             <1>       mov    [audio_user], al ; 0
47352 0000FFED C3                       <1>       retn
47353                                   <1>
47354                                   <1> sound_volume:
47355                                   <1>       ; FUNCTION = 11
47356                                   <1>       ; Set sound volume level
47357                                   <1>       ; 28/05/2017
47358                                   <1>       ; 20/05/2017
47359                                   <1>       ; 22/04/2017, 24/04/2017
47360                                   <1>       ; bl = component (0 = master/playback/lineout volume)
47361                                   <1>       ; cl = left channel volume level (0 to 31)
47362                                   <1>       ; ch = right channel volume level (0 to 31)
47363                                   <1>
47364 0000FFEE 80FB80                   <1>       cmp    bl, 80h
47365 0000FFF1 720E                     <1>       jb     short snd_vol_1
47366 0000FFF3 0F8772FFFFFF             <1>       ja     snd_nothing ; temporary.
47367                                   <1>       ; Set volume level for next play (BL>= 80h)
47368 0000FFF9 66890D[DA650100]         <1>       mov    [audio_master_volume], cx
47369 00010000 C3                       <1>       retn
47370                                   <1> snd_vol_1:
47371                                   <1>       ; set volume level immediate (BL< 80h)
47372 00010001 80FB00                   <1>       cmp    bl, 0
```

```
47373 00010004 0F8761FFFFFF        <1>       ja    snd_nothing ; temporary.
47374                              <1>
47375 0001000A E8F9010000          <1>       call  snd_dev_check
47376 0001000F 0F8256FFFFFF        <1>       jc    snd_nothing ; temporary.
47377 00010015 E8FB010000          <1>       call  snd_buf_check
47378 0001001A 0F824BFFFFFF        <1>       jc    snd_nothing ; temporary.
47379                              <1>
47380 00010020 A0[A5650100]        <1>       mov   al, [audio_device]
47381 00010025 3C03                <1>       cmp   al, 3 ; VIA VT 8237R (vt8233)
47382 00010027 0F84A4180000        <1>       je    vt8233_volume
47383                              <1>       ; 28/05/2017
47384 0001002D 0F8738FFFFFF        <1>       ja    snd_nothing ; temporary.
47385                              <1>       ;ja   hda_volume
47386                              <1>       ; Sound Blaster 16
47387 00010033 3C01                <1>       cmp   al, 1 ; SB 16
47388 00010035 0F84321B0000        <1>       je    sb16_volume
47389 0001003B E9EF1D0000          <1>       jmp   ac97_volume
47390                              <1>
47391                              <1> soundc_disable:
47392                              <1>       ; FUNCTION = 12
47393                              <1>       ; Disable audio device (and unlink DMA memory)
47394                              <1>       ; 28/05/2017
47395                              <1>       ; 24/05/2017
47396                              <1>       ; 22/04/2017
47397 00010040 E8C3010000          <1>       call  snd_dev_check
47398 00010045 0F827DFBFFFF        <1>       jc    soundc_dev_err ; temporary.
47399                              <1>       ;call snd_buf_check
47400                              <1>       ;jc   sndc_owner_error ; temporary.
47401                              <1>
47402 0001004B A0[A5650100]        <1>       mov   al, [audio_device]
47403 00010050 3C03                <1>       cmp   al, 3 ; VIA VT 8237R (vt8233)
47404 00010052 7418                <1>       je    short snd_disable_1
47405 00010054 0F8711FFFFFF        <1>       ja    snd_nothing ; temporary.
47406 0001005A 3C01                <1>       cmp   al, 1 ; Sound Blaster 16
47407 0001005C 7507                <1>       jne   short snd_disable_0
47408 0001005E E8851B0000          <1>       call  sb16_stop
47409 00010063 EB0C                <1>       jmp   short snd_disable_2
47410                              <1> snd_disable_0:
47411 00010065 E8B31E0000          <1>       call  ac97_stop
47412 0001006A EB05                <1>       jmp   short snd_disable_2
47413                              <1> snd_disable_1:
47414 0001006C E842170000          <1>       call  vt8233_stop
47415                              <1> snd_disable_2:
47416 00010071 A0[A7650100]        <1>       mov   al, [audio_intr]
47417 00010076 29DB                <1>       sub   ebx, ebx ; 0 = reset
47418 00010078 E85EFAFFFF          <1>       call  set_dev_IRQ_service
47419                              <1>
47420                              <1>       ;mov  al, [audio_intr]
47421 0001007D 28E4                <1>       sub   ah, ah ; 0 = reset
47422 0001007F E8C0F6FFFF          <1>       call  set_hardware_int_vector
47423                              <1>
47424 00010084 31C0                <1>       xor   eax, eax
47425 00010086 A2[A5650100]        <1>       mov   byte [audio_device], al
47426 0001008B A2[A7650100]        <1>       mov   byte [audio_intr], al
47427 00010090 8705[C4650100]      <1>       xchg  eax, [audio_dma_buff]
47428                              <1>       ; 24/05/2017
47429                              <1>       ;or   eax, eax
47430                              <1>       ;jz   short snd_disable_3
47431                              <1>       ;cmp  eax, sb16_dma_buffer ; default DMA buffer
47432                              <1>       ;je   short snd_disable_3
47433 00010096 803D[A4650100]00    <1>       cmp   byte [audio_pci], 0 ; AC97 audio controller ?
47434 0001009D 7612                <1>       jna   short snd_disable_3
47435 0001009F C605[A4650100]00    <1>       mov   byte [audio_pci], 0
47436                              <1>       ;sub  ecx, ecx
47437                              <1>       ;xchg ecx, [audio_dmabuff_size]
47438 000100A6 8B0D[C8650100]      <1>       mov   ecx, [audio_dmabuff_size]
47439 000100AC E88055FFFF          <1>       call  deallocate_memory_block
47440                              <1> snd_disable_3:
47441 000100B1 C3                  <1>       retn
47442                              <1>
47443                              <1> sound_dma_map:
47444                              <1>       ; FUNCTION = 13
47445                              <1>       ; Map audio dma buff addr to user's buffer addr
47446                              <1>       ; 12/05/2017
47447 000100B2 21C9                <1>       and   ecx, ecx
47448 000100B4 0F8415FBFFFF        <1>       jz    sound_buff_error
47449 000100BA 803D[A5650100]01    <1>       cmp   byte [audio_device], 1
47450 000100C1 7229                <1>       jb    short snd_dma_map_1
47451                              <1> snd_dma_map_0:
47452 000100C3 A1[C4650100]        <1>       mov   eax, [audio_dma_buff]
47453 000100C8 21C0                <1>       and   eax, eax
47454 000100CA 7420                <1>       jz    short snd_dma_map_1
47455                              <1>       ;
47456 000100CC 8A1D[CD650100]      <1>       mov   bl, [audio_user]
47457 000100D2 08DB                <1>       or    bl, bl
47458 000100D4 7416                <1>       jz    short snd_dma_map_1
47459 000100D6 3A1D[B3030300]      <1>       cmp   bl, [u.uno]
47460 000100DC 0F8531FCFFFF        <1>       jne   sndc_owner_error
47461                              <1>       ;
47462 000100E2 8B1D[C8650100]      <1>       mov   ebx, [audio_dmabuff_size]
47463 000100E8 21DB                <1>       and   ebx, ebx
47464 000100EA 750A                <1>       jnz   short snd_dma_map_2
47465                              <1> snd_dma_map_1:
47466 000100EC B8[00000200]        <1>       mov   eax, sb16_dma_buffer
47467 000100F1 BB00000100          <1>       mov   ebx, 65536
47468                              <1> snd_dma_map_2:
47469 000100F6 81C1FF0F0000        <1>       add   ecx, PAGE_SIZE-1 ; 4095
47470 000100FC 6681E100F0          <1>       and   cx, ~PAGE_OFF ; not 4095
47471 00010101 39D9                <1>       cmp   ecx, ebx
47472 00010103 0F87C6FAFFFF        <1>       ja    sound_buff_error
47473 00010109 50                  <1>       push  eax
47474 0001010A 89D3                <1>       mov   ebx, edx
47475 0001010C C1E90C              <1>       shr   ecx, 12 ; byte count to page count
```

```
47476                                  <1>       ; eax = physical address of (audio) dma buffer
47477                                  <1>       ; ebx = virtual address of (audio) dma buffer (user's pgdir)
47478                                  <1>       ; ecx = page count (>0)
47479 0001010F E88D55FFFF              <1>       call   direct_memory_access
47480 00010114 58                      <1>       pop    eax
47481 00010115 0F82B4FAFFFF            <1>       jc     sound_buff_error
47482 0001011B A3[64030300]            <1>       mov    [u.r0], eax
47483 00010120 C3                      <1>       retn
47484                                  <1>
47485                                  <1> soundc_info:
47486                                  <1>       ; FUNCTION = 14
47487                                  <1>       ; Get Audio Controller Info
47488                                  <1>       ; 10/06/2017
47489                                  <1>       ; 05/06/2017
47490 00010121 20DB                    <1>       and    bl, bl ; 0
47491 00010123 740A                    <1>       jz     short sndc_info_0
47492                                  <1>       ; invalid parameter !
47493 00010125 B817000000              <1>       mov    eax, ERR_INV_PARAMETER ; 23
47494                                  <1> ;sndc_inf_error:
47495                                  <1> ;     mov    [u.r0], eax
47496                                  <1> ;     mov    [u.error], eax
47497                                  <1> ;     jmp    error
47498 0001012A E9ACFAFFFF              <1>       jmp    sysaudio_err
47499                                  <1>
47500                                  <1> sndc_info_0:
47501 0001012F E8D4000000              <1>       call   snd_dev_check
47502 00010134 0F828EFAFFFF            <1>       jc     soundc_dev_err
47503                                  <1>
47504 0001013A 8B1D[B0650100]          <1>       mov    ebx, [audio_vendor]
47505 00010140 8B0D[AC650100]          <1>       mov    ecx, [audio_dev_id]
47506                                  <1>       ;mov   al,  [audio_device]
47507 00010146 3C02                    <1>       cmp    al, 2 ; AC'97 (ICH)
47508 00010148 7513                    <1>       jne    short sndc_info_1
47509                                  <1>       ; Intel AC97 (ICH) Audio Controller (=2)
47510 0001014A 668B15[DE650100]        <1>       mov    dx, [NABMBAR]
47511 00010151 C1E210                  <1>       shl    edx, 16
47512 00010154 668B15[DC650100]        <1>       mov    dx, [NAMBAR]
47513 0001015B EB07                    <1>       jmp    short sndc_info_2
47514                                  <1> sndc_info_1:
47515                                  <1>       ; 05/06/2017
47516                                  <1>       ; Note: Intel HDA code (here) is not ready yet!
47517                                  <1>       ; !!! SB16 or VT8233 (VT8237R) !!!
47518 0001015D 0FB715[AA650100]        <1>       movzx  edx, word [audio_io_base]
47519                                  <1> sndc_info_2:
47520 00010164 88C4                    <1>       mov    ah, al ;  [audio_device]
47521 00010166 A0[A7650100]            <1>       mov    al, [audio_intr]
47522                                  <1>
47523                                  <1>       ; EAX = IRQ Number in AL
47524                                  <1>       ;       Audio Device Number in AH
47525                                  <1>       ; EBX = DEV/VENDOR ID
47526                                  <1>       ;       (DDDDDDDDDDDDDDDDVVVVVVVVVVVVVVVV)
47527                                  <1>       ; ECX = BUS/DEV/FN
47528                                  <1>       ;       (00000000BBBBBBBBDDDDDFFF00000000)
47529                                  <1>       ; EDX = NABMBAR/NAMBAR (for AC97)
47530                                  <1>       ;       (Low word, DX = NAMBAR address)
47531                                  <1>       ; EDX = Base IO Addr (DX) for SB16 & VT8233
47532                                  <1>
47533                                  <1>       ; 10/06/2017
47534 0001016B A3[64030300]            <1>       mov    [u.r0], eax
47535 00010170 8B2D[60030300]          <1>       mov    ebp, [u.usp]
47536 00010176 895D10                  <1>       mov    [ebp+16], ebx  ; ebx
47537 00010179 895514                  <1>       mov    [ebp+20], edx  ; edx
47538 0001017C 894D18                  <1>       mov    [ebp+24], ecx  ; ecx
47539                                  <1>
47540 0001017F C3                      <1>       retn
47541                                  <1>
47542                                  <1> sound_data:
47543                                  <1>       ; FUNCTION = 15
47544                                  <1>       ; Get Current Sound data for graphics
47545                                  <1>       ; 22/06/2017
47546                                  <1>       ;
47547 00010180 E883000000              <1>       call   snd_dev_check
47548 00010185 0F823DFAFFFF            <1>       jc     soundc_dev_err ; Device not ready !
47549                                  <1>
47550 0001018B 80FB00                  <1>       cmp    bl, 0
47551 0001018E 760A                    <1>       jna    short sound_data_0
47552                                  <1>
47553                                  <1>       ; Only PCM OUT buffer data is valid for now!
47554 00010190 B817000000              <1>       mov    eax, ERR_INV_PARAMETER  ; 23
47555 00010195 E941FAFFFF              <1>       jmp    sysaudio_err
47556                                  <1>
47557                                  <1> sound_data_0:
47558 0001019A A1[C4650100]            <1>       mov    eax, [audio_dma_buff]
47559 0001019F 09C0                    <1>       or     eax, eax
47560 000101A1 0F8428FAFFFF            <1>       jz     sound_buff_error
47561                                  <1>
47562 000101A7 803D[A5650100]04        <1>       cmp    byte [audio_device], 4 ; Intel HDA
47563 000101AE 744F                    <1>       je     short sound_data_4 ; temporary ! (22/06/2017)
47564                                  <1>
47565 000101B0 21C9                    <1>       and    ecx, ecx
47566                                  <1>       ;jnz   short sound_data_1 ; sample tranfer
47567                                  <1>
47568                                  <1>       ; Return only DMA Buffer pointer/offset...
47569                                  <1>       ; (If DMA Buffer has been mapped to user's
47570                                  <1>       ;  memory space; program can get graphics
47571                                  <1>       ;  data by using only this pointer value.)
47572                                  <1>
47573                                  <1>       ;call  get_dma_buffer_offset
47574                                  <1>       ;; eax = DMA buffer offset
47575                                  <1>       ;;     (!not half buffer offset!)
47576                                  <1>       ;mov   [u.r0], eax
47577                                  <1>       ;retn
47578                                  <1>
```

```
47579 000101B2 0F84441F0000        <1>       jz    get_dma_buffer_offset
47580                              <1>
47581                              <1> sound_data_1:
47582                              <1>       ;mov   eax, [audio_dmabuff_size]
47583                              <1>       ;shr   eax, 1 ; half buffer size
47584                              <1>       ;cmp   ecx, eax
47585                              <1>       ;ja    short sound_buff_error
47586                              <1>
47587 000101B8 3B0D[C8650100]      <1>       cmp    ecx, [audio_dmabuff_size]
47588 000101BE 0F870BFAFFFF        <1>       ja     sound_buff_error
47589                              <1>
47590 000101C4 89D0                <1>       mov    eax, edx
47591 000101C6 25FF0F0000          <1>       and    eax, PAGE_OFF ; 4095 (0FFFh)
47592 000101CB 81F900100000        <1>       cmp    ecx, 4096
47593 000101D1 7605                <1>       jna    short sound_data_2
47594 000101D3 B900100000          <1>       mov    ecx, 4096 ; max. 1 page
47595                              <1> sound_data_2:
47596 000101D8 01C8                <1>       add    eax, ecx
47597 000101DA 3D00100000          <1>       cmp    eax, 4096
47598 000101DF 7606                <1>       jna    short sound_data_3
47599 000101E1 6625FF0F            <1>       and    ax, PAGE_OFF ; 4095 (0FFFh)
47600 000101E5 29C1                <1>       sub    ecx, eax
47601                              <1>       ; here, ECX has been adjusted to fit
47602                              <1>       ;      in page border..  (<= 4096, >0)
47603                              <1> sound_data_3:
47604 000101E7 51                  <1>       push   ecx
47605 000101E8 52                  <1>       push   edx
47606 000101E9 89D3                <1>       mov    ebx, edx
47607 000101EB E89F50FFFF          <1>       call   get_physical_addr
47608 000101F0 5A                  <1>       pop    edx
47609 000101F1 59                  <1>       pop    ecx
47610 000101F2 0F82D7F9FFFF        <1>       jc     sound_buff_error
47611                              <1>
47612                              <1>       ; eax = physical address of user's buffer
47613 000101F8 89C3                <1>       mov    ebx, eax
47614                              <1>       ; ecx = byte (transfer) count
47615                              <1>       ;call  get_current_sound_data
47616                              <1>       ;retn
47617 000101FA E9721E0000          <1>       jmp    get_current_sound_data
47618                              <1>
47619                              <1> sound_data_4:
47620                              <1>       ; Intel HDA code is not ready yet !
47621                              <1>       ; 22/06/2017
47622 000101FF 31C0                <1>       xor    eax, eax
47623 00010201 48                  <1>       dec    eax
47624 00010202 A3[64030300]        <1>       mov    [u.r0], eax ; 0FFFFFFFFh
47625 00010207 C3                  <1>       retn
47626                              <1>
47627                              <1> snd_dev_check:
47628                              <1>       ; 10/06/2017
47629                              <1>       ; 05/06/2017
47630                              <1>       ; 24/05/2017
47631                              <1>       ; 22/04/2017
47632                              <1>       ; 21/04/2017
47633                              <1>       ; ... device check at first
47634 00010208 A0[A5650100]        <1>       mov    al, [audio_device]
47635 0001020D 3C01                <1>       cmp    al, 1 ; SB 16
47636 0001020F 7203                <1>       jb     short snd_dev_chk_retn ; error !
47637                              <1>       ;cmp   al, 4 ; Intel HDA
47638                              <1>       ;ja    short snd_dbchk_stc ; invalid !
47639                              <1>       ; 10/06/2017
47640 00010211 3C05                <1>       cmp    al, 5
47641 00010213 F5                  <1>       cmc
47642                              <1> snd_dev_chk_retn:
47643 00010214 C3                  <1>       retn
47644                              <1>
47645                              <1> snd_buf_check:
47646                              <1>       ; 10/06/2017
47647                              <1>       ; 22/04/2017
47648                              <1>       ; 21/04/2017
47649                              <1>       ; ... buffer & (buffer) owner check at second
47650 00010215 833D[B8650100]00    <1>       cmp    dword [audio_buffer], 0
47651 0001021C 760D                <1>       jna    short snd_dbchk_stc
47652                              <1> snd_user_check:
47653 0001021E A0[B3030300]        <1>       mov    al, [u.uno]
47654 00010223 3A05[CD650100]      <1>       cmp    al, [audio_user]
47655                              <1>       ;jne   short snd_dbchk_stc
47656                              <1>       ;retn
47657 00010229 74E9                <1>       je     short snd_dev_chk_retn
47658                              <1>
47659                              <1> snd_dbchk_stc:
47660 0001022B F9                  <1>       stc
47661 0001022C C3                  <1>       retn
47662                              <1>
47663                              <1> sound_update:
47664                              <1>       ; FUNCTION = 16
47665                              <1>       ; bl =
47666                              <1>       ;    0 = automatic (sequental) update (with flag switch!)
47667                              <1>       ;    1 = update dma half buffer 1 (without flag switch!)
47668                              <1>       ;    2 = update dma half buffer 2 (without flag switch!)
47669                              <1>       ;  FFh = get current flag value
47670                              <1>       ;      0 = dma half buffer 1 (will be played next)
47671                              <1>       ;      1 = dma half buffer 2 (will be played next)
47672                              <1>
47673                              <1>       ; 10/10/2017
47674                              <1>
47675                              <1>       ; ... device check at first
47676 0001022D A0[A5650100]        <1>       mov    al, [audio_device]
47677 00010232 08C0                <1>       or     al, al ; 0 ; pc speaker or invalid
47678 00010234 0F848EF9FFFF        <1>       jz     soundc_dev_err
47679                              <1>
47680                              <1>       ; ... buffer & (buffer) owner check at second
47681 0001023A 833D[B8650100]00    <1>       cmp    dword [audio_buffer], 0
```

```
47682 00010241 0F8688F9FFFF        <1>       jna    sound_buff_error
47683 00010247 A0[B3030300]        <1>       mov    al, [u.uno]
47684 0001024C 3A05[CD650100]      <1>       cmp    al, [audio_user]
47685 00010252 0F85BBFAFFFF        <1>       jne    sndc_owner_error
47686                              <1>
47687                              <1>       ; Transfer ring 3 (user's) audio buffer content to dma buffer
47688 00010258 8B3D[C4650100]      <1>       mov    edi, [audio_dma_buff] ; dma buffer (ring 0)
47689 0001025E 09FF                <1>       or     edi, edi
47690 00010260 0F8469F9FFFF        <1>       jz     sound_buff_error
47691 00010266 8B35[BC650100]      <1>       mov    esi, [audio_p_buffer] ; physical address (ring 3)
47692 0001026C 8B0D[C0650100]      <1>       mov    ecx, [audio_buff_size]
47693                              <1>
47694                              <1>       ;movzx eax, byte [audio_flag]
47695 00010272 A0[CC650100]        <1>       mov    al, [audio_flag]
47696                              <1>
47697 00010277 FEC3                <1>       inc    bl
47698 00010279 7427                <1>       jz     short snd_update_3 ; bl = 0FFh
47699 0001027B FECB                <1>       dec    bl
47700 0001027D 7411                <1>       jz     short snd_update_0 ; bl = 0
47701                              <1>
47702 0001027F 80FB02              <1>       cmp    bl, 2
47703 00010282 7417                <1>       je     short snd_update_1 ; dma half buffer 2
47704 00010284 7217                <1>       jb     short snd_update_2 ; dma half buffer 1
47705                              <1>
47706                              <1>       ; invalid parameter !
47707 00010286 B817000000          <1>       mov    eax, ERR_INV_PARAMETER ; 23
47708                              <1> ;     mov    [u.r0], eax
47709                              <1> ;     mov    [u.error], eax
47710                              <1> ;     jmp    error
47711 0001028B E94BF9FFFF          <1>       jmp    sysaudio_err
47712                              <1>
47713                              <1> snd_update_0:
47714 00010290 8035[CC650100]01    <1>       xor    byte [audio_flag], 1 ; update flag !!!
47715 00010297 3C01                <1>       cmp    al, 1
47716 00010299 7202                <1>       jb     short snd_update_2 ; dma half buffer 1
47717                              <1> snd_update_1:
47718                              <1>       ; dma half buffer 2
47719 0001029B 01CF                <1>       add    edi, ecx
47720                              <1> snd_update_2:
47721                              <1>       ;rep   movsb
47722 0001029D C1E902              <1>       shr    ecx, 2
47723 000102A0 F3A5                <1>       rep    movsd
47724                              <1> snd_update_3:
47725 000102A2 A3[64030300]        <1>       mov    [u.r0], eax
47726                              <1>
47727 000102A7 C3                  <1>       retn
47728                              <1>
47729                              <1>
47730                              <1> set_irq_callback_service:
47731                              <1>       ; 10/06/2017
47732                              <1>       ; 12/05/2017
47733                              <1>       ; 24/04/2017
47734                              <1>       ; 22/04/2017
47735                              <1>       ; caller: 'syscalbac' or 'sysaudio' or ...
47736                              <1>       ; 13/04/2017, 14/04/2017, 17/04/2017
47737                              <1>       ; 24/02/2017, 26/02/2017, 28/02/2017
47738                              <1>       ; 21/02/2017 - TRDOS 386 (TRDOS v2.0)
47739                              <1>       ;
47740                              <1>       ; Link or unlink IRQ callback service to/from user (ring 3)
47741                              <1>       ;
47742                              <1>       ; INPUT ->
47743                              <1>       ;     If AL = 0, the caller is 'syscalbac';
47744                              <1>       ;        otherwise, the caller is 'sysaudio' or ...
47745                              <1>       ;        (AL = user number)
47746                              <1>       ;
47747                              <1>       ;     BL = IRQ number (Hardware interrupt request number)
47748                              <1>       ;          (0 t0 15 but IRQ 0,1,2,6,8,14,15 are prohibited)
47749                              <1>       ;          IRQ numbers 3,4,5,7,9,10,11,12,13 are valid
47750                              <1>       ;          (numbers >15 are invalid)
47751                              <1>       ;
47752                              <1>       ;     BH = 0 = Unlink IRQ (in BL) from user (ring 3) service
47753                              <1>       ;          1 = Link IRQ by using Signal Response Byte method
47754                              <1>       ;          2 = Link IRQ by using Callback service method
47755                              <1>       ;          3 = Link IRQ by using Auto Increment S.R.B. method
47756                              <1>       ;         >3 = invalid
47757                              <1>       ;            (syscallback version will return to user)
47758                              <1>       ;
47759                              <1>       ;     CL = Signal Return/Response Byte value
47760                              <1>       ;
47761                              <1>       ;     If BH = 2, kernel will put a counter value
47762                              <1>       ;               (into the S.R.B. addr)
47763                              <1>       ;              between 0 to 255. (start value = CL+1)
47764                              <1>       ;
47765                              <1>       ;     NOTE: counter value, for example: even and odd numbers
47766                              <1>       ;           may be used for -audio- DMA buffer switch
47767                              <1>       ;           within double buffer method, etc.
47768                              <1>       ;
47769                              <1>       ;     EDX = Signal return (Response) byte address
47770                              <1>       ;                          - or -
47771                              <1>       ;           Interrupt/Callback service/routine address
47772                              <1>       ;
47773                              <1>       ;           (virtual address in user's memory space)
47774                              <1>       ;
47775                              <1>       ; OUTPUT ->
47776                              <1>       ;     CF = 0 & EAX = 0 -> Successful setting
47777                              <1>       ;     CF = 1 & EAX > 0 -> IRQ is prohibited or locked
47778                              <1>       ;                    by another process
47779                              <1>       ;             eax = ERR_PERM_DENIED -> prohibited or locked
47780                              <1>       ;             eax = ERR_INV_PARAMETER ->
47781                              <1>       ;                   invalid parameter/option or bad address
47782                              <1>       ;
47783                              <1>       ; TRDOS 386 - IRQ CALLBACK structures (parameters):
47784                              <1>       ;
```

```
47785                                  <1>       ;       [u.irqlock] = 1 word, IRQ flags (0-15) that indicates
47786                                  <1>       ;                 which IRQs are locked by (that) user.
47787                                  <1>       ;                  Lock and unlock (by user) will change
47788                                  <1>       ;                 these flags or 'terminate process' (sysexit)
47789                                  <1>       ;                 will clear these flags and unlock those IRQs.
47790                                  <1>       ;
47791                                  <1>       ;                 Bit 0 is for IRQ 0 and Bit 15 is for IRQ 15
47792                                  <1>       ;
47793                                  <1>       ;       IRQ(x).owner      : 1 byte, user, [u.uno], 0 = free (unlocked)
47794                                  <1>       ;
47795                                  <1>       ;       IRQ(x).method : 1 byte for callback method & status
47796                                  <1>       ;                    0 = Signal Response Byte method
47797                                  <1>       ;                    1 = Callback service method
47798                                  <1>       ;                    >1 = invalid for current 'syscalback'.
47799                                  <1>       ;                 or(+) 80h = IRQ is in use by system (ring 0)
47800                                  <1>       ;                             function (audio etc.) or
47801                                  <1>       ;                           a device driver.
47802                                  <1>       ;                 (system function will ignore the lock/owner)
47803                                  <1>       ;
47804                                  <1>       ;       IRQ(x).srb: 1 byte, Signal Return/Response byte value
47805                                  <1>       ;                    (a fixed value by user or a counter value
47806                                  <1>       ;                   from 0 to 255, which is increased by every
47807                                  <1>       ;                   interrupt just before putting it into
47808                                  <1>       ;                   the Signal Response byte address
47809                                  <1>       ;                   (This is not used in callback serv method)
47810                                  <1>       ;
47811                                  <1>       ;       IRQ(x).addr      : 1 dword
47812                                  <1>       ;                    Signal Response Byte address (physical)
47813                                  <1>       ;                              -or-
47814                                  <1>       ;                    Callback service address (virtual)
47815                                  <1>       ;
47816                                  <1>       ;       IRQ(x).dev: 1 byte
47817                                  <1>       ;                              0 = Default device or kernel function
47818                                  <1>       ;                                        -or-
47819                                  <1>       ;                              1-255 = Assigned device driver number
47820                                  <1>       ;
47821                                  <1>       ;          (x) = 3,4,5,7,9,10,11,12,13
47822                                  <1>       ;
47823                                  <1>
47824 000102A8 80FB0F                 <1>       cmp    bl, 15
47825 000102AB 7729                   <1>       ja     short scbs_2
47826                                  <1>
47827 000102AD 80FF03                 <1>       cmp    bh, 3
47828 000102B0 7724                   <1>       ja     short scbs_2  ; invalid parameter
47829                                  <1>
47830 000102B2 0FB6FB                 <1>       movzx  edi, bl ; save IRQ number
47831                                  <1>
47832                                  <1>              ;  IRQ 0,1,2,6,8,14,15 are prohibited
47833                                  <1>       ;IRQenum: ; 'trdosk9.s'
47834                                  <1>       ;      db  0,0,0,1,2,3,0,4,0,5,6,7,8,9,0,0
47835                                  <1>
47836 000102B5 0FB6B7[08100100]       <1>       movzx  esi, byte [edi+IRQenum] ; IRQ availability
47837                                  <1>                                      ; enumeration/index
47838                                  <1>       ;dec   esi
47839 000102BC 664E                   <1>       dec    si
47840 000102BE 780F                   <1>       js     short scbs_1 ;  0 -> 0FFFFh
47841                                  <1>
47842                                  <1>       ; ESI = IRQ callback parameters index number (0 to 8)
47843                                  <1>
47844 000102C0 08FF                   <1>       or     bh, bh
47845 000102C2 7419                   <1>       jz     short scbs_4 ; unlink the IRQ (in BL)
47846                                  <1>
47847 000102C4 FECF                   <1>       dec    bh
47848                                  <1>       ; bh = method (0 = signal response byte, 1 = callback)
47849                                  <1>       ;            (2 = auto increment of signal response byte)
47850                                  <1>
47851 000102C6 80BE[56650100]00       <1>       cmp    byte [esi+IRQ.owner], 0 ; locked ?
47852 000102CD 7637                   <1>       jna    short scbs_6 ; no... OK...
47853                                  <1>
47854                                  <1> scbs_1:
47855                                  <1>       ; permission denied (prohibited IRQ)
47856 000102CF B80B000000             <1>       mov    eax, ERR_PERM_DENIED
47857 000102D4 F9                     <1>       stc
47858 000102D5 C3                     <1>       retn
47859                                  <1> scbs_2:
47860 000102D6 F9                     <1>       stc
47861                                  <1> scbs_3:
47862 000102D7 B817000000             <1>       mov    eax, ERR_INV_PARAMETER
47863 000102DC C3                     <1>       retn
47864                                  <1>
47865                                  <1> scbs_4: ; unlink the requested IRQ (if it belongs to current user)
47866                                  <1>       ; 10/06/2017
47867                                  <1>       ; 22/04/2017
47868                                  <1>       ; 14/04/2017
47869                                  <1>       ; If AL = 0 -> The caller is 'syscalbac'
47870 000102DD 8AA6[56650100]         <1>       mov    ah, [esi+IRQ.owner]
47871 000102E3 3A25[B3030300]         <1>       cmp    ah, [u.uno]
47872 000102E9 75E4                   <1>       jne    short scbs_1
47873                                  <1>
47874 000102EB FE0D[D6030300]         <1>       dec    byte [u.irqc] ; decrease IRQ count (in use)
47875                                  <1>
47876                                  <1>       ;sub   ah, ah
47877                                  <1>       ;mov   [esi+IRQ.owner], ah ; 0 ; free !!!
47878                                  <1>       ;and   byte [esi+IRQ.method], 80h
47879                                  <1>       ;mov   [esi+IRQ.srb], ah ; 0
47880                                  <1>       ;mov   [esi+IRQ.dev], ah ; 0
47881                                  <1>       ;mov   dword [esi+IRQ.addr], 0
47882                                  <1>       ;mov   dword [u.r0], 0
47883                                  <1>
47884                                  <1>       ;mov   byte [esi+IRQ.owner], 0
47885                                  <1>
47886                                  <1>       ; 22/04/2017
47887 000102F1 29C0                   <1>       sub    eax, eax
```

```
47888 000102F3 8886[56650100]      <1>       mov    [esi+IRQ.owner], al ; 0
47889                              <1>       ; 10/06/2017
47890 000102F9 8686[68650100]      <1>       xchg   al, [esi+IRQ.method]
47891 000102FF 2480                <1>       and    al, 80h
47892 00010301 745E                <1>       jz     short scbs_12
47893                              <1>       ; Audio device must be disabled -later- ! ([IRQ.medhod] = 80h)
47894                              <1>
47895                              <1> ;     cmp    byte [esi+IRQ.method], 80h ; device drv or kernel extension ?
47896                              <1> ;     jb     short scbs_12 ; bh = 0 reset to default IRQ handler
47897                              <1> ;
47898                              <1> ;     and    al, al
47899                              <1> ;     jz     short  scbs_5  ; the caller is 'syscalbac'
47900                              <1> ;     ; The caller is 'sysaudio' or ...
47901 00010303 30C0                <1>       xor    al, al
47902                              <1> ;     mov    [esi+IRQ.method], al ; 0 ; reset kernel extension flag
47903                              <1> ;scbs_5:
47904                              <1> ;     sub    ah, ah
47905                              <1>       ;mov   [u.r0], eax ; 0
47906 00010305 C3                  <1>       retn
47907                              <1>
47908                              <1> scbs_6:
47909                              <1>       ; 14/04/2017
47910 00010306 20C0                <1>       and    al, al
47911 00010308 7405                <1>       jz     short scbs_7  ; the caller is 'syscalbac'
47912                              <1>       ; AL = user number ([u.uno] or [audio.user] or ...)
47913                              <1>       ; The caller is 'sysaudio' or ...
47914                              <1>       ;
47915                              <1>       ; bh = method (0 = signal response byte, 1 = callback)
47916                              <1>       ;           (2 = auto increment of signal response byte)
47917                              <1>
47918 0001030A 80CF80              <1>       or     bh, 80h             ; Kernel extension flag !
47919 0001030D EB0A                <1>       jmp    short scbs_8
47920                              <1> scbs_7:
47921 0001030F 8A86[68650100]      <1>       mov    al, [esi+IRQ.method] ; >= 80h = kernel is using this IRQ
47922 00010315 2480                <1>       and    al, 80h ; use only bit 7 (kernel function flag)
47923 00010317 08C7                <1>       or     bh, al        ; method
47924                              <1>                            ; 0 = signal response byte, 1 = callback
47925                              <1>                            ; 2 = auto increment of s.r.b.
47926                              <1> scbs_8:
47927 00010319 A0[B3030300]        <1>       mov    al, [u.uno] ; user (process) number (1 to 16)
47928 0001031E 8886[56650100]      <1>       mov    [esi+IRQ.owner], al ; lock the IRQ for user
47929 00010324 88BE[68650100]      <1>       mov    [esi+IRQ.method], bh
47930                              <1>
47931                              <1> ;     test   bh, 1
47932                              <1> ;     jnz    short scbs_9  ; Callback method, CX will not be used
47933                              <1> ;
47934                              <1> ;     test   bh, 2         ; use auto increment (counter) method
47935                              <1> ;     jz     short scbs_10 ; (count can be used for buffer switch)
47936                              <1> ;scbs_9:
47937                              <1> ;     xor    ecx, ecx ; 0
47938                              <1> scbs_10:
47939                              <1>       ;mov   [esi+IRQ.method], bh
47940 0001032A 888E[71650100]      <1>       mov    [esi+IRQ.srb], cl
47941 00010330 C686[5F650100]00    <1>       mov    byte [esi+IRQ.dev], 0 ; device number is always 0
47942                              <1>                            ; for this system call
47943                              <1>       ;test  bh, 1
47944 00010337 80E701              <1>       and    bh, 1 ; 17/04/2017
47945 0001033A 7513                <1>       jnz    short scbs_11 ; callback method, use virtual address
47946                              <1>
47947 0001033C 53                  <1>       push   ebx ; IRQ number (in BL)
47948 0001033D 89D3                <1>       mov    ebx, edx
47949                              <1>       ; ebx = virtual address
47950                              <1>       ; [u.pgdir] = page directory's physical address
47951 0001033F FE05[F6640100]      <1>       inc    byte [no_page_swap] ; 1
47952                              <1>                   ; Do not add this page to swap queue
47953                              <1>                   ; and remove it from swap queue if it is
47954                              <1>                   ; on the queue.
47955 00010345 E8454FFFFF          <1>       call   get_physical_addr
47956 0001034A 5B                  <1>       pop    ebx
47957 0001034B 728A                <1>       jc     scbs_3 ; invalid address !
47958                              <1>       ; eax = physical address of the virtual address in user's space
47959 0001034D 89C2                <1>       mov    edx, eax
47960                              <1> scbs_11:
47961 0001034F 66C1E602            <1>       shl    si, 2        ; byte (index) to dword (offset)
47962 00010353 8996[7A650100]      <1>       mov    [esi+IRQ.addr], edx
47963                              <1>
47964 00010359 FE05[D6030300]      <1>       inc    byte [u.irqc]; increase IRQ (in use) count
47965                              <1>
47966 0001035F FEC7                <1>       inc     bh ; 17/04/2017
47967                              <1>       ; bh > 0 -> set to requested IRQ handler (IRQ_u_list)
47968                              <1> scbs_12:
47969 00010361 88D8                <1>       mov    al, bl ; IRQ number
47970 00010363 88FC                <1>       mov    ah, bh ; 0 = reset, >0 = set
47971 00010365 E8DAF3FFFF          <1>       call   set_hardware_int_vector
47972                              <1>
47973 0001036A 31C0                <1>       xor    eax, eax
47974                              <1>       ;mov   [u.r0], eax ; 0
47975                              <1>
47976 0001036C C3                  <1>       retn   ; return with success (cf=0, eax=0)
47977                              <1>
47978                              <1>
47979                              <1> sysdma: ; DMA FUNCTIONS
47980                              <1>       ; 02/09/2017
47981                              <1>       ; 28/08/2017
47982                              <1>       ; 20/08/2017 - TRDOS 386 (TRDOS v2.0)
47983                              <1>       ;
47984                              <1>       ; Inputs:
47985                              <1>       ;     BH = 0 -> Allocate DMA buffer
47986                              <1>       ;        BL = 0 -> Use the system's default DMA
47987                              <1>       ;                (SB16) Buffer
47988                              <1>       ;             Buffer Size (max.) = 65536 bytes
47989                              <1>       ;        BL > 0 -> Allocate (a new) DMA buffer
47990                              <1>       ;        ECX = DMA Buffer Size in bytes (<=128KB)
```

```
                        <1>       ;        EDX = Virtual Address of DMA buffer
                        <1>       ;
                        <1>       ;     BH = 1 -> Initialize (Start) DMA service
                        <1>       ;          BL, bit 0 to 3 = Channel Number (0 to 7)
                        <1>       ;          BL, bit 7 = Auto Initialized Mode
                        <1>       ;                  (If bit 7 is set)
                        <1>       ;             bit 6 = Record (read) mode (0= playback)
                        <1>       ;          ECX = byte count (0 = use dma buffer size)
                        <1>       ;          EDX = physical buffer address
                        <1>       ;               (0 = use dma buffer -start- address)
                        <1>       ;
                        <1>       ;     BH = 2 -> Get Current DMA Buffer Offset
                        <1>       ;          BL = DMA channel number
                        <1>       ;
                        <1>       ;     BH = 3 -> Get Current DMA count down value
                        <1>       ;          BL = DMA channel number (0 tO 7)
                        <1>       ;
                        <1>       ;     BH = 4 -> Get Current DMA channel (in progress)
                        <1>       ;
                        <1>       ;     BH = 5 -> Get System's Default DMA Buffer Address
                        <1>       ;
                        <1>       ;     BH = 6 -> Get Current DMA Buffer Address
                        <1>       ;
                        <1>       ;     BH = 7 -> Stop DMA service
                        <1>       ;
                        <1>       ;
                        <1>       ; Outputs:
                        <1>       ;
                        <1>       ;     For BH = 0 ; Allocate DMA buffer
                        <1>       ;         EAX = Physical address of DMA buffer
                        <1>       ;         ECX = Allocated buffer size in bytes
                        <1>       ;              - page count * 4096 -
                        <1>       ;                (may be bigger than requested)
                        <1>       ;           If BL input > 0,
                        <1>       ;              'sysalloc:' system call will be used with
                        <1>       ;              EBX (for 'sysalloc') = EDX (for 'sysdma')
                        <1>       ;              ECX is same, byte count (buffer size)
                        <1>       ;              EDX = 1024*1024*16 ; 16 MB upper limit
                        <1>       ;           If BL input = 0,
                        <1>       ;              Default DMA buffer (SB16 buffer) will be
                        <1>       ;              checked and if it is free, it's address
                        <1>       ;              will be returned in EAX and it's size
                        <1>       ;              will be returned in ECX (as 65536)
                        <1>       ;
                        <1>       ;           If CF = 1, error code is in EAX
                        <1>       ;              EAX = -1 ; DMA buffer allocation error!
                        <1>       ;              EAX = 11 ; 'Permission Denied' error !
                        <1>       ;
                        <1>       ;              Note: 'sysalloc' error return method
                        <1>       ;                    will be applied if BL input > 0 !
                        <1>       ;
                        <1>       ;         For BH = 1 ; Initialize (Start) DMA
                        <1>       ;             EAX = 0 (Successful)
                        <1>       ;             If CF = 1, error code is in EAX
                        <1>       ;
                        <1>       ;         For BH = 2 ; Get Current DMA Buffer Offset
                        <1>       ;             EAX = DMA Buffer Offset (in bytes)
                        <1>       ;             ;
                        <1>       ;              AX = DMA buffer offset
                        <1>       ;             EAX bits 16 to 23 = Page register value
                        <1>       ;
                        <1>       ;         For BH = 3 ; Get Current DMA count down value
                        <1>       ;             EAX = Count down value (remain bytes)
                        <1>       ;
                        <1>       ;         For BH = 4 ; Get Current DMA channel (in progress)
                        <1>       ;             EAX = DMA channel number (0 to 7)
                        <1>       ;               AH = 0 if the owner is the caller process
                        <1>       ;               AH > 0 if the dma channel is in use by
                        <1>       ;                      another user/process
                        <1>       ;             EAX = -1 (0FFFFFFFFh)
                        <1>       ;                  if DMA service is not in use
                        <1>       ;                     (stopped or not initialized/started)
                        <1>       ;
                        <1>       ;         For BH = 5 ; Get System's Default DMA Buff Addr
                        <1>       ;             EAX = Default DMA Buffer Address (Physical)
                        <1>       ;                = offset 'sb16_dma_buffer:'
                        <1>       ;             ECX = Buffer size
                        <1>       ;                = 65536
                        <1>       ;
                        <1>       ;         For BH = 6 ; Get Current DMA Buffer Address
                        <1>       ;             EAX = Current DMA buffer address (Physical)
                        <1>       ;             ECX = Current DMA buffer size (setting value)
                        <1>       ;             Note: These values are for current dma channel
                        <1>       ;                   settings for the user/process
                        <1>       ;             ** For now (for current TRDOS 386 version)
                        <1>       ;               only one user/process can use only one
                        <1>       ;               dma channel & one dma buffer at same time
                        <1>       ;               (no multi tasking on DMA service) !!! **
                        <1>       ;             (Once, current DMA user must stop it's own DMA
                        <1>       ;              DMA service, than another user/program
                        <1>       ;              can use DMA service with same dma channel
                        <1>       ;              or with another DMA channel.)
                        <1>       ;
                        <1>       ;         For BH = 7 ; Stop DMA service (for current user
                        <1>       ;             and current DMA channel)
                        <1>       ;             EAX = 0 ; successful
                        <1>       ;             CF = 1 & EAX > 0 (= -1) -> Error
                        <1>
0001036D 80FF07         <1>       cmp   bh, 7
00010370 7612           <1>       jna   short sysdma_0
                        <1>
                        <1> sysdma_err:
00010372 31C0           <1>       xor   eax, eax
```

```
48094 00010374 48                  <1>       dec    eax ; -1
48095                              <1> sysdma_perm_err:
48096 00010375 A3[64030300]        <1>       mov    [u.r0], eax
48097 0001037A A3[C8030300]        <1>       mov    [u.error], eax ; DMA service error !
48098 0001037F E939C1FFFF          <1>       jmp    error
48099                              <1>
48100                              <1> sysdma_0:
48101 00010384 08FF                <1>       or     bh, bh
48102 00010386 0F85BA000000        <1>       jnz    sysdma_1
48103                              <1>
48104 0001038C 20DB                <1>       and    bl, bl
48105 0001038E 7416                <1>       jz     short sysdma_01
48106                              <1>
48107                              <1>       ; redirect system call to 'sysalloc'
48108 00010390 89D3                <1>       mov    ebx, edx ; virtual address of DMA buffer
48109                              <1>       ;ecx = Buffer size in bytes
48110                              <1>       ; DMA buffer address <= 16MB upper limit
48111 00010392 BA00000001          <1>       mov    edx, 1024*1024*16 ; 16MB limit for DMA buff
48112                              <1>
48113 00010397 C705[E8690100]FFFF- <1>       mov    dword [dma_addr], 0FFFFFFFFh ; -1
48114 0001039F FFFF                <1>
48115                              <1>
48116 000103A1 E9DBECFFFF          <1>       jmp    sysalloc
48117                              <1>
48118                              <1> sysdma_01:
48119 000103A6 B8[00000200]        <1>       mov    eax, sb16_dma_buffer
48120                              <1>
48121 000103AB 803D[A5650100]01    <1>       cmp    byte [audio_device], 1
48122 000103B2 722A                <1>       jb     short sysdma_03
48123                              <1>
48124 000103B4 3B05[C4650100]      <1>       cmp    eax, [audio_dma_buff]
48125 000103BA 7507                <1>       jne    short sysdma_02
48126                              <1>
48127                              <1> sysdma_0_err:
48128 000103BC B80B000000          <1>       mov    eax, ERR_PERM_DENIED
48129 000103C1 EBB2                <1>       jmp    short sysdma_perm_err
48130                              <1>
48131                              <1> sysdma_02:
48132                              <1>       ; Only one user is permitted for audio/dma functions
48133                              <1>
48134 000103C3 833D[C4650100]00    <1>       cmp    dword [audio_dma_buff], 0
48135 000103CA 7612                <1>       jna    short sysdma_03
48136                              <1>
48137 000103CC 8A1D[CD650100]      <1>       mov    bl, [audio_user]
48138 000103D2 08DB                <1>       or     bl, bl
48139 000103D4 7408                <1>       jz     short sysdma_03
48140                              <1>
48141 000103D6 3A1D[B3030300]      <1>       cmp    bl, [u.uno]
48142 000103DC 75DE                <1>       jne    short sysdma_0_err
48143                              <1>
48144                              <1> sysdma_03:
48145 000103DE 8A1D[E5690100]      <1>       mov    bl, [dma_user]
48146 000103E4 20DB                <1>       and    bl, bl
48147 000103E6 750E                <1>       jnz    short sysdma_04
48148                              <1>
48149 000103E8 8A1D[B3030300]      <1>       mov    bl, [u.uno]
48150 000103EE 881D[E5690100]      <1>       mov    [dma_user], bl
48151                              <1>
48152 000103F4 EB15                <1>       jmp    short sysdma_05
48153                              <1>
48154                              <1> sysdma_04:
48155 000103F6 8B35[E8690100]      <1>       mov    esi, [dma_addr]
48156 000103FC 21F6                <1>       and    esi, esi
48157 000103FE 740B                <1>       jz     short sysdma_05
48158                              <1>
48159 00010400 46                  <1>       inc    esi ; -1 -> 0
48160 00010401 7408                <1>       jz     short sysdma_05
48161                              <1>
48162 00010403 3A1D[B3030300]      <1>       cmp    bl, [u.uno]
48163 00010409 75B1                <1>       jne    short sysdma_0_err
48164                              <1>
48165                              <1> sysdma_05:
48166                              <1>       ; edx = virtual address (user's buffer address)
48167                              <1>       ;
48168 0001040B 81F900000100        <1>       cmp    ecx, 65536   ; byte count (buffer size)
48169 00010411 0F875BFFFFFF        <1>       ja     sysdma_err
48170                              <1>       ;
48171 00010417 81C1FF0F0000        <1>       add    ecx, PAGE_SIZE-1 ; 4095
48172 0001041D 6681E100F0          <1>       and    cx, ~PAGE_OFF ; not 4095
48173                              <1>       ;cmp   ecx, 65536
48174                              <1>       ;ja    sysdma_err ;
48175 00010422 51                  <1>       push   ecx    ; buffer size (allocated pages * 4096)
48176 00010423 50                  <1>       push   eax    ; offset sb16_dma_buffer
48177 00010424 89D3                <1>       mov    ebx, edx
48178 00010426 C1E90C              <1>       shr    ecx, 12 ; byte count to page count
48179                              <1>       ; eax = physical address of (audio) dma buffer
48180                              <1>       ; ebx = virtual address of (audio) dma buffer (user's pgdir)
48181                              <1>       ; ecx = page count (>0)
48182 00010429 E87352FFFF          <1>       call   direct_memory_access
48183 0001042E 58                  <1>       pop    eax
48184 0001042F 59                  <1>       pop    ecx
48185 00010430 0F823CFFFFFF        <1>       jc     sysdma_err
48186                              <1>
48187 00010436 A3[E8690100]        <1>       mov    [dma_addr], eax
48188 0001043B 890D[EC690100]      <1>       mov    [dma_size], ecx ; dma buffer size (in bytes)
48189                              <1>
48190                              <1>       ;mov   [u.r0], eax ; DMA Buffer Address (Physical)
48191                              <1>
48192                              <1>       ;mov   ebp, [u.usp]  ; ebp points to user's registers
48193                              <1>       ;mov   [ebp+24], ecx ; return to user with ecx value
48194                              <1>
48195                              <1>       ;jmp   sysret
48196                              <1>
```

```
48197                                  <1>       ; 28/08/2017
48198 00010441 E9C4000000              <1>       jmp    sysdma_51
48199                                  <1>
48200                                  <1> sysdma_1:
48201 00010446 80FF01                  <1>       cmp    bh, 1
48202 00010449 0F87A6000000            <1>       ja     sysdma_5
48203                                  <1>
48204 0001044F F6C340                  <1>       test   bl, 40h      ; record (read) mode -BL, bit 6-
48205 00010452 0F851AFFFFFF            <1>       jnz    sysdma_err ; not ready yet!
48206                                  <1>
48207 00010458 A1[E8690100]            <1>       mov    eax, [dma_addr] ; physical address of dma buffer
48208 0001045D 21C0                    <1>       and    eax, eax
48209 0001045F 0F840DFFFFFF            <1>       jz     sysdma_err
48210                                  <1>
48211 00010465 09D2                    <1>       or     edx, edx
48212 00010467 7504                    <1>       jnz    short sysdma_11
48213                                  <1>
48214 00010469 89C2                    <1>       mov    edx, eax
48215 0001046B EB08                    <1>       jmp    short sysdma_12
48216                                  <1> sysdma_11:
48217 0001046D 39C2                    <1>       cmp    edx, eax
48218 0001046F 0F82FDFEFFFF            <1>       jb     sysdma_err
48219                                  <1> sysdma_12:
48220 00010475 21C9                    <1>       and    ecx, ecx
48221 00010477 7508                    <1>       jnz    short sysdma_13
48222                                  <1>
48223 00010479 8B0D[EC690100]          <1>       mov    ecx, [dma_size]
48224 0001047F EB0C                    <1>       jmp    short sysdma_14
48225                                  <1> sysdma_13:
48226 00010481 3B0D[EC690100]          <1>       cmp    ecx, [dma_size]
48227 00010487 0F87E5FEFFFF            <1>       ja     sysdma_err
48228                                  <1> sysdma_14:
48229 0001048D 89C6                    <1>       mov    esi, eax
48230 0001048F 0335[EC690100]          <1>       add    esi, [dma_size]
48231                                  <1>
48232 00010495 89D0                    <1>       mov    eax, edx
48233 00010497 01C8                    <1>       add    eax, ecx
48234 00010499 0F82D3FEFFFF            <1>       jc     sysdma_err ; 02/09/2017
48235                                  <1>
48236 0001049F 39F0                    <1>       cmp    eax, esi
48237 000104A1 0F87CBFEFFFF            <1>       ja     sysdma_err
48238                                  <1>
48239 000104A7 8B3D[C4650100]          <1>       mov    edi, [audio_dma_buff]
48240 000104AD 8B35[E8690100]          <1>       mov    esi, [dma_addr]
48241                                  <1>
48242 000104B3 09FF                    <1>       or     edi, edi
48243 000104B5 7424                    <1>       jz     short sysdma_16
48244                                  <1>
48245 000104B7 803D[A5650100]01        <1>       cmp    byte [audio_device], 1
48246 000104BE 7208                    <1>       jb     short sysdma_15
48247                                  <1>
48248                                  <1>       ; Sound Blaster 16
48249 000104C0 39FE                    <1>       cmp    esi, edi
48250 000104C2 0F84F4FEFFFF            <1>       je     sysdma_0_err ; permmission denied !
48251                                  <1>
48252                                  <1> sysdma_15:
48253 000104C8 C605[E7690100]48        <1>       mov    byte [dma_mode], 48h ; single mode playback
48254                                  <1>
48255 000104CF F6C380                  <1>       test   bl, 80h ; DMA mode - BL, bit 7, auto init -
48256 000104D2 7407                    <1>       jz     short sysdma_16
48257                                  <1>       ; Auto initialized playback (write) mode
48258 000104D4 8005[E7690100]10        <1>       add    byte [dma_mode], 10h ; = 58h
48259                                  <1> sysdma_16:
48260 000104DB 80E307                  <1>       and    bl, 07h
48261 000104DE 881D[E6690100]          <1>       mov    [dma_channel], bl
48262 000104E4 8915[F0690100]          <1>       mov    [dma_start], edx
48263 000104EA 890D[F4690100]          <1>       mov    [dma_count], ecx
48264                                  <1>
48265                                  <1>       ; 28/08/2017
48266                                  <1>       ;call  dma_init
48267                                  <1>       ;jmp   sysret
48268 000104F0 E94B010000              <1>       jmp    dma_init
48269                                  <1>
48270                                  <1> sysdma_5:
48271 000104F5 80FF05                  <1>       cmp    bh, 5
48272 000104F8 7223                    <1>       jb     short sysdma_3
48273 000104FA 0F87CE000000            <1>       ja     sysdma_6
48274                                  <1>
48275                                  <1>       ; Get the system's default dma buffer addr and size
48276 00010500 B8[00000200]            <1>       mov    eax, sb16_dma_buffer
48277 00010505 B900000100              <1>       mov    ecx, 65536 ; Buffer size in bytes
48278                                  <1>
48279                                  <1> sysdma_51:
48280                                  <1>       ; 0 = there is not a dma buffer (in use or available)
48281 0001050A A3[64030300]            <1>       mov    [u.r0], eax
48282                                  <1>
48283 0001050F 8B2D[60030300]          <1>       mov    ebp, [u.usp]  ; ebp points to user's registers
48284 00010515 894D18                  <1>       mov    [ebp+24], ecx ; return to user with ecx value
48285                                  <1>
48286 00010518 E9C0BFFFFF              <1>       jmp    sysret
48287                                  <1>
48288                                  <1> sysdma_3:
48289 0001051D 80FF03                  <1>       cmp    bh, 3
48290 00010520 7231                    <1>       jb     short sysdma_2
48291 00010522 776B                    <1>       ja     short sysdma_4
48292                                  <1>
48293                                  <1>       ; Get current dma count down value (remain bytes)
48294                                  <1>       ; 28/08/2017
48295 00010524 0FB635[E6690100]        <1>       movzx  esi, byte [dma_channel]
48296 0001052B 0FB696[40100100]        <1>       movzx  edx, byte [dma_flip+esi]
48297 00010532 EE                      <1>       out    dx, al              ; flip-flop clear
48298 00010533 8A96[20100100]          <1>       mov    dl, [dma_cnt+esi] ; dma count register addr
48299 00010539 EC                      <1>       in     al, dx
```

```
48300 0001053A 0FB6D8              <1>       movzx  ebx, al
48301 0001053D EC                  <1>       in     al, dx
48302 0001053E 88C7                <1>       mov     bh, al
48303                              <1>
48304 00010540 6683FE04            <1>       cmp    si, 4 ; channel number ?
48305 00010544 7202                <1>       jb     short sysdma_31 ; 8 bit dma channel
48306                              <1>
48307 00010546 D1E3                <1>       shl    ebx, 1 ; word count to byte count
48308                              <1>
48309                              <1> sysdma_31:
48310 00010548 891D[64030300]      <1>       mov    [u.r0], ebx
48311                              <1>
48312 0001054E E98ABFFFFF          <1>       jmp    sysret
48313                              <1>
48314                              <1> sysdma_2:
48315                              <1>       ; Get current dma buffer offset (& page)
48316                              <1>       ; 28/08/2017
48317 00010553 0FB635[E6690100]    <1>       movzx  esi, byte [dma_channel]
48318 0001055A 0FB696[40100100]    <1>       movzx  edx, byte [dma_flip+esi]
48319 00010561 EE                  <1>       out    dx, al              ; flip-flop clear
48320 00010562 8A96[18100100]      <1>       mov    dl, [dma_adr+esi]
48321 00010568 EC                  <1>       in     al, dx              ; get dma position
48322 00010569 0FB6D8              <1>       movzx  ebx, al
48323 0001056C EC                  <1>       in     al, dx
48324 0001056D 88C7                <1>       mov    bh, al
48325                              <1>
48326 0001056F 6683FE04            <1>       cmp    si, 4 ; channel number ?
48327 00010573 7202                <1>       jb     short sysdma_21 ; 8 bit dma channel
48328                              <1>
48329 00010575 D1E3                <1>       shl    ebx, 1 ; word offset to byte offset
48330                              <1>
48331                              <1> sysdma_21:
48332 00010577 891D[64030300]      <1>       mov    [u.r0], ebx
48333                              <1>
48334 0001057D 8A96[28100100]      <1>       mov    dl, [dma_page+esi]
48335 00010583 EC                  <1>       in     al, dx              ; get dma page
48336                              <1>
48337                              <1>       ;add   [u.ro+2], al
48338 00010584 0805[66030300]      <1>       or     [u.r0+2], al
48339                              <1>
48340 0001058A E94EBFFFFF          <1>       jmp    sysret
48341                              <1>
48342                              <1> sysdma_4:
48343                              <1>       ; Get current DMA channel number
48344                              <1>       ; 28/08/2017
48345 0001058F 8A25[E5690100]      <1>       mov    ah, [dma_user]
48346 00010595 20E4                <1>       and    ah, ah
48347 00010597 750F                <1>       jnz    short sysdma_42
48348                              <1>
48349                              <1> sysdma_41:
48350                              <1>       ; Not a valid dma channel (in use)
48351 00010599 C705[64030300]FFFF- <1>       mov    dword [u.r0], -1 ; 0FFFFFFFFh
48352 000105A1 FFFF                <1>
48353 000105A3 E935BFFFFF          <1>       jmp    sysret
48354                              <1>
48355                              <1> sysdma_42:
48356 000105A8 8B35[E8690100]      <1>       mov    esi, [dma_addr]
48357 000105AE 21F6                <1>       and    esi, esi
48358 000105B0 74E7                <1>       jz     short sysdma_41
48359                              <1>
48360 000105B2 46                  <1>       inc    esi ; -1 -> 0
48361 000105B3 74E4                <1>       jz     short sysdma_41
48362                              <1>
48363 000105B5 A0[E6690100]        <1>       mov    al, [dma_channel]
48364                              <1>
48365 000105BA 3A25[B3030300]      <1>       cmp    ah, [u.uno]
48366 000105C0 7502                <1>       jne    short sysdma_43
48367                              <1>
48368 000105C2 30E4                <1>       xor    ah, ah ; DMA channel in use by current user
48369                              <1>
48370                              <1> sysdma_43:
48371 000105C4 A3[64030300]        <1>       mov    [u.r0], eax ; AL = dma channel number
48372                              <1>                          ; AH > 0 if the the channel
48373                              <1>                          ; in use by another user/process
48374 000105C9 E90FBFFFFF          <1>       jmp    sysret
48375                              <1>
48376                              <1> sysdma_6:
48377 000105CE 80FF06              <1>       cmp    bh, 6
48378 000105D1 7710                <1>       ja     short sysdma_7
48379                              <1>
48380                              <1>       ; 28/08/2017
48381                              <1>       ; Get current DMA buffer addr and size
48382 000105D3 A1[E8690100]        <1>       mov    eax, [dma_addr] ; dma buffer address
48383 000105D8 8B0D[EC690100]      <1>       mov    ecx, [dma_size] ; dma buffer size (in bytes)
48384                              <1>
48385 000105DE E927FFFFFF          <1>       jmp    sysdma_51
48386                              <1>
48387                              <1> sysdma_7:
48388                              <1>       ; DMA service STOP
48389 000105E3 A0[B3030300]        <1>       mov    al, [u.uno]
48390 000105E8 3A05[E5690100]      <1>       cmp    al, [dma_user]
48391 000105EE 751D                <1>       jne    short sysdma_72
48392                              <1>
48393 000105F0 28C0                <1>       sub    al, al ; 0
48394                              <1>
48395 000105F2 A2[E5690100]        <1>       mov    [dma_user], al ; clear user
48396                              <1>
48397 000105F7 8605[E7690100]      <1>       xchg   al, [dma_mode]
48398 000105FD 20C0                <1>       and    al, al
48399                              <1>       ;jz    short sysdma_err
48400 000105FF 7527                <1>       jnz    short sysdma_73
48401                              <1>
48402                              <1> sysdma_71:
```

```
48403 00010601 31C0                <1>       xor    eax, eax
48404 00010603 A3[64030300]        <1>       mov    [u.r0], eax; 0
48405 00010608 E9D0BEFFFF          <1>       jmp    sysret
48406                              <1>
48407                              <1> sysdma_72:
48408                              <1>       ; 28/08/2017
48409 0001060D 803D[E5690100]00    <1>       cmp    byte [dma_user], 0
48410 00010614 76EB                <1>       jna    short sysdma_71 ; Nothing to do !
48411                              <1>
48412 00010616 833D[E8690100]00    <1>       cmp    dword [dma_addr], 0
48413 0001061D 0F8799FDFFFF        <1>       ja     sysdma_0_err
48414                              <1>
48415 00010623 A2[E5690100]        <1>       mov    [dma_user], al ; reset to current user
48416                              <1>
48417                              <1> sysdma_73:
48418                              <1>       ; 28/08/2017
48419 00010628 0FB635[E6690100]    <1>       movzx  esi, byte [dma_channel]
48420 0001062F 0FB696[30100100]    <1>       movzx  edx, byte [dma_mask+esi]
48421 00010636 A0[E6690100]        <1>       mov    al, [dma_channel]
48422 0001063B 0C04                <1>       or     al, 4
48423 0001063D EE                  <1>       out    dx, al
48424                              <1>
48425 0001063E EBC1                <1>       jmp    short sysdma_71
48426                              <1>
48427                              <1> dma_init:
48428                              <1>       ; 28/08/2017
48429                              <1>       ; 20/08/2017
48430                              <1>       ; DMA initialization
48431                              <1>       ; 14/08/2017
48432                              <1>       ; 03/08/2017, 06/08/2017, 08/08/2017
48433                              <1>       ; 02/07/2017, 13/07/2017, 16/07/2017, 30/07/2017
48434                              <1>       ; (Derived from 'DMA_INIT' procedure in SB16MOD.ASM)
48435                              <1>       ; Modified for TRDOS 386 DMA buffer allocation & initialization !
48436                              <1>
48437 00010640 8B1D[F0690100]      <1>       mov    ebx, [dma_start]
48438 00010646 8B0D[F4690100]      <1>       mov    ecx, [dma_count]
48439                              <1>
48440 0001064C 0FB635[E6690100]    <1>       movzx  esi, byte [dma_channel]
48441                              <1>
48442 00010653 6683FE04            <1>       cmp    si, 4
48443 00010657 7205                <1>       jb     short gdmi1
48444                              <1>       ; 08/08/2017
48445 00010659 66D1E9              <1>       shr    cx, 1 ; word count
48446 0001065C D1EB                <1>       shr    ebx, 1 ; convert byte offset to word offset
48447                              <1> gdmi1:
48448                              <1>       ;mov   [dma_poff], bx ; 08/08/2017
48449 0001065E 6649                <1>       dec    cx                  ; dma size = block size - 1
48450                              <1>
48451 00010660 0FB696[30100100]    <1>       movzx  edx, byte [dma_mask+esi] ; 30/07/2017
48452 00010667 A0[E6690100]        <1>       mov    al, [dma_channel]
48453 0001066C 0C04                <1>       or     al, 4
48454 0001066E EE                  <1>       out    dx, al              ; dma channel mask
48455                              <1>
48456 0001066F 30C0                <1>       xor    al, al ; 0 ; any value ! 08/08/2017
48457 00010671 8A96[40100100]      <1>       mov    dl, [dma_flip+esi]
48458 00010677 EE                  <1>       out    dx, al              ; flip-flop clear
48459                              <1>
48460 00010678 8A96[38100100]      <1>       mov    dl, [dma_mod+esi]
48461 0001067E A0[E6690100]        <1>       mov    al, [dma_channel]  ; 13/07/2017
48462 00010683 2403                <1>       and    al, 3
48463                              <1>       ; 08/08/2017
48464 00010685 0A05[E7690100]      <1>       or     al, [dma_mode] ; 58h    ; dma mode for SB16
48465 0001068B EE                  <1>       out    dx, al
48466                              <1>
48467 0001068C 8A96[18100100]      <1>       mov    dl, [dma_adr+esi]
48468 00010692 88D8                <1>       mov    al, bl
48469 00010694 EE                  <1>       out    dx, al              ; offset low
48470                              <1>
48471 00010695 88F8                <1>       mov    al, bh
48472 00010697 EE                  <1>       out    dx, al              ; offset high
48473                              <1>
48474 00010698 8A96[20100100]      <1>       mov    dl, [dma_cnt+esi]
48475 0001069E 88C8                <1>       mov    al, cl
48476 000106A0 EE                  <1>       out    dx, al              ; size low
48477                              <1>
48478 000106A1 88E8                <1>       mov    al, ch
48479 000106A3 EE                  <1>       out    dx, al              ; size high
48480                              <1>
48481 000106A4 8A96[28100100]      <1>       mov    dl, [dma_page+esi]
48482                              <1>       ; 14/08/2017
48483 000106AA 6683FE04            <1>       cmp    si, 4
48484 000106AE 7305                <1>       jnb    short gdmi2
48485 000106B0 C1EB10              <1>       shr    ebx, 16
48486 000106B3 EB06                <1>       jmp    short gdmi3
48487                              <1> gdmi2:
48488                              <1>       ; 09/08/2017
48489 000106B5 C1EB0F              <1>       shr    ebx, 15       ; complete 16 bit shift
48490 000106B8 80E3FE              <1>       and    bl, 0FEh ; clear bit 0 (not necessary)
48491                              <1> gdmi3:
48492 000106BB 88D8                <1>       mov    al, bl
48493 000106BD EE                  <1>       out    dx, al              ; page
48494                              <1>
48495 000106BE 8A96[30100100]      <1>       mov    dl, [dma_mask+esi]
48496 000106C4 A0[E6690100]        <1>       mov    al, [dma_channel]  ; 13/07/2017
48497 000106C9 2403                <1>       and    al, 3
48498 000106CB EE                  <1>       out    dx, al              ; dma channel unmask
48499                              <1>
48500                              <1>       ;retn
48501                              <1>       ; 28/08/2017
48502 000106CC E90CBEFFFF          <1>       jmp    sysret
48503                              <1>
48504                              <1> otty:
48505                              <1> sret:
```

```
48506                                   <1> ocvt:
48507                                   <1> ctty:
48508                                   <1> cret:
48509                                   <1> ccvt:
48510                                   <1> rtty:
48511                                   <1> wtty:
48512                                   <1> rmem:
48513                                   <1> wmem:
48514                                   <1> rfd:
48515                                   <1> rhd:
48516                                   <1> wfd:
48517                                   <1> whd:
48518                                   <1> rlpt:
48519                                   <1> wlpt:
48520                                   <1> rcvt:
48521                                   <1> xmtt:
48522 000106D1 C3                       <1>       retn
48523                                       %include 'trdosk9.s' ; 04/01/2016
48524                                   <1> ; ****************************************************************************
48525                                   <1> ; TRDOS386.ASM (TRDOS 386 Kernel - v2.0.0) - INITIALIZED DATA : trdosk9.s
48526                                   <1> ; ----------------------------------------------------------------------------
48527                                   <1> ; Last Update: 27/12/2017
48528                                   <1> ; ----------------------------------------------------------------------------
48529                                   <1> ; Beginning: 04/01/2016
48530                                   <1> ; ----------------------------------------------------------------------------
48531                                   <1> ; ----------------------------------------------------------------------------
48532                                   <1> ; Assembler: NASM version 2.11 (trdos386.s)
48533                                   <1> ; ----------------------------------------------------------------------------
48534                                   <1> ; Derived from TRDOS Operating System v1.0 (8086) source code by Erdogan Tan
48535                                   <1> ; TRDOS2.ASM (09/11/2011)
48536                                   <1> ; ****************************************************************************
48537                                   <1> ; DRV_INIT.ASM [26/09/2009] Last Update: 07/08/2011
48538                                   <1> ; MAINPROG.ASM [17/01/2004] Last Update: 09/11/2011
48539                                   <1> ; CMD_INTR.ASM [29/01/2005] Last Update: 09/11/2011
48540                                   <1> ; FILE.ASM [29/10/2009] Last Update: 09/10/2011
48541                                   <1>
48542                                   <1> ; 12/02/2016
48543                                   <1> Last_DOS_DiskNo:
48544 000106D2 01                       <1>             db 1 ; A: = 0 & B: = 1
48545                                   <1>
48546                                   <1> Restore_CDIR:
48547 000106D3 FF                       <1>             db 0FFh ; Initial value -> any number except 0
48548                                   <1>
48549                                   <1> msg_CRLF_temp:
48550 000106D4 070D0A00                 <1>             db 07h, 0Dh, 0Ah, 0
48551                                   <1>
48552                                   <1> Magic_Bytes:
48553 000106D8 04                       <1>             db 4
48554 000106D9 01                       <1>             db 1
48555                                   <1> mainprog_Version:
48556 000106DA 07                       <1>             db 7
48557 000106DB 5B5452444F535D204D-      <1>             db "[TRDOS] Main Program v2.0.271217"
48558 000106E4 61696E2050726F6772-      <1>
48559 000106ED 616D2076322E302E32-      <1>
48560 000106F6 3731323137               <1>
48561 000106FB 0D0A                     <1>             db 0Dh, 0Ah
48562 000106FD 286329204572646F67-      <1>             db "(c) Erdogan Tan 2005-2017"
48563 00010706 616E2054616E203230-      <1>
48564 0001070F 30352D32303137           <1>
48565 00010716 0D0A00                   <1>             db 0Dh, 0Ah, 0
48566                                   <1>
48567                                   <1> MainProgCfgFile: ; 14/04/2016
48568 00010719 4D41494E50524F472E-      <1>             db "MAINPROG.CFG", 0
48569 00010722 43464700                 <1>
48570                                   <1>
48571                                   <1> TRDOSPromptLabel:
48572 00010726 5452444F53               <1>             db "TRDOS"
48573 0001072B 00                       <1>             db 0
48574 0001072C 00<rept>                 <1>                times 5 db 0
48575 00010731 00                       <1>             db 0
48576                                   <1>
48577                                   <1> ; INTERNAL COMMANDS
48578                                   <1> Command_List:
48579 00010732 44495200                 <1> Cmd_Dir:    db "DIR", 0
48580 00010736 434400                   <1> Cmd_Cd:           db "CD", 0
48581 00010739 433A00                   <1> Cmd_Drive:  db "C:", 0
48582 0001073C 56455200                 <1> Cmd_Ver:    db "VER", 0
48583 00010740 4558495400               <1> Cmd_Exit:   db "EXIT", 0
48584 00010745 50524F4D505400           <1> Cmd_Prompt: db "PROMPT", 0
48585 0001074C 564F4C554D4500           <1> Cmd_Volume: db "VOLUME", 0
48586 00010753 4C4F4E474E414D4500       <1> Cmd_LongName:     db "LONGNAME", 0
48587 0001075C 4441544500               <1> Cmd_Date:   db "DATE", 0
48588 00010761 54494D4500               <1> Cmd_Time:   db "TIME", 0
48589 00010766 52554E00                 <1> Cmd_Run:    db "RUN", 0
48590 0001076A 53455400                 <1> Cmd_Set:    db "SET", 0
48591 0001076E 434C5300                 <1> Cmd_Cls:    db "CLS", 0
48592 00010772 53484F5700               <1> Cmd_Show:   db "SHOW", 0
48593 00010777 44454C00                 <1> Cmd_Del:    db "DEL", 0
48594 0001077B 41545452494200           <1> Cmd_Attrib: db "ATTRIB", 0
48595 00010782 52454E414D4500           <1> Cmd_Rename: db "RENAME", 0
48596 00010789 524D44495200             <1> Cmd_Rmdir:  db "RMDIR", 0
48597 0001078F 4D4B44495200             <1> Cmd_Mkdir:  db "MKDIR", 0
48598 00010795 434F505900               <1> Cmd_Copy:   db "COPY", 0
48599 0001079A 4D4F564500               <1> Cmd_Move:   db "MOVE", 0
48600 0001079F 5041544800               <1> Cmd_Path:   db "PATH", 0
48601 000107A4 4D454D00                 <1> Cmd_Mem:    db "MEM", 0
48602 000107A8 00                       <1>             db 0
48603 000107A9 46494E4400               <1> Cmd_Find:   db "FIND", 0
48604 000107AE 4543484F00               <1> Cmd_Echo:   db "ECHO", 0
48605 000107B3 2A00                     <1> Cmd_Remark: db "*", 0
48606 000107B5 3F00                     <1> Cmd_Help:   db "?", 0
48607 000107B7 44455649434500           <1> Cmd_Device: db "DEVICE", 0
48608 000107BE 4445564C49535400         <1> Cmd_DevList:db "DEVLIST", 0
```

```
48609 000107C6 434844495200        <1> Cmd_Chdir:   db "CHDIR", 0
48610 000107CC 4245455000          <1> Cmd_Beep:    db "BEEP", 0
48611                              <1>
48612 000107D1 00                  <1>              db 0
48613                              <1>
48614                              <1> ; 15/02/2016 (FILE.ASM, 09/10/2011)
48615                              <1> invalid_fname_chars:
48616 000107D2 222728292A2B2C2F    <1>              db 22h, 27h, 28h, 29h, 2Ah, 2Bh, 2Ch, 2Fh
48617 000107DA 3A3B3C3D3E3F40     <1>              db 3Ah, 3Bh, 3Ch, 3Dh, 3Eh, 3Fh, 40h
48618 000107E1 5B5C5D5E60          <1>              db 5Bh, 5Ch, 5Dh, 5Eh, 60h
48619                              <1> sizeInvFnChars  equ ($ - invalid_fname_chars)
48620                              <1> ;
48621                              <1>
48622                              <1> Msg_Enter_Date:
48623 000107E6 456E746572206E6577- <1>                  db 'Enter new date (dd-mm-yy): '
48624 000107EF 206461746520286464- <1>
48625 000107F8 2D6D6D2D7979293A20  <1>
48626 00010801 00                  <1>                  db 0
48627                              <1> Msg_Show_Date:
48628 00010802 43757272656E742064- <1>                  db   'Current date is '
48629 0001080B 61746520697320      <1>
48630 00010812 30                  <1> Day:             db   '0'
48631 00010813 30                  <1>              db   '0'
48632 00010814 2F                  <1>                  db   '/'
48633 00010815 30                  <1> Month:           db   '0'
48634 00010816 30                  <1>              db   '0'
48635 00010817 2F                  <1>                  db   '/'
48636 00010818 30                  <1> Century:         db   '0'
48637 00010819 30                  <1>                  db   '0'
48638 0001081A 30                  <1> Year:            db   '0'
48639 0001081B 30                  <1>              db   '0'
48640 0001081C 0D0A00              <1>                  db   0Dh, 0Ah, 0
48641                              <1>
48642                              <1> Msg_Enter_Time:
48643 0001081F 456E746572206E6577- <1>              db 'Enter new time: '
48644 00010828 2074696D653A20      <1>
48645 0001082F 00                  <1>              db 0
48646                              <1> Msg_Show_Time:
48647 00010830 43757272656E742074- <1>              db   'Current time is '
48648 00010839 696D6520697320      <1>
48649 00010840 30                  <1> Hour:            db   '0'
48650 00010841 30                  <1>              db   '0'
48651 00010842 3A                  <1>              db   ':'
48652 00010843 30                  <1> Minute:          db   '0'
48653 00010844 30                  <1>              db   '0'
48654 00010845 3A                  <1>              db   ':'
48655 00010846 30                  <1> Second:          db   '0'
48656 00010847 30                  <1>              db   '0'
48657 00010848 0D0A00              <1>              db   0Dh, 0Ah, 0
48658                              <1>
48659                              <1> ;VolSize_Unit1:   dd 0
48660                              <1> ;VolSize_Unit2:   dd 0
48661                              <1>
48662                              <1> VolSize_KiloBytes:
48663 0001084B 206B696C6F62797465- <1>              db " kilobytes", 0Dh, 0Ah, 0
48664 00010854 730D0A00            <1>
48665                              <1> VolSize_Bytes:
48666 00010858 2062797465730D0A00  <1>              db " bytes", 0Dh, 0Ah, 0
48667                              <1> Volume_in_drive:
48668 00010861 0D0A                <1>              db 0Dh, 0Ah
48669                              <1> Vol_FS_Name:
48670 00010863 54522046533120      <1>              db "TR FS1 "
48671 0001086A 566F6C756D6520696E- <1>              db "Volume in drive "
48672 00010873 20647269766520      <1>
48673 0001087A 30                  <1> Vol_Drv_Name:   db 30h
48674 0001087B 3A                  <1>              db ":"
48675 0001087C 20697320            <1>              db " is "
48676 00010880 0D0A00              <1>              db 0Dh, 0Ah, 0
48677                              <1> Dir_Drive_Str:
48678 00010883 54522D444F53204472- <1>                  db "TR-DOS Drive "
48679 0001088C 69766520            <1>
48680                              <1> Dir_Drive_Name:
48681 00010890 303A                <1>                  db "0:"
48682 00010892 0D0A                <1>                  db  0Dh, 0Ah
48683                              <1> Vol_Str_Header:
48684 00010894 566F6C756D65204E61- <1>                  db "Volume Name: "
48685 0001089D 6D653A20            <1>
48686                              <1> Vol_Name:
48687 000108A1 00<rept>            <1>              times 64 db 0
48688 000108E1 00                  <1>              db 0
48689                              <1> Vol_Serial_Header:
48690 000108E2 0D0A                <1>              db 0Dh, 0Ah
48691 000108E4 566F6C756D65205365- <1>              db "Volume Serial No: "
48692 000108ED 7269616C204E6F3A20  <1>
48693                              <1> Vol_Serial1:
48694 000108F6 30303030            <1>              db "0000"
48695 000108FA 2D                  <1>              db "-"
48696                              <1> Vol_Serial2:
48697 000108FB 30303030            <1>              db "0000"
48698 000108FF 0D0A00              <1>              db 0Dh, 0Ah, 0
48699                              <1>
48700                              <1> ;Vol_Tot_Sec_Str_Start:
48701                              <1> ;            dd 0
48702                              <1> Vol_Total_Sector_Header:
48703 00010902 0D0A                <1>              db 0Dh, 0Ah
48704 00010904 566F6C756D65205369- <1>              db "Volume Size : ", 0
48705 0001090D 7A65203A2000        <1>
48706                              <1> ;Vol_Tot_Sec_Str:
48707                              <1> ;            db "0000000000"
48708                              <1> ;Vol_Tot_Sec_Str_End:
48709                              <1> ;            db 0
48710                              <1> ;Vol_Free_Sectors_Str_Start:
48711                              <1> ;            dd 0
```

```
48712                                   <1> Vol_Free_Sectors_Header:
48713 00010913 467265652053706163-      <1>             db "Free Space  : ", 0
48714 0001091C 6520203A2000             <1>
48715                                   <1> ;Vol_Free_Sectors_Str:
48716                                   <1> ;           db "0000000000"
48717                                   <1> ;Vol_Free_Sectors_Str_End:
48718                                   <1> ;           db 0
48719                                   <1>
48720                                   <1> Dir_Str_Header:
48721 00010922 4469726563746F7279-      <1>                 db "Directory: "
48722 0001092B 3A20                     <1>
48723 0001092D 2F                       <1> Dir_Str_Root:   db "/"
48724 0001092E 00<rept>                 <1> Dir_Str:        times 64 db 0
48725 0001096E 00000000                 <1>                 dd 0
48726 00010972 00                       <1>                 db 0
48727                                   <1>
48728                                   <1> Msg_Bad_Command:
48729 00010973 42616420636F6D6D61-      <1>                 db "Bad command or file name!"
48730 0001097C 6E64206F722066696C-      <1>
48731 00010985 65206E616D6521           <1>
48732 0001098C 0D0A00                   <1>                 db 0Dh, 0Ah, 0
48733                                   <1>
48734                                   <1> msgl_drv_not_ready:
48735 0001098F 070D0A                   <1>             db 07h, 0Dh, 0Ah
48736                                   <1>
48737                                   <1> ; CMD_INTR.ASM - 09/11/2011 - Messages
48738                                   <1>
48739                                   <1> Msg_Not_Ready_Read_Err:
48740 00010992 4472697665206E6F74-      <1>                 db "Drive not ready or read error!"
48741 0001099B 207265616479206F72-      <1>
48742 000109A4 2072656164657272-        <1>
48743 000109AD 6F7221                   <1>
48744 000109B0 0D0A00                   <1>                 db 0Dh, 0Ah, 0
48745                                   <1>
48746                                   <1> Msg_Not_Ready_Write_Err:
48747 000109B3 4472697665206E6F74-      <1>                 db "Drive not ready or write error!"
48748 000109BC 207265616479206F72-      <1>
48749 000109C5 2077726974657272-        <1>
48750 000109CE 726F7221                 <1>
48751 000109D2 0D0A00                   <1>                 db 0Dh, 0Ah, 0
48752                                   <1>
48753                                   <1> Msg_Dir_Not_Found:
48754 000109D5 4469726563746F7279-      <1>                 db "Directory not found!"
48755 000109DE 206E6F7420666F756E-      <1>
48756 000109E7 6421                     <1>
48757 000109E9 0D0A00                   <1>                 db 0Dh, 0Ah, 0
48758                                   <1>
48759                                   <1> Msg_File_Not_Found:
48760 000109EC 46696C65206E6F7420-      <1>                 db "File not found!"
48761 000109F5 666F756E6421             <1>
48762 000109FB 0D0A00                   <1>                 db 0Dh, 0Ah, 0
48763                                   <1>
48764                                   <1> Msg_File_Directory_Not_Found:
48765 000109FE 46696C65206F722064-      <1>                 db "File or directory not found!"
48766 00010A07 69726563746F727920-      <1>
48767 00010A10 6E6F7420666F756E64-      <1>
48768 00010A19 21                       <1>
48769 00010A1A 0D0A00                   <1>                 db 0Dh, 0Ah, 0
48770                                   <1>
48771                                   <1> Msg_LongName_Not_Found:
48772 00010A1D 4C6F6E67206E616D65-      <1>                 db "Long name not found!"
48773 00010A26 206E6F7420666F756E-      <1>
48774 00010A2F 6421                     <1>
48775 00010A31 0D0A00                   <1>                 db 0Dh, 0Ah, 0
48776                                   <1>
48777                                   <1> beep_Insufficient_Memory: ; 20/02/2017
48778 00010A34 0D0A                     <1>             db 0Dh, 0Ah
48779 00010A36 07                       <1>             db 07h
48780                                   <1> Msg_Insufficient_Memory:
48781 00010A37 496E73756666696369-      <1>                 db "Insufficient memory!"
48782 00010A40 656E74206D656D6F72-      <1>
48783 00010A49 7921                     <1>
48784 00010A4B 0D0A00                   <1>                 db 0Dh, 0Ah, 0
48785                                   <1>
48786                                   <1> Msg_Error_Code:
48787 00010A4E 436F6D6D616E642066-      <1>                 db 'Command failed! Error code : '
48788 00010A57 61696C656421204572-      <1>
48789 00010A60 726F7220636F646520-      <1>
48790 00010A69 3A20                     <1>
48791 00010A6B 303068                   <1> error_code_hex: db '00h'
48792 00010A6E 0A0A00                   <1>                 db 0Ah, 0Ah, 0
48793                                   <1>
48794 00010A71 90                       <1> align 2
48795                                   <1>
48796                                   <1> ; 10/02/2016
48797                                   <1> ; DIR.ASM - 09/10/2011
48798                                   <1>
48799 00010A72 3C4449523E20202020-      <1> Type_Dir:       db '<DIR>     ' ; 10 bytes
48800 00010A7B 20                       <1>
48801                                   <1>
48802                                   <1> File_Name:
48803 00010A7C 20<rept>                 <1>                 times 12 db 20h
48804 00010A88 20                       <1>             db 20h
48805                                   <1> Dir_Or_FileSize:
48806 00010A89 20<rept>                 <1>                 times 10 db 20h
48807 00010A93 20                       <1>             db 20h
48808                                   <1> File_Attribute:
48809 00010A94 20202020                 <1>             dd 20202020h
48810 00010A98 20                       <1>             db 20h
48811                                   <1> File_Day:
48812 00010A99 3030                     <1>                 db '0','0'
48813 00010A9B 2F                       <1>             db '/'
48814                                   <1> File_Month:
```

```
48815 00010A9C 3030                <1>                 db '0','0'
48816 00010A9E 2F                  <1>             db '/'
48817                              <1> File_Year:
48818 00010A9F 30303030            <1>                 db '0','0','0','0'
48819 00010AA3 20                  <1>             db 20h
48820                              <1> File_Hour:
48821 00010AA4 3030                <1>                 db '0','0'
48822 00010AA6 3A                  <1>             db ':'
48823                              <1> File_Minute:
48824 00010AA7 3030                <1>                 db '0','0'
48825 00010AA9 00                  <1>             db 0
48826                              <1>
48827                              <1> Decimal_File_Count_Header:
48828 00010AAA 0D0A                <1>             db 0Dh, 0Ah
48829                              <1> Decimal_File_Count:
48830 00010AAC 00<rept>            <1>             times 6 db 0
48831                              <1>
48832 00010AB2 2066696C6528732920- <1> str_files:   db " file(s) & "
48833 00010ABB 2620                <1>
48834                              <1> Decimal_Dir_Count:
48835 00010ABD 00<rept>            <1>             times 6 db 0
48836                              <1> str_dirs:
48837 00010AC3 206469726563746F72- <1>             db " directory(s) "
48838 00010ACC 7928732920          <1>
48839 00010AD1 0D0A00              <1>             db 0Dh, 0Ah, 0
48840                              <1>
48841 00010AD4 206279746528732920- <1> str_bytes:   db " byte(s) in file(s)"
48842 00010ADD 696E2066696C652873- <1>
48843 00010AE6 29                  <1>
48844 00010AE7 0D0A00              <1>             db 0Dh, 0Ah, 0
48845                              <1>
48846                              <1> ; CMD_INTR.ASM - 09/11/2011
48847                              <1> ; 07/10/2010
48848                              <1> Msg_invalid_name_chars:
48849 00010AEA 496E76616C69642066- <1>                 db "Invalid file or directory name characters!"
48850 00010AF3 696C65206F72206469- <1>
48851 00010AFC 726563746F7279206E- <1>
48852 00010B05 616D65206368617261- <1>
48853 00010B0E 637465727321        <1>
48854 00010B14 0D0A00              <1>             db 0Dh, 0Ah, 0
48855                              <1> ; 21/02/2016
48856 00010B17 46696C65206F722064- <1> Msg_Name_Exists: db "File or directory name exists!"
48857 00010B20 69726563746F727920- <1>
48858 00010B29 6E616D652065786973- <1>
48859 00010B32 747321              <1>
48860 00010B35 0D0A00              <1>                 db 0Dh, 0Ah, 0
48861                              <1> Msg_DoYouWantMkdir:
48862 00010B38 446F20796F75207761- <1>                 db "Do you want to make directory ", 0
48863 00010B41 6E7420746F206D616B- <1>
48864 00010B4A 65206469726563746F- <1>
48865 00010B53 72792000            <1>
48866 00010B57 2028592F4E29203F20- <1> Msg_YesNo:      db " (Y/N) ? ", 0
48867 00010B60 00                  <1>
48868 00010B61 000D0A00            <1> Y_N_nextline:      db 0, 0Dh, 0Ah, 0
48869 00010B65 4F4B2E0D0A00        <1> Msg_OK:            db "OK.", 0Dh, 0Ah, 0
48870                              <1>
48871                              <1> ; 27/02/2016
48872                              <1> Msg_DoYouWantRmDir:
48873 00010B6B 446F20796F75207761- <1>                 db "Do you want to delete directory ", 0
48874 00010B74 6E7420746F2064656C- <1>
48875 00010B7D 657465206469726563- <1>
48876 00010B86 746F72792000        <1>
48877                              <1> Msg_Dir_Not_Empty:
48878 00010B8C 4469726563746F7279- <1>                 db "Directory not empty!"
48879 00010B95 206E6F7420656D7074- <1>
48880 00010B9E 7921                <1>
48881 00010BA0 0D0A00              <1>                 db 0Dh, 0Ah, 0
48882                              <1>
48883                              <1> Msg_DoYouWantDelete:
48884 00010BA3 446F20796F75207761- <1>                 db "Do you want to delete file ",0
48885 00010BAC 6E7420746F2064656C- <1>
48886 00010BB5 6574652066696C6520- <1>
48887 00010BBE 00                  <1>
48888                              <1>
48889 00010BBF 44656C657465642E2E- <1> Msg_Deleted:    db "Deleted...", 0Dh, 0Ah, 0
48890 00010BC8 2E0D0A00            <1>
48891                              <1>
48892                              <1> Msg_Permission_Denied:
48893 00010BCC 07                  <1>                 db 7
48894 00010BCD 5065726D697373696F- <1>                 db "Permission denied!", 0Dh, 0Ah, 0
48895 00010BD6 6E2064656E69656421- <1>
48896 00010BDF 0D0A00              <1>
48897                              <1>
48898                              <1> ; 04/03/2016
48899 00010BE2 4E657720            <1> Msg_New:        db "New "
48900 00010BE6 00                  <1>                 db 0
48901                              <1> Str_Attributes:
48902 00010BE7 417474726962757465- <1>                 db "Attributes : "
48903 00010BF0 73203A20            <1>
48904 00010BF4 4E4F524D414C        <1> Attr_Chars:     db "NORMAL"
48905 00010BFA 00                  <1>                 db 0
48906                              <1>
48907                              <1> ; 06/03/2016
48908                              <1> ; CMD_INTR.ASM - 16/11/2010
48909                              <1> Msg_DoYouWantRename:
48910 00010BFB 446F20796F75207761- <1>                 db "Do you want to rename ", 0
48911 00010C04 6E7420746F2072656E- <1>
48912 00010C0D 616D652000          <1>
48913 00010C12 66696C652000        <1> Rename_File:    db "file ", 0
48914 00010C18 6469726563746F7279- <1> Rename_Directory: db "directory ", 0
48915 00010C21 2000                <1>
48916 00010C23 00<rept>            <1> Rename_OldName: times 13 db 0
48917 00010C30 20617320            <1> Msg_File_rename_as: db " as "
```

```
48918 00010C34 00<rept>            <1> Rename_NewName: times 13 db 0
48919                              <1>
48920                              <1> ; 08/03/2016
48921                              <1> ; CMD_INTR.ASM - 01/08/2010 - 23/04/2011
48922                              <1> msg_not_same_drv:
48923 00010C41 4E6F742073616D6520- <1>                 db "Not same drive!"
48924 00010C4A 647269766521        <1>
48925 00010C50 0D0A00              <1>                 db 0Dh, 0Ah, 0
48926                              <1>
48927                              <1> Msg_DoYouWantMoveFile:
48928 00010C53 446F20796F75207761- <1>                 db "Do you want to move file", 0
48929 00010C5C 6E7420746F206D6F76- <1>
48930 00010C65 652066696C6500      <1>
48931                              <1>
48932                              <1> msg_insufficient_disk_space:
48933 00010C6C 496E73756666696369- <1>                 db "Insufficient disk space!"
48934 00010C75 656E74206469736B20- <1>
48935 00010C7E 737061636521        <1>
48936 00010C84 0D0A00              <1>                 db 0Dh, 0Ah, 0
48937                              <1>
48938                              <1> ; 01/08/2010
48939                              <1> msg_source_file:
48940 00010C87 0D0A536F7572636520- <1>              db 0Dh, 0Ah, "Source file name      :   "
48941 00010C90 66696C65206E616D65- <1>
48942 00010C99 2020202020203A2020- <1>
48943 00010CA2 20                  <1>
48944                              <1> msg_source_file_drv:
48945 00010CA3 203A00              <1>              db " :", 0
48946                              <1> msg_destination_file:
48947 00010CA6 0D0A44657374696E61- <1>              db 0Dh, 0Ah, "Destination file name :   "
48948 00010CAF 74696F6E2066696C65- <1>
48949 00010CB8 206E616D65203A2020- <1>
48950 00010CC1 20                  <1>
48951                              <1> msg_destination_file_drv:
48952 00010CC2 203A00              <1>              db " :", 0
48953                              <1> msg_copy_nextline:
48954 00010CC5 0D0A00              <1>              db 0Dh, 0Ah, 0
48955                              <1>
48956                              <1> ; 15/03/2016
48957                              <1> ; CMD_INTR.ASM
48958                              <1>
48959                              <1> Msg_DoYouWantOverWriteFile:
48960 00010CC8 446F20796F75207761- <1>                  db "Do you want to overwrite file ",0
48961 00010CD1 6E7420746F206F7665- <1>
48962 00010CDA 727772697465206669- <1>
48963 00010CE3 6C652000            <1>
48964                              <1>
48965                              <1> Msg_DoYouWantCopyFile:
48966 00010CE7 446F20796F75207761- <1>                  db "Do you want to copy file",0
48967 00010CF0 6E7420746F20636F70- <1>
48968 00010CF9 792066696C6500      <1>
48969                              <1>
48970                              <1> Msg_read_file_error_before_EOF:
48971 00010D00 46696C652072656164- <1>              db "File reading error! (before EOF)"
48972 00010D09 696E67206572726F72- <1>
48973 00010D12 2120286265666F7265- <1>
48974 00010D1B 20454F4629          <1>
48975 00010D20 0A0A00              <1>              db 0Ah, 0Ah, 0
48976                              <1>
48977                              <1> ; 18/03/2016
48978                              <1> ; TRDOS 386 (v2.0) mainprog copy procedure
48979                              <1> msg_reading:
48980 00010D23 52656164696E672E2E- <1>              db "Reading... ", 0
48981 00010D2C 2E2000              <1>
48982                              <1> msg_writing:
48983 00010D2F 57726974696E672E2E- <1>              db "Writing... ", 0
48984 00010D38 2E2000              <1>
48985                              <1> percentagestr:
48986 00010D3B 2020202500          <1>              db "   %", 0  ; "  0%" .. "100%"
48987                              <1> ; 11/04/2016
48988                              <1> Msg_No_Set_Space:
48989 00010D40 496E73756666696369- <1>                  db "Insufficient environment space!"
48990 00010D49 656E7420656E766972- <1>
48991 00010D52 6F6E6D656E74207370- <1>
48992 00010D5B 61636521            <1>
48993 00010D5F 0D0A00              <1>                  db 0Dh, 0Ah, 0
48994                              <1> ; 18/04/2016
48995                              <1> isc_msg:
48996 00010D62 0D0A                <1>              db 0Dh, 0Ah
48997 00010D64 494E56414C49442053- <1>              db "INVALID SYSTEM CALL", 0
48998 00010D6D 595354454D2043414C- <1>
48999 00010D76 4C00                <1>
49000                              <1> usi_msg:
49001 00010D78 0D0A                <1>              db 0Dh, 0Ah
49002 00010D7A 554E444546494E4544- <1>              db "UNDEFINED SOFTWARE INTERRUPT", 0
49003 00010D83 20534F465457415245- <1>
49004 00010D8C 20494E544552525550- <1>
49005 00010D95 5400                <1>
49006                              <1> ifc_msg:
49007 00010D97 0D0A                <1>              db 0Dh, 0Ah
49008 00010D99 494E56414C49442046- <1>              db "INVALID FUNCTION CALL"
49009 00010DA2 554E4354494F4E2043- <1>
49010 00010DAB 414C4C              <1>
49011                              <1> inv_msg_for_trdos_v2:
49012 00010DAE 20                  <1>              db 20h
49013 00010DAF 666F72205452444F53- <1>              db "for TRDOS v2!"
49014 00010DB8 20763221            <1>
49015 00010DBC 07                  <1>              db 07h
49016 00010DBD 0D0A                <1>              db 0Dh, 0Ah
49017 00010DBF 0D0A                <1>              db 0Dh, 0Ah
49018 00010DC1 494E5420            <1>              db "INT "
49019 00010DC5 303068              <1> int_num_str: db "00h"
49020 00010DC8 0D0A                <1>              db 0Dh, 0Ah
```

```
49021 00010DCA 454158203A20        <1>             db "EAX : "
49022 00010DD0 303030303030303068- <1> eax_str:    db "00000000h", 0Dh, 0Ah
49023 00010DD9 0D0A                <1>
49024 00010DDB 454950203A20        <1>             db "EIP : "
49025 00010DE1 303030303030303068- <1> eip_str:    db "00000000h", 0Dh, 0Ah, 0
49026 00010DEA 0D0A00              <1>
49027                              <1>
49028                              <1> ; 07/10/2016
49029                              <1> ; Device names & parameters (for kernel devices)
49030                              <1>
49031 00010DED 90                  <1> align 2
49032                              <1> KDEV_NAME:
49033 00010DEE 5454590000000000    <1>             db 'TTY',0,0,0,0,0 ; 1
49034 00010DF6 4D454D0000000000    <1>             db 'MEM',0,0,0,0,0 ; 2
49035 00010DFE 4644300000000000    <1>             db 'FD0',0,0,0,0,0 ; 3
49036 00010E06 4644310000000000    <1>             db 'FD1',0,0,0,0,0 ; 4
49037 00010E0E 4844300000000000    <1>             db 'HD0',0,0,0,0,0 ; 5
49038 00010E16 4844310000000000    <1>             db 'HD1',0,0,0,0,0 ; 6
49039 00010E1E 4844320000000000    <1>             db 'HD2',0,0,0,0,0 ; 7
49040 00010E26 4844330000000000    <1>             db 'HD3',0,0,0,0,0 ; 8
49041 00010E2E 4C50540000000000    <1>             db 'LPT',0,0,0,0,0 ; 9
49042 00010E36 5454593000000000    <1>             db 'TTY0',0,0,0,0 ; 10
49043 00010E3E 5454593100000000    <1>             db 'TTY1',0,0,0,0 ; 11
49044 00010E46 5454593200000000    <1>             db 'TTY2',0,0,0,0 ; 12
49045 00010E4E 5454593300000000    <1>             db 'TTY3',0,0,0,0 ; 13
49046 00010E56 5454593400000000    <1>             db 'TTY4',0,0,0,0 ; 14
49047 00010E5E 5454593500000000    <1>             db 'TTY5',0,0,0,0 ; 15
49048 00010E66 5454593600000000    <1>             db 'TTY6',0,0,0,0 ; 16
49049 00010E6E 5454593700000000    <1>             db 'TTY7',0,0,0,0 ; 17
49050 00010E76 5454593800000000    <1>             db 'TTY8',0,0,0,0 ; 18
49051 00010E7E 5454593900000000    <1>             db 'TTY9',0,0,0,0 ; 19
49052 00010E86 434F4D3100000000    <1>             db 'COM1',0,0,0,0 ; 18
49053 00010E8E 434F4D3200000000    <1>             db 'COM2',0,0,0,0 ; 19
49054                              <1>             ;db 'CONSOLE',0      ; 1
49055                              <1>             ;db 'PRINTER',0   ; 9
49056                              <1>             ;db 'CDROM'    ; 20
49057                              <1>             ;db 'CDROM0'   ; 20
49058                              <1>             ;db 'CDROM1'   ; 21
49059                              <1>             ;db 'DVD'      ; 22
49060                              <1>             ;db 'DVD0'     ; 22
49061                              <1>             ;db 'DVD1'     ; 23
49062                              <1>             ;db 'USB'      ; 24
49063                              <1>             ;db 'USB0'     ; 24
49064                              <1>             ;db 'USB1'     ; 25
49065                              <1>             ;db 'USB2'     ; 26
49066                              <1>             ;db 'USB3'        ; 27
49067                              <1>             ;db 'KEYBOARD'       ; 1
49068                              <1>             ;db 'MOUSE'    ; 28
49069                              <1>             ;db 'SOUND'    ; 29
49070                              <1>             ;db 'VGA',0,0,0,0 ; 30
49071                              <1>             ;db 'CGA',0,0,0,0 ; 31
49072                              <1>             ;db 'AUDIO',0,0,0 ; 29
49073                              <1>             ;db 'VIDEO',0,0,0 ; 32
49074                              <1>             ;db 'MUSIC',0,0,0 ; 33
49075                              <1>             ;db 'ETHERNET'       ; 34
49076                              <1>             ;db 'SD0',0,0,0,0,0 ; 35
49077                              <1>             ;db 'SD1',0,0,0,0,0 ; 36
49078                              <1>             ;db 'SD2',0,0,0,0,0 ; 37
49079                              <1>             ;db 'SD3',0,0,0,0,0 ; 38
49080                              <1>             ;db 'SATA0'    ; 35
49081                              <1>             ;db 'SATA1'    ; 36
49082                              <1>             ;db 'SATA2'       ; 37
49083                              <1>             ;db 'SATA3'       ; 38
49084                              <1>             ;db 'PATA0',0,0,0  ; 5
49085                              <1>             ;db 'PATA1',0,0,0  ; 6
49086                              <1>             ;db 'PATA2',0,0,0  ; 7
49087                              <1>             ;db 'PATA3',0,0,0  ; 8
49088                              <1>             ;db 'WIRELESS'       ; 39
49089                              <1>             ;db 'HDMI',0,0,0,0 ; 40
49090 00010E96 4E554C4C00000000    <1>             db 'NULL',0,0,0,0 ; 0
49091                              <1>
49092                              <1> NumOfKernelDevNames equ ($-KDEV_NAME) / 8 ; 20 (07/10/2016)
49093                              <1>
49094                              <1> KDEV_NUMBER:
49095 00010E9E 0102030405060708 09 <1>             db 1,2,3,4,5,6,7,8,9
49096 00010EA7 0A0B0C0D0E0F101112- <1>             db 10,11,12,13,14,15,16,17,18,19
49097 00010EB0 13                  <1>
49098 00010EB1 121300              <1>             db 18,19,0
49099                              <1>
49100                              <1> NumOfKernelDevices equ $ - KDEV_NUMBER
49101                              <1>
49102                              <1> KDEV_OADDR:
49103 00010EB4 [D1060100]          <1>             dd otty ;tty  ; 1
49104 00010EB8 [D1060100]          <1>             dd sret ;mem  ; 2
49105 00010EBC [D1060100]          <1>             dd sret ;fd0  ; 3
49106 00010EC0 [D1060100]          <1>             dd sret ;fd1  ; 4
49107 00010EC4 [D1060100]          <1>             dd sret ;hd0  ; 5
49108 00010EC8 [D1060100]          <1>             dd sret ;hd1  ; 6
49109 00010ECC [D1060100]          <1>             dd sret ;hd2  ; 7
49110 00010ED0 [D1060100]          <1>             dd sret ;hd3  ; 8
49111 00010ED4 [D1060100]          <1>             dd sret ;lpt  ; 9
49112 00010ED8 [D1060100]          <1>             dd ocvt ;tty0 ; 10
49113 00010EDC [D1060100]          <1>             dd ocvt ;tty1 ; 11
49114 00010EE0 [D1060100]          <1>             dd ocvt ;tty2 ; 12
49115 00010EE4 [D1060100]          <1>             dd ocvt ;tty3 ; 13
49116 00010EE8 [D1060100]          <1>             dd ocvt ;tty4 ; 14
49117 00010EEC [D1060100]          <1>             dd ocvt ;tty5 ; 15
49118 00010EF0 [D1060100]          <1>             dd ocvt ;tty6 ; 16
49119 00010EF4 [D1060100]          <1>             dd ocvt ;tty7 ; 17
49120 00010EF8 [D1060100]          <1>             dd ocvt ;tty8 ; 18
49121 00010EFC [D1060100]          <1>             dd ocvt ;tty9 ; 19
49122                              <1>             ;dd ocvt ;com1 ; 18
49123                              <1>             ;dd ocvt ;com2 ; 19
```

```
49124 00010F00 [D1060100]          <1>              dd sret ;null ; 20
49125                              <1> KDEV_CADDR:
49126 00010F04 [D1060100]          <1>              dd ctty ;tty  ; 1
49127 00010F08 [D1060100]          <1>              dd cret ;mem  ; 2
49128 00010F0C [D1060100]          <1>              dd cret ;fd0  ; 3
49129 00010F10 [D1060100]          <1>              dd cret ;fd1  ; 4
49130 00010F14 [D1060100]          <1>              dd cret ;hd0  ; 5
49131 00010F18 [D1060100]          <1>              dd cret ;hd1  ; 6
49132 00010F1C [D1060100]          <1>              dd cret ;hd2  ; 7
49133 00010F20 [D1060100]          <1>              dd cret ;hd3  ; 8
49134 00010F24 [D1060100]          <1>              dd cret ;lpt  ; 9
49135 00010F28 [D1060100]          <1>              dd ocvt ;tty0 ; 10
49136 00010F2C [D1060100]          <1>              dd ccvt ;tty1 ; 11
49137 00010F30 [D1060100]          <1>              dd ccvt ;tty2 ; 12
49138 00010F34 [D1060100]          <1>              dd ccvt ;tty3 ; 13
49139 00010F38 [D1060100]          <1>              dd ccvt ;tty4 ; 14
49140 00010F3C [D1060100]          <1>              dd ccvt ;tty5 ; 15
49141 00010F40 [D1060100]          <1>              dd ccvt ;tty6 ; 16
49142 00010F44 [D1060100]          <1>              dd ccvt ;tty7 ; 17
49143 00010F48 [D1060100]          <1>              dd ccvt ;tty8 ; 18
49144 00010F4C [D1060100]          <1>              dd ccvt ;tty9 ; 19
49145                              <1>              ;dd ccvt ;com1 ; 18
49146                              <1>              ;dd ccvt ;com2 ; 19
49147 00010F50 [D1060100]          <1>              dd cret ;null ; 20
49148                              <1>
49149                              <1> KDEV_RADDR:
49150 00010F54 [D1060100]          <1>              dd rtty ;tty  ; 1
49151 00010F58 [D1060100]          <1>              dd rmem ;mem  ; 2
49152 00010F5C [D1060100]          <1>              dd rfd  ;fd0  ; 3
49153 00010F60 [D1060100]          <1>              dd rfd  ;fd1  ; 4
49154 00010F64 [D1060100]          <1>              dd rhd  ;hd0  ; 5
49155 00010F68 [D1060100]          <1>              dd rhd  ;hd1  ; 6
49156 00010F6C [D1060100]          <1>              dd rhd  ;hd2  ; 7
49157 00010F70 [D1060100]          <1>              dd rhd  ;hd3  ; 8
49158 00010F74 [D1060100]          <1>              dd rlpt ;lpt  ; 9
49159 00010F78 [D1060100]          <1>              dd rcvt ;tty0 ; 10
49160 00010F7C [D1060100]          <1>              dd rcvt ;tty1 ; 11
49161 00010F80 [D1060100]          <1>              dd rcvt ;tty2 ; 12
49162 00010F84 [D1060100]          <1>              dd rcvt ;tty3 ; 13
49163 00010F88 [D1060100]          <1>              dd rcvt ;tty4 ; 14
49164 00010F8C [D1060100]          <1>              dd rcvt ;tty5 ; 15
49165 00010F90 [D1060100]          <1>              dd rcvt ;tty6 ; 16
49166 00010F94 [D1060100]          <1>              dd rcvt ;tty7 ; 17
49167 00010F98 [D1060100]          <1>              dd rcvt ;tty8 ; 18
49168 00010F9C [D1060100]          <1>              dd rcvt ;tty9 ; 19
49169                              <1>              ;dd rcvt ;com1 ; 18
49170                              <1>              ;dd rcvt ;com2 ; 19
49171 00010FA0 [C2FA0000]          <1>              dd rnull ;null ; 20
49172                              <1> KDEV_WADDR:
49173 00010FA4 [D1060100]          <1>              dd wtty ;tty  ; 1
49174 00010FA8 [D1060100]          <1>              dd wmem ;mem  ; 2
49175 00010FAC [D1060100]          <1>              dd wfd  ;fd0  ; 3
49176 00010FB0 [D1060100]          <1>              dd wfd  ;fd1  ; 4
49177 00010FB4 [D1060100]          <1>              dd whd  ;hd0  ; 5
49178 00010FB8 [D1060100]          <1>              dd whd  ;hd1  ; 6
49179 00010FBC [D1060100]          <1>              dd whd  ;hd2  ; 7
49180 00010FC0 [D1060100]          <1>              dd whd  ;hd3  ; 8
49181 00010FC4 [D1060100]          <1>              dd wlpt ;lpt  ; 9
49182 00010FC8 [D1060100]          <1>              dd xmtt ;tty0 ; 10
49183 00010FCC [D1060100]          <1>              dd xmtt ;tty1 ; 11
49184 00010FD0 [D1060100]          <1>              dd xmtt ;tty2 ; 12
49185 00010FD4 [D1060100]          <1>              dd xmtt ;tty3 ; 13
49186 00010FD8 [D1060100]          <1>              dd xmtt ;tty4 ; 14
49187 00010FDC [D1060100]          <1>              dd xmtt ;tty5 ; 15
49188 00010FE0 [D1060100]          <1>              dd xmtt ;tty6 ; 16
49189 00010FE4 [D1060100]          <1>              dd xmtt ;tty7 ; 17
49190 00010FE8 [D1060100]          <1>              dd xmtt ;tty8 ; 18
49191 00010FEC [D1060100]          <1>              dd xmtt ;tty9 ; 19
49192                              <1>              ;dd xmtt ;com1 ; 18
49193                              <1>              ;dd xmtt ;com2 ; 19
49194 00010FF0 [C3FA0000]          <1>              dd wnull ;null ; 20
49195                              <1>
49196                              <1> ; DEV_ACCESS bits:
49197                              <1>       ; bit 0 = accessable by normal users
49198                              <1>       ; bit 1 = read access permission
49199                              <1>       ; bit 2 = write access permission
49200                              <1>       ; bit 3 = IOCTL permission to users
49201                              <1>       ; bit 4 = block device if it is set
49202                              <1>       ; bit 5 = 16 bit or 1024 byte data
49203                              <1>       ; bit 6 = 32 bit or 2048 byte data
49204                              <1>       ; bit 7 = installable device driver
49205                              <1>
49206                              <1> KDEV_ACCESS: ; 08/10/2016
49207 00010FF4 07                  <1>              db  00000111b; tty, 1
49208 00010FF5 07                  <1>              db  00000111b; mem, 2
49209 00010FF6 8F                  <1>              db  10001111b; fd0, 3
49210 00010FF7 8F                  <1>              db  10001111b; fd1, 4
49211 00010FF8 8F                  <1>              db  10001111b; hd0, 5
49212 00010FF9 8F                  <1>              db  10001111b; hd1, 6
49213 00010FFA 8F                  <1>              db  10001111b; hd2, 7
49214 00010FFB 8F                  <1>              db  10001111b; hd3, 8
49215 00010FFC 07                  <1>              db  00000111b   ; lpt, 9
49216 00010FFD 07                  <1>              db  00000111b; tty0, 10
49217 00010FFE 07                  <1>              db  00000111b; tty1, 11
49218 00010FFF 07                  <1>              db  00000111b; tty2, 12
49219 00011000 07                  <1>              db  00000111b; tty3, 13
49220 00011001 07                  <1>              db  00000111b; tty4, 14
49221 00011002 07                  <1>              db  00000111b; tty5, 15
49222 00011003 07                  <1>              db  00000111b; tty6, 16
49223 00011004 07                  <1>              db  00000111b; tty7, 17
49224 00011005 07                  <1>              db  00000111b; tty8, 18
49225 00011006 07                  <1>              db  00000111b; tty9, 19
49226                              <1>              ;db 00000111b; com1, 18
```

```
49227                                  <1>             ;db 00000111b; com2, 19
49228 00011007 00                      <1>             db  00000000b   ; null, 0
49229                                  <1>
49230                                  <1> ; 07/10/2016
49231                                  <1> NumOfInstallableDevices equ 8
49232                                  <1> NUMIDEV            equ NumOfInstallableDevices ; 8
49233                                  <1> NUMOFDEVICES equ NumOfKernelDevices + NumOfInstallableDevices
49234                                  <1>
49235                                  <1> ; 26/02/2017
49236                                  <1> ; IRQ Callback (& Signal Response Byte) service availibity
49237                                  <1> ; 'syscalbac'
49238                                  <1> ; ************************************************
49239                                  <1> ; IRQ 00 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15
49240                                  <1> ; --- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
49241                                  <1> ; --- 00 00 00 01 02 03 00 04 00 05 06 07 08 09 00 00
49242                                  <1> ; ************************************************
49243                                  <1> IRQenum:
49244 00011008 000000010203000400-     <1>       db  0,0,0,1,2,3,0,4,0,5,6,7,8,9,0,0
49245 00011011 05060708090000          <1>
49246                                  <1>
49247                                  <1> ; 28/08/2017
49248                                  <1> ; 20/08/2017
49249                                  <1> ; DMA Registers (for 'sysdma')
49250                                  <1> ; 02/07/2017 (sb16mod.s)
49251 00011018 00020406C0C4C8CC        <1> dma_adr:    db 0,2,4,6,0C0h,0C4h,0C8h,0CCh
49252 00011020 01030507C2C6CACE        <1> dma_cnt:    db 1,3,5,7,0C2h,0C6h,0CAh,0CEh
49253 00011028 878381828F8B898A        <1> dma_page:   db 87h,83h,81h,82h,8Fh,8Bh,89h,8Ah ; 03/08/2017
49254 00011030 0A0A0A0AD4D4D4D4        <1> dma_mask:   db 0Ah,0Ah,0Ah,0Ah,0D4h,0D4h,0D4h,0D4h
49255 00011038 0B0B0B0BD6D6D6D6        <1> dma_mod:    db 0Bh,0Bh,0Bh,0Bh,0D6h,0D6h,0D6h,0D6h
49256 00011040 0C0C0C0CD8D8D8D8        <1> dma_flip:   db 0Ch,0Ch,0Ch,0Ch,0D8h,0D8h,0D8h,0D8h
49257
49258                                      ; 27/08/2014
49259                                  scr_row:
49260 00011048 E0810B00                      dd 0B8000h + 0A0h + 0A0h + 0A0h ; Row 3
49261                                  scr_col:
49262 0001104C 00000000                      dd 0
49263
49264                                  Align 4
49265                                        ; 15/04/2016
49266                                        ; TRDOS 386 (TRDOS v2.0)
49267
49268                                        ; 21/08/2014
49269                                  ilist:
49270                                        ;times       32 dd cpu_except ; INT 0 to INT 1Fh
49271                                        ;
49272                                        ; Exception list
49273                                        ; 25/08/2014
49274 00011050 [17090000]                    dd     exc0   ; 0h,  Divide-by-zero Error
49275 00011054 [1E090000]                    dd     exc1
49276 00011058 [25090000]                    dd     exc2
49277 0001105C [2C090000]                    dd     exc3
49278 00011060 [30090000]                    dd     exc4
49279 00011064 [34090000]                    dd     exc5
49280 00011068 [38090000]                    dd     exc6   ; 06h,  Invalid Opcode
49281 0001106C [3C090000]                    dd     exc7
49282 00011070 [40090000]                    dd     exc8
49283 00011074 [44090000]                    dd     exc9
49284 00011078 [48090000]                    dd     exc10
49285 0001107C [4C090000]                    dd     exc11
49286 00011080 [50090000]                    dd     exc12
49287 00011084 [54090000]                    dd     exc13  ; 0Dh, General Protection Fault
49288 00011088 [58090000]                    dd     exc14  ; 0Eh, Page Fault
49289 0001108C [5C090000]                    dd     exc15
49290 00011090 [60090000]                    dd     exc16
49291 00011094 [64090000]                    dd     exc17
49292 00011098 [68090000]                    dd     exc18
49293 0001109C [6C090000]                    dd     exc19
49294 000110A0 [70090000]                    dd     exc20
49295 000110A4 [74090000]                    dd     exc21
49296 000110A8 [78090000]                    dd     exc22
49297 000110AC [7C090000]                    dd     exc23
49298 000110B0 [80090000]                    dd     exc24
49299 000110B4 [84090000]                    dd     exc25
49300 000110B8 [88090000]                    dd     exc26
49301 000110BC [8C090000]                    dd     exc27
49302 000110C0 [90090000]                    dd     exc28
49303 000110C4 [94090000]                    dd     exc29
49304 000110C8 [98090000]                    dd     exc30
49305 000110CC [9C090000]                    dd     exc31
49306                                  IRQ_list: ; 28/02/2017 ('syscalbac')
49307                                        ; Interrupt list
49308 000110D0 [8B060000]                    dd     timer_int    ; INT 20h
49309                                               ;dd    irq0
49310 000110D4 [FF0D0000]                    dd     kb_int       ; 24/01/2016
49311                                               ;dd    irq1
49312 000110D8 [6D080000]                    dd     irq2
49313                                               ; COM2 int
49314 000110DC [71080000]                    dd     irq3
49315                                               ; COM1 int
49316 000110E0 [7C080000]                    dd     irq4
49317 000110E4 [87080000]                    dd     irq5
49318                                  ;DISKETTE_INT: ;06/02/2015
49319 000110E8 [B0410000]                    dd     fdc_int             ; 16/02/2015, IRQ 6 handler
49320                                               ;dd    irq6
49321                                  ; Default IRQ 7 handler against spurious IRQs (from master PIC)
49322                                  ; 25/02/2015 (source: http://wiki.osdev.org/8259_PIC)
49323 000110EC [F60B0000]                    dd     default_irq7 ; 25/02/2015
49324                                               ;dd    irq7
49325                                  ; Real Time Clock Interrupt
49326 000110F0 [F6070000]                    dd     rtc_int             ; 23/02/2015, IRQ 8 handler
49327                                               ;dd    irq8   ; INT 28h
49328 000110F4 [97080000]                    dd     irq9
49329 000110F8 [9B080000]                    dd     irq10
```

```
49330 000110FC [9F080000]                 dd    irq11
49331 00011100 [A3080000]                 dd    irq12
49332 00011104 [A7080000]                 dd    irq13
49333                                ;HDISK_INT1:  ;06/02/2015
49334 00011108 [2D4B0000]                 dd    hdc1_int     ; 21/02/2015, IRQ 14 handler
49335                                           ;dd   irq14
49336                                ;HDISK_INT2:  ;06/02/2015
49337 0001110C [544B0000]                 dd    hdc2_int     ; 21/02/2015, IRQ 15 handler
49338                                           ;dd   irq15  ; INT 2Fh
49339                                           ; 14/08/2015
49340                                     ;dd   sysent       ; INT 30h (system calls)
49341
49342                                     ; 15/04/2016
49343                                     ; TRDOS 386(TRDOS v2.0) Software Interrupts
49344
49345 00011110 [6D110100]                 dd    int30h       ; Reserved for
49346                                                        ; !!! Retro UNIX (RUNIX) !!!
49347                                                        ; !!! SINGLIX !!! System Calls
49348 00011114 [F2140000]                 dd    int31h       ; Video BIOS (IBM PC/AT, Int 10h)
49349 00011118 [1E0C0000]                 dd    int32h       ; Keyboard Functions (IBM PC/AT, Int 16h)
49350 0001111C [66420000]                 dd    int33h       ; DISK I/O (IBM PC/AT, Int 13h)
49351 00011120 [73F30000]                 dd    int34h       ; #IOCTL# (I/O port access support for ring 3)
49352 00011124 [82590000]                 dd    int35h       ; Time/Date Functions (IBM PC/AT, Int 1Ah)
49353 00011128 [AA0A0000]                 dd    ignore_int   ; INT 36h : Timer Functions
49354 0001112C [AA0A0000]                 dd    ignore_int   ; INT 37h
49355 00011130 [AA0A0000]                 dd    ignore_int   ; INT 38h
49356 00011134 [AA0A0000]                 dd    ignore_int   ; INT 39h
49357 00011138 [AA0A0000]                 dd    ignore_int   ; INT 3Ah
49358 0001113C [AA0A0000]                 dd    ignore_int   ; INT 3Bh
49359 00011140 [AA0A0000]                 dd    ignore_int   ; INT 3Ch
49360 00011144 [AA0A0000]                 dd    ignore_int   ; INT 3Dh
49361 00011148 [AA0A0000]                 dd    ignore_int   ; INT 3Eh
49362 0001114C [AA0A0000]                 dd    ignore_int   ; INT 3Fh
49363 00011150 [8BC30000]                 dd    sysent       ; INT 40h : !!! TRDOS 386 System Calls !!!
49364                                     ;dd   ignore_int
49365 00011154 00000000                   dd    0
49366
49367                                ; 20/08/2014
49368                                  ; /* This is the default interrupt "handler" :-) */
49369                                  ; Linux v0.12 (head.s)
49370                                int_msg:
49371 00011158 556E6B6E6F776E2069-          db "Unknown interrupt ! ", 0
49372 00011161 6E7465727275707420-
49373 0001116A 212000
49374
49375                                ; 15/04/2016
49376                                ; TRDOS 386 (TRDOS v2.0)
49377
49378                                ; 29/04/2016
49379                                int30h:
49380                                trdos_isc_routine:
49381                                     ; 02/05/2016
49382                                     ; 01/05/2016
49383                                     ; 29/04/2016
49384                                     ; 18/04/2016
49385                                     ; 15/04/2016 (TRDOS 386 = TRDOS v2.0)
49386                                     ; 17/04/2011 (TRDOS v1.0, 'IFC.ASM')
49387                                     ; 03/02/2011 ('trdos_ifc_routine')
49388                                     ;
49389 0001116D 8B1C24                    mov   ebx, [esp] ; EIP (next)
49390 00011170 83EB02                    sub   ebx, 2 ; EIP (CD ##h)
49391
49392 00011173 89C1                       mov   ecx, eax
49393 00011175 8A4301                     mov   al, [ebx+1] ; CDh ##h
49394
49395 00011178 66BA1000                   mov   dx, KDATA
49396 0001117C 8EDA                       mov   ds, dx
49397 0001117E 8EC2                       mov   es, dx
49398
49399 00011180 FC                         cld
49400 00011181 8B15[20520100]             mov   edx, [k_page_dir]
49401 00011187 0F22DA                     mov   cr3, edx
49402
49403 0001118A E83A21FFFF                 call  bytetohex
49404 0001118F 66A3[C50D0100]             mov   [int_num_str], ax
49405
49406 00011195 89D8                       mov   eax, ebx ; EIP
49407 00011197 E86D21FFFF                 call  dwordtohex
49408 0001119C 8915[E10D0100]             mov   [eip_str], edx
49409 000111A2 A3[E50D0100]               mov   [eip_str+4], eax
49410
49411 000111A7 89C8                       mov   eax, ecx
49412 000111A9 E85B21FFFF                 call  dwordtohex
49413 000111AE 8915[D00D0100]             mov   [eax_str], edx
49414 000111B4 A3[D40D0100]               mov   [eax_str+4], eax
49415
49416 000111B9 43                         inc   ebx
49417 000111BA 8A03                       mov   al, [ebx] ; Interrupt number
49418
49419                                trdos_isc_handler:
49420 000111BC 80FE30                     cmp   dh, 30h ; Retro UNIX, SINGLIX System calls
49421 000111BF 7507                       jne   short trdos_usi_handler
49422 000111C1 BE[620D0100]               mov   esi, isc_msg
49423 000111C6 EB05                       jmp   short loc_write_inv_system_call_msg
49424
49425                                trdos_usi_handler:
49426 000111C8 BE[780D0100]               mov   esi, usi_msg
49427
49428                                loc_write_inv_system_call_msg:
49429 000111CD E88B51FFFF                 call  print_msg
49430                                     ; 29/04/2016
49431 000111D2 BE[AE0D0100]               mov   esi, inv_msg_for_trdos_v2
49432 000111D7 E88151FFFF                 call  print_msg
```

```
                                    loc_ifc_terminate_process:
                                          ; u.uno = process number
                                          ; 29/04/2016
                                          ; 02/05/2016
000111DC FE05[5B030300]                   inc   byte [sysflg] ; 0FFh -> 0
000111E2 B801000000                       mov   eax, 1
000111E7 E978B4FFFF                       jmp   sysexit
                                    ; 07/03/2015
                                    ; Temporary Code
                                    display_disks:
000111EC 803D[F65C0000]00                 cmp   byte [fd0_type], 0
000111F3 7605                             jna   short ddsks1
000111F5 E87D000000                       call  pdskm
                                    ddsks1:
000111FA 803D[F75C0000]00                 cmp   byte [fd1_type], 0
00011201 760C                             jna   short ddsks2
00011203 C605[47130100]31                 mov   byte [dskx], '1'
0001120A E868000000                       call  pdskm
                                    ddsks2:
0001120F 803D[F85C0000]00                 cmp   byte [hd0_type], 0
00011216 7654                             jna   short ddsk6
00011218 66C705[45130100]68-              mov   word [dsktype], 'hd'
00011220 64
00011221 C605[47130100]30                 mov   byte [dskx], '0'
00011228 E84A000000                       call  pdskm
                                    ddsks3:
0001122D 803D[F95C0000]00                 cmp   byte [hd1_type], 0
00011234 7636                             jna   short ddsk6
00011236 C605[47130100]31                 mov   byte [dskx], '1'
0001123D E835000000                       call  pdskm
                                    ddsks4:
00011242 803D[FA5C0000]00                 cmp   byte [hd2_type], 0
00011249 7621                             jna   short ddsk6
0001124B C605[47130100]32                 mov   byte [dskx], '2'
00011252 E820000000                       call  pdskm
                                    ddsks5:
00011257 803D[FB5C0000]00                 cmp   byte [hd3_type], 0
0001125E 760C                             jna   short ddsk6
00011260 C605[47130100]33                 mov   byte [dskx], '3'
00011267 E80B000000                       call  pdskm
                                    ddsk6:
0001126C BE[6F130100]                     mov   esi, nextline
00011271 E806000000                       call  pdskml
                                    pdskm_ok:
00011276 C3                               retn
                                    pdskm:
00011277 BE[43130100]                     mov   esi, dsk_ready_msg
                                    pdskml:
0001127C AC                               lodsb
0001127D 08C0                             or    al, al
0001127F 74F5                             jz    short pdskm_ok
00011281 56                               push  esi
                                          ; 13/05/2016
00011282 BB07000000                         mov     ebx, 7  ; Black background,
                                                    ; light gray forecolor
                                                    ; Video page 0 (bh=0)
00011287 E8260AFFFF                       call  _write_tty
0001128C 5E                               pop   esi
0001128D EBED                             jmp   short pdskml

0001128F 90                         Align 2
                                          ; 21/08/2014
                                    exc_msg:
00011290 435055206578636570-              db "CPU exception ! "
00011299 74696F6E202120
                                    excnstr:          ; 25/08/2014
000112A0 3F3F68202045495020-              db "??h", "  EIP : "
000112A9 3A20
                                    EIPstr: ; 29/08/2014
000112AB 00<rept>                         times 12 db 0

                                          ; 23/02/2015
                                          ; 25/08/2014
                                    ;scounter:
                                    ;     db 5
                                    ;     db 19

                                    ; 06/11/2014
                                    ; Memory Information message
                                    ; 14/08/2015
                                    msg_memory_info:
000112B7 07                               db    07h
000112B8 0D0A                             db    0Dh, 0Ah
                                          ;db   "MEMORY ALLOCATION INFO", 0Dh, 0Ah, 0Dh, 0Ah
000112BA 546F74616C206D656D-              db    "Total memory : "
000112C3 6F7279203A20
                                    mem_total_b_str: ; 10 digits
000112C9 303030303030303030-              db    "0000000000 bytes", 0Dh, 0Ah
000112D2 302062797465730D0A
000112DB 2020202020202020-                db    "              ", 20h, 20h, 20h
000112E4 2020202020202020
                                    mem_total_p_str: ; 7 digits
000112ED 303030303030302070-              db    "0000000 pages", 0Dh, 0Ah
000112F6 616765730D0A
000112FC 0D0A                             db    0Dh, 0Ah
000112FE 46726565206D656D6F-              db    "Free memory  : "
00011307 727920203A20
                                    free_mem_b_str:  ; 10 digits
0001130D 3F3F3F3F3F3F3F3F3F-              db    "?????????? bytes", 0Dh, 0Ah
```

```
49536 00011316 3F2062797465730D0A
49537 0001131F 202020202020202020-          db    "               ", 20h, 20h, 20h
49538 00011328 202020202020202020
49539                                  free_mem_p_str:  ; 7 digits
49540 00011331 3F3F3F3F3F3F3F2070-          db    "??????? pages", 0Dh, 0Ah
49541 0001133A 616765730D0A
49542 00011340 0D0A00                       db    0Dh, 0Ah, 0
49543
49544                                  dsk_ready_msg:
49545 00011343 0D0A                         db    0Dh, 0Ah
49546                                  dsktype:
49547 00011345 6664                         db    'fd'
49548                                  dskx:
49549 00011347 30                           db    '0'
49550 00011348 20                           db    20h
49551 00011349 697320524541445920-          db    'is READY ...'
49552 00011352 2E2E2E
49553 00011355 00                           db    0
49554
49555                                  setup_error_msg:
49556 00011356 0D0A                         db 0Dh, 0Ah
49557 00011358 4469736B2053657475-          db 'Disk Setup Error !'
49558 00011361 70204572726F722021
49559 0001136A 0D0A00                       db 0Dh, 0Ah,0
49560
49561                                  next2line: ; 08/02/2016
49562 0001136D 0D0A                         db    0Dh, 0Ah
49563                                  nextline:
49564 0001136F 0D0A00                       db    0Dh, 0Ah, 0
49565
49566                                  ; KERNEL - SYSINIT Messages
49567                                  ; 24/08/2015
49568                                  ; 13/04/2015 - (Retro UNIX 386 v1 Beginning)
49569                                  ; 14/07/2013
49570                                  ;kernel_init_err_msg:
49571                                  ;     db 0Dh, 0Ah
49572                                  ;     db 07h
49573                                  ;     db 'Kernel initialization ERROR !'
49574                                  ;     db 0Dh, 0Ah, 0
49575
49576                                  ;welcome_msg:
49577                                  ;     db 0Dh, 0Ah
49578                                  ;     db 07h
49579                                  ;     db 'Welcome to TRDOS 386 Operating System !'
49580                                  ;     db 0Dh, 0Ah
49581                                  ;     db 'by Erdogan Tan - 27/12/2017 (v2.0.0)'
49582                                  ;     db 0Dh, 0Ah, 0
49583
49584                                  panic_msg:
49585 00011372 0D0A07                       db 0Dh, 0Ah, 07h
49586 00011375 4552524F523A204B65-          db 'ERROR: Kernel Panic !'
49587 0001137E 726E656C2050616E69-
49588 00011387 632021
49589 0001138A 0D0A00                       db 0Dh, 0Ah, 0
49590
49591                                  ;msgl_drv_not_ready:
49592                                  ;     db 07h, 0Dh, 0Ah
49593                                  ;       db 'Drive not ready or read error !'
49594                                  ;       db 0Dh, 0Ah, 0
49595
49596                                  starting_msg:
49597 0001138D 5475726B6973682052-          db "Turkish Rational DOS v2.0 [27/12/2017] ...", 0
49598 00011396 6174696F6E616C2044-
49599 0001139F 4F532076322E30205B-
49600 000113A8 32372F31322F323031-
49601 000113B1 375D202E2E2E00
49602                                  NextLine:
49603 000113B8 0D0A00                       db 0Dh, 0Ah, 0
49604
49605                                  %include 'audio.s' ; 03/04/2017
49606                             <1> ; ****************************************************************************
49607                             <1> ; TRDOS386.ASM (TRDOS 386 Kernel) - v2.0.0 - audio.s
49608                             <1> ; ----------------------------------------------------------------------------
49609                             <1> ; Last Update: 22/10/2017
49610                             <1> ; ----------------------------------------------------------------------------
49611                             <1> ; Beginning: 03/04/2017
49612                             <1> ; ----------------------------------------------------------------------------
49613                             <1> ; Assembler: NASM version 2.11 (trdos386.s)
49614                             <1> ; ****************************************************************************
49615                             <1>
49616                             <1> ; AUDIO CONTROLLER & CODEC DEFINITIONS & CODE FOR TRDOS 386
49617                             <1>
49618                             <1> ;=============================================================================
49619                             <1> ;               EQUATES
49620                             <1> ;=============================================================================
49621                             <1>
49622                             <1> ; PCI EQUATES
49623                             <1>
49624                             <1> BIT0  EQU 1
49625                             <1> BIT1  EQU 2
49626                             <1> BIT2  EQU 4
49627                             <1> BIT3  EQU 8
49628                             <1> BIT4  EQU 10h
49629                             <1> BIT5  EQU 20h
49630                             <1> BIT6  EQU 40h
49631                             <1> BIT7  EQU 80h
49632                             <1> BIT8  EQU 100h
49633                             <1> BIT9  EQU 200h
49634                             <1> BIT10 EQU 400h
49635                             <1> BIT11 EQU 800h
49636                             <1> BIT12 EQU 1000h
49637                             <1> BIT13 EQU 2000h
49638                             <1> BIT14 EQU 4000h
```

```
49639                                   <1> BIT15 EQU 8000h
49640                                   <1> BIT16 EQU 10000h
49641                                   <1> BIT17 EQU 20000h
49642                                   <1> BIT18 EQU 40000h
49643                                   <1> BIT19 EQU 80000h
49644                                   <1> BIT20 EQU 100000h
49645                                   <1> BIT21 EQU 200000h
49646                                   <1> BIT22 EQU 400000h
49647                                   <1> BIT23 EQU 800000h
49648                                   <1> BIT24 EQU 1000000h
49649                                   <1> BIT25 EQU 2000000h
49650                                   <1> BIT26 EQU 4000000h
49651                                   <1> BIT27 EQU 8000000h
49652                                   <1> BIT28 EQU 10000000h
49653                                   <1> BIT29 EQU 20000000h
49654                                   <1> BIT30 EQU 40000000h
49655                                   <1> BIT31 EQU 80000000h
49656                                   <1> NOT_BIT31 EQU 7FFFFFFFh
49657                                   <1>
49658                                   <1> ; PCI equates
49659                                   <1> ; PCI function address (PFA)
49660                                   <1> ; bit 31 = 1
49661                                   <1> ; bit 23:16 = bus number     (0-255)
49662                                   <1> ; bit 15:11 = device number  (0-31)
49663                                   <1> ; bit 10:8 = function number (0-7)
49664                                   <1> ; bit 7:0 = register number  (0-255)
49665                                   <1>
49666                                   <1> IO_ADDR_MASK    EQU     0FFFEh  ; mask off bit 0 for reading BARs
49667                                   <1> PCI_INDEX_PORT  EQU     0CF8h
49668                                   <1> PCI_DATA_PORT   EQU     0CFCh
49669                                   <1> PCI32           EQU     BIT31   ; bitflag to signal 32bit access
49670                                   <1> PCI16           EQU     BIT30   ; bitflag for 16bit access
49671                                   <1> NOT_PCI32_PCI16    EQU   03FFFFFFh ; NOT BIT31+BIT30 ; 19/03/2017
49672                                   <1>
49673                                   <1> PCI_FN0         EQU     0 << 8
49674                                   <1> PCI_FN1         EQU     1 << 8
49675                                   <1> PCI_FN2         EQU     2 << 8
49676                                   <1> PCI_FN3         EQU     3 << 8
49677                                   <1> PCI_FN4         EQU     4 << 8
49678                                   <1> PCI_FN5         EQU     5 << 8
49679                                   <1> PCI_FN6         EQU     6 << 8
49680                                   <1> PCI_FN7         EQU     7 << 8
49681                                   <1>
49682                                   <1> PCI_CMD_REG  EQU    04h    ; reg 04, command reg
49683                                   <1> IO_ENA              EQU    BIT0   ; i/o decode enable
49684                                   <1> MEM_ENA             EQU    BIT1   ; memory decode enable
49685                                   <1> BM_ENA              EQU     BIT2 ; bus master enable
49686                                   <1>
49687                                   <1> ; VIA VT8233 EQUATES
49688                                   <1>
49689                                   <1> VIA_VID             equ 1106h    ; VIA's PCI vendor ID
49690                                   <1> VT8233_DID      equ 3059h ; VT8233 (VT8235) device ID
49691                                   <1>
49692                                   <1> PCI_IO_BASE          equ 10h
49693                                   <1> AC97_INT_LINE        equ 3Ch
49694                                   <1> VIA_ACLINK_CTRL      equ 41h
49695                                   <1> VIA_ACLINK_STAT      equ 40h
49696                                   <1> VIA_ACLINK_C00_READY equ 01h ; primary codec ready
49697                                   <1>
49698                                   <1> VIA_REG_AC97     equ 80h ; dword
49699                                   <1>
49700                                   <1> VIA_ACLINK_CTRL_ENABLE    equ   80h ; 0: disable, 1: enable
49701                                   <1> VIA_ACLINK_CTRL_RESET     equ   40h ; 0: assert, 1: de-assert
49702                                   <1> VIA_ACLINK_CTRL_SYNC      equ   20h ; 0: release SYNC, 1: force SYNC hi
49703                                   <1> VIA_ACLINK_CTRL_VRA       equ   08h ; 0: disable VRA, 1: enable VRA
49704                                   <1> VIA_ACLINK_CTRL_PCM       equ   04h ; 0: disable PCM, 1: enable PCM
49705                                   <1> VIA_ACLINK_CTRL_INIT      equ  (VIA_ACLINK_CTRL_ENABLE +
VIA_ACLINK_CTRL_RESET +                                 VIA_ACLINK_CTRL_PCM +                                   VIA_ACLINK_CTRL_VRA)
49706                                   <1>
49707                                   <1> CODEC_AUX_VOL             equ   04h
49708                                   <1> VIA_REG_AC97_BUSY  equ   01000000h ;(1<<24)
49709                                   <1> VIA_REG_AC97_CMD_SHIFT    equ   10h ; 16
49710                                   <1> VIA_REG_AC97_PRIMARY_VALID equ 02000000h ;(1<<25)
49711                                   <1> VIA_REG_AC97_READ  equ   00800000h ;(1<<23)
49712                                   <1> VIA_REG_AC97_CODEC_ID_SHIFT   equ  1Eh ; 30
49713                                   <1> VIA_REG_AC97_CODEC_ID_PRIMARY equ  0
49714                                   <1> VIA_REG_AC97_DATA_SHIFT equ   0
49715                                   <1> VIADEV_PLAYBACK         equ   0
49716                                   <1> VIA_REG_OFFSET_STATUS   equ   0    ;; byte - channel status
49717                                   <1> VIA_REG_OFFSET_CONTROL  equ   01h  ;; byte - channel control
49718                                   <1> VIA_REG_CTRL_START equ   80h  ;; WO
49719                                   <1> VIA_REG_CTRL_TERMINATE  equ   40h  ;; WO
49720                                   <1> VIA_REG_CTRL_PAUSE      equ   08h  ;; RW
49721                                   <1> VIA_REG_CTRL_RESET      equ   01h  ;; RW - probably reset? undocumented
49722                                   <1> VIA_REG_OFFSET_STOP_IDX equ   08h  ;; dword - stop index, channel type, sample rate
49723                                   <1> VIA8233_REG_TYPE_16BIT  equ   200000h ;; RW
49724                                   <1> VIA8233_REG_TYPE_STEREO equ   100000h ;; RW
49725                                   <1> VIA_REG_OFFSET_CURR_INDEX equ 0Fh ;; byte - channel current index (for via8233 only)
49726                                   <1> VIA_REG_OFFSET_TABLE_PTR equ  04h  ;; dword - channel table pointer
49727                                   <1> VIA_REG_OFFSET_CURR_PTR equ   04h  ;; dword - channel current pointer
49728                                   <1> VIA_REG_OFS_PLAYBACK_VOLUME_L equ  02h ;; byte
49729                                   <1> VIA_REG_OFS_PLAYBACK_VOLUME_R equ  03h ;; byte
49730                                   <1> VIA_REG_CTRL_AUTOSTART    equ   20h
49731                                   <1> VIA_REG_CTRL_INT_EOL      equ   02h
49732                                   <1> VIA_REG_CTRL_INT_FLAG     equ   01h
49733                                   <1> VIA_REG_CTRL_INT   equ  (VIA_REG_CTRL_INT_FLAG +                                VIA_REG_CTRL_INT_EOL
+                                VIA_REG_CTRL_AUTOSTART)
49734                                   <1>
49735                                   <1> VIA_REG_STAT_STOPPED      equ   04h    ;; RWC
49736                                   <1> VIA_REG_STAT_EOL   equ   02h    ;; RWC
49737                                   <1> VIA_REG_STAT_FLAG  equ   01h    ;; RWC
49738                                   <1> VIA_REG_STAT_ACTIVE       equ   80h    ;; RO
49739                                   <1> ; 28/11/2016
```

```
49740                                  <1> VIA_REG_STAT_LAST  equ   40h    ;; RO
49741                                  <1> VIA_REG_STAT_TRIGGER_QUEUED equ 08h  ;; RO
49742                                  <1> VIA_REF_CTRL_INT_STOP     equ   04h  ; Interrupt on Current Index = Stop Index
49743                                  <1>                              ; and End of Block
49744                                  <1>
49745                                  <1> VIA_REG_OFFSET_CURR_COUNT equ 0Ch ;; dword - channel current count, index
49746                                  <1>
49747                                  <1> PORTB       EQU    061h
49748                                  <1> REFRESH_STATUS     EQU    010h   ; Refresh signal status
49749                                  <1>
49750                                  <1> ; AC97 Codec registers.
49751                                  <1>
49752                                  <1> ; each codec/mixer register is 16bits
49753                                  <1>
49754                                  <1> CODEC_RESET_REG                equ     00h  ; reset codec
49755                                  <1> CODEC_MASTER_VOL_REG           equ     02h  ; master volume
49756                                  <1> CODEC_HP_VOL_REG               equ     04h  ; headphone volume
49757                                  <1> CODEC_MASTER_MONO_VOL_REG      equ     06h  ; master mono volume
49758                                  <1> CODEC_MASTER_TONE_REG          equ     08h  ; master tone (R+L)
49759                                  <1> CODEC_PCBEEP_VOL_REG           equ     0Ah  ; PC beep volume
49760                                  <1> CODEC_PHONE_VOL_REG            equ     0Bh  ; phone volume
49761                                  <1> CODEC_MIC_VOL_REG              equ     0Eh  ; MIC volume
49762                                  <1> CODEC_LINE_IN_VOL_REG          equ     10h  ; line input volume
49763                                  <1> CODEC_CD_VOL_REG               equ     12h  ; CD volume
49764                                  <1> CODEC_VID_VOL_REG              equ     14h  ; video volume
49765                                  <1> CODEC_AUX_VOL_REG              equ     16h  ; aux volume
49766                                  <1> CODEC_PCM_OUT_REG              equ     18h  ; PCM output volume
49767                                  <1> CODEC_RECORD_SELECT_REG        equ     1Ah  ; record select input
49768                                  <1> CODEC_RECORD_VOL_REG           equ     1Ch  ; record volume
49769                                  <1> CODEC_RECORD_MIC_VOL_REG       equ     1Eh  ; record mic volume
49770                                  <1> CODEC_GP_REG                   equ     20h  ; general purpose
49771                                  <1> CODEC_3D_CONTROL_REG           equ     22h  ; 3D control
49772                                  <1> ; 24h is reserved
49773                                  <1> CODEC_POWER_CTRL_REG           equ     26h  ; powerdown control
49774                                  <1> CODEC_EXT_AUDIO_REG            equ     28h  ; extended audio
49775                                  <1> CODEC_EXT_AUDIO_CTRL_REG       equ     2Ah  ; extended audio control
49776                                  <1> CODEC_PCM_FRONT_DACRATE_REG    equ     2Ch  ; PCM out sample rate
49777                                  <1> CODEC_PCM_SURND_DACRATE_REG    equ     2Eh  ; surround sound sample rate
49778                                  <1> CODEC_PCM_LFE_DACRATE_REG      equ     30h  ; LFE sample rate
49779                                  <1> CODEC_LR_ADCRATE_REG           equ     32h  ; PCM in sample rate
49780                                  <1> CODEC_MIC_ADCRATE_REG          equ     34h  ; mic in sample rate
49781                                  <1>
49782                                  <1> ; VT8233 SGD bits (21/04/2017)
49783                                  <1> FLAG  EQU BIT30
49784                                  <1> EOL   EQU BIT31
49785                                  <1>
49786                                  <1> ; INTEL ICH EQUATES
49787                                  <1> ; 28/05/2017
49788                                  <1> INTEL_VID   equ   8086h  ; Intel's PCI vendor ID
49789                                  <1> ICH_DID           equ   2415h  ; ICH (82801AA) device ID
49790                                  <1> NAMBAR_REG     equ      10h    ; native audio mixer Base Address Register
49791                                  <1> NABMBAR_REG    equ      14h    ; native audio bus mastering Base Addr Reg
49792                                  <1>
49793                                  <1> PI_CR_REG      equ    0Bh     ; PCM in Control Register
49794                                  <1> PO_CR_REG   equ    1Bh     ; PCM out Control Register
49795                                  <1> MC_CR_REG   equ    2Bh     ; MIC in Control Register
49796                                  <1>
49797                                  <1> PI_SR_REG   equ    6       ; PCM in Status register
49798                                  <1> PO_SR_REG   equ    16h     ; PCM out Status register
49799                                  <1> MC_SR_REG   equ    26h     ; MIC in Status register
49800                                  <1>
49801                                  <1> IOCE        equ    BIT4    ; interrupt on complete enable.
49802                                  <1> FEIFE       equ    BIT3    ; set if you want an interrupt to fire
49803                                  <1> LVBIE       equ    BIT2    ; last valid buffer interrupt enable.
49804                                  <1> RR          equ    BIT1    ; reset registers.  Nukes all regs
49805                                  <1>                               ; except bits 4:2 of this register.
49806                                  <1>                               ; Only set this bit if BIT 0 is 0
49807                                  <1> RPBM        equ    BIT0    ; Run/Pause
49808                                  <1>                            ; set this bit to start the codec!
49809                                  <1>
49810                                  <1> PI_BDBAR_REG equ    0       ; PCM in buffer descriptor BAR
49811                                  <1> PO_BDBAR_REG equ    10h     ; PCM out buffer descriptor BAR
49812                                  <1> MC_BDBAR_REG equ    20h     ; MIC in buffer descriptor BAR
49813                                  <1>
49814                                  <1> PI_CIV_REG  equ    4       ; PCM in current Index value (RO)
49815                                  <1> PO_CIV_REG  equ    14h     ; PCM out current Index value (RO)
49816                                  <1> MC_CIV_REG  equ    24h     ; MIC in current Index value (RO)
49817                                  <1>
49818                                  <1> PI_LVI_REG  equ    5       ; PCM in Last Valid Index
49819                                  <1> PO_LVI_REG  equ    15h     ; PCM out Last Valid Index
49820                                  <1> MC_LVI_REG  equ    25h     ; MIC in Last Valid Index
49821                                  <1>
49822                                  <1> IOC         equ    BIT31; Fire an interrupt whenever this
49823                                  <1>                           ; buffer is complete.
49824                                  <1> BUP         equ    BIT30; Buffer Underrun Policy.
49825                                  <1>
49826                                  <1> GLOB_CNT_REG equ    2Ch     ; Global Control Register
49827                                  <1> GLOB_STS_REG equ    30h     ; Global Status register (RO)
49828                                  <1>
49829                                  <1> CTRL_ST_CREADY     equ   BIT8+BIT9+BIT28 ; Primary Codec Ready
49830                                  <1>
49831                                  <1> CODEC_REG_POWERDOWN   equ 26h
49832                                  <1> CODEC_REG_ST          equ 26h
49833                                  <1>
49834                                  <1> ; 22/06/2017
49835                                  <1> PO_PICB_REG  equ 18h      ; PCM Out Position In Current Buffer Register
49836                                  <1>
49837                                  <1> ;============================================================================
49838                                  <1> ;               CODE
49839                                  <1> ;============================================================================
49840                                  <1>
49841                                  <1> ; CODE for INTEL ICH AC'97 AUDIO CONTROLLER
49842                                  <1>
```

```
                                <1> DetectICH:
                                <1>       ; 10/06/2017
                                <1>       ; 05/06/2017
                                <1>       ; 29/05/2017
                                <1>       ; 28/05/2017
000113BB B886801524             <1>       mov     eax, (ICH_DID << 16) + INTEL_VID
000113C0 E876000000             <1>         call    pciFindDevice
000113C5 730D                   <1>         jnc     short d_ac97_1
                                <1> d_ac97_0:
                                <1> ; couldn't find the audio device!
000113C7 C3                     <1>       retn
                                <1>
                                <1> ; CODE for VIA VT8233 AUDIO CONTROLLER
                                <1>
                                <1> DetectVT8233:
                                <1>       ; 10/06/2017
                                <1>       ; 05/06/2017
                                <1>       ; 29/05/2017
                                <1>       ; 03/04/2017
000113C8 B806115930             <1>       mov     eax, (VT8233_DID << 16) + VIA_VID
000113CD E869000000             <1>         call    pciFindDevice
                                <1> ;       jnc     short d_vt8233_0
                                <1> ; couldn't find the audio device!
                                <1> ;     retn
000113D2 72F3                   <1>       jc    short d_ac97_0  ; 28/05/2017
                                <1> d_vt8233_0:
                                <1>       ; 24/03/2017 ('player.asm')
                                <1>       ; 12/11/2016
                                <1>       ; Erdogan Tan - 8/11/2016
                                <1>       ; References: Kolibrios - vt823x.asm (2016)
                                <1>       ;           VIA VT8235 V-Link South Bridge (VT8235-VIA.PDF)(2002)
                                <1>       ;           lowlevel.eu - AC97 (2016)
                                <1>       ;           .wav player for DOS by Jeff Leyda (2002) -this file-
                                <1>       ;           Linux kernel - via82xx.c (2016)
                                <1> d_ac97_1:
                                <1>       ; eax = BUS/DEV/FN
                                <1>       ;     00000000BBBBBBBBDDDDDFFF00000000
                                <1>       ; edx = DEV/VENDOR
                                <1>       ;     DDDDDDDDDDDDDDDDVVVVVVVVVVVVVVVV
                                <1>
000113D4 A3[AC650100]           <1>       mov   [audio_dev_id], eax
000113D9 8915[B0650100]         <1>       mov   [audio_vendor], edx
                                <1>
                                <1>       ; init controller
000113DF B004                   <1>       mov   al, PCI_CMD_REG ; command register (04h)
000113E1 E8E2000000             <1>       call  pciRegRead32
                                <1>
                                <1>       ; eax = BUS/DEV/FN/REG
                                <1>       ; edx = STATUS/COMMAND
                                <1>       ;     SSSSSSSSSSSSSSSSCCCCCCCCCCCCCCCC
000113E6 8915[B4650100]         <1>       mov   [audio_stats_cmd], edx
                                <1>
000113EC B010                   <1>       mov   al, PCI_IO_BASE ; IO base address register (10h)
                                <1>       ;mov  al, NAMBAR_REG      ; Native Audio Mixer BAR (10h)
000113EE E8D5000000             <1>       call  pciRegRead32
                                <1>
000113F3 66813D[B0650100]86-    <1>       cmp   word [audio_vendor], 8086h ; AC'97 ?
000113FB 80                     <1>
000113FC 751F                   <1>       jne   short d_vt8233_1
                                <1>
000113FE 6683E2FE               <1>       and   dx, 0FFFEh ; Audio Codec IO_ADDR_MASK
00011402 668915[DC650100]       <1>       mov   [NAMBAR], dx
                                <1>
00011409 B014                   <1>       mov   al, NABMBAR_REG ; Native Audio Bus Mastering BAR (14h)
0001140B E8B8000000             <1>       call  pciRegRead32
                                <1>
00011410 6683E2C0               <1>       and   dx, 0FFC0h ; Audio Controller IO_ADDR_MASK
00011414 668915[DE650100]       <1>       mov   [NABMBAR], dx
                                <1>         ;mov [audio_io_base], dx
                                <1>
0001141B EB0B                   <1>       jmp   short d_ac97_2
                                <1>
                                <1> d_vt8233_1:
0001141D 6683E2C0               <1>       and     dx, 0FFC0h ; Audio Controller IO_ADDR_MASK
00011421 668915[AA650100]       <1>         mov     [audio_io_base], dx
                                <1>
                                <1> d_ac97_2:
                                <1>       ; 10/06/2017
00011428 B03C                   <1>       mov   al, AC97_INT_LINE ; Interrupt Line Register (3Ch)
                                <1>       ;call pciRegRead32
0001142A E886000000             <1>       call  pciRegRead8
                                <1>
                                <1>       ;and  edx, 0FFh
0001142F 6681E2FF00             <1>       and   dx, 0FFh
                                <1>
00011434 8815[A7650100]         <1>       mov     [audio_intr], dl
                                <1>
0001143A C3                     <1>       retn
                                <1>
                                <1>       ;; (Note: Interrupts are already enabled by TRDOS 386 kernel!)
                                <1>       ;mov  cx, dx
                                <1>
                                <1>       ;in   al, 0A1h ; irq 8-15
                                <1>       ;mov  ah, al
                                <1>       ;in   al, 21h  ; irq 0-7
                                <1>       ;btr  ax, dx  ; unmask ; 17/03/2017
                                <1>       ;;bts ax, dx   ; MASK interrupt ; 10/06/2017
                                <1>       ;out  21h, al  ; irq <= 7
                                <1>       ;mov  al, ah
                                <1>       ;out  0A1h, al ; irq > 7
                                <1>       ;
                                <1>
                                <1>       ; 10/06/2017
```

```
49946                                  <1>       ; === Intel ICH I/O Controller Hub Datasheet, Section 8.1.16 ===
49947                                  <1>       ; PRQ[n]_ROUT Register (61h, PRQB) Bit 7:
49948                                  <1>       ; Interrupt Routing Enable (IRQEN).
49949                                  <1>       ; 0 = The corresponding PIRQ is routed to one of the ISA-compatible
49950                                  <1>       ;     interrupts specified in bits[3:0].
49951                                  <1>       ; 1 = The PIRQ is not routed to the 8259.
49952                                  <1>       ; Note: If the PIRQ is intended to cause an interrupt to the ICH's
49953                                  <1>       ;     integrated I/O APIC, then this bit should be set to 0 and
49954                                  <1>       ;     the APIC_EN bit should be set to 1.
49955                                  <1>       ;     The IRQEN must be set to 0 and the PIRQ routed to
49956                                  <1>       ;     an 8259 interrupt via the IRQ Routing filed (bits[3:0).
49957                                  <1>       ;     The corresponding 8259 interrupt must be masked via the
49958                                  <1>       ;     appropriated bit in the 8259's OCW1 (Interrupt Mask)
49959                                  <1>       ;     register. The IOAPIC must then be enabled by setting
49960                                  <1>       ;     the APIC_EN bit in the GEN_CNTL register.
49961                                  <1>
49962                                  <1>       ;mov   eax, 0F861h  ; D31:F0
49963                                  <1>             ;AL=61h : PIRQ[B] Routing Control Reg, LPC interface
49964                                  <1>       ;;mov  dl,  [audio_intr]
49965                                  <1>       ;call  pciRegWrite8
49966                                  <1>       ;;mov  al, 0D0h ; General Control Register (GEN_CTL)
49967                                  <1>       ;;call pciRegRead32
49968                                  <1>       ;;or   edx, 100h ; Bit 8, APIC_EN (Enable I/O APIC)
49969                                  <1>       ;;;call      pciRegWrite32
49970                                  <1>       ;;and  edx, ~100h
49971                                  <1>       ;;call pciRegWrite32 ; ; Bit 8, APIC_EN (Disable I/O APIC)
49972                                  <1>       ;
49973                                  <1>
49974                                  <1>       ;mov   dx, 4D1h     ; 8259 ELCR2
49975                                  <1>       ;in    al, dx
49976                                  <1>       ;mov   ah, al
49977                                  <1>       ;;mov  dx, 4D0h     ; 8259 ELCR1
49978                                  <1>       ;dec   dl
49979                                  <1>       ;in    al, dx
49980                                  <1>       ;bts   ax, cx
49981                                  <1>       ;;mov  dx, 4D0h
49982                                  <1>       ;out   dx, al       ; set level-triggered mode
49983                                  <1>       ;mov   al, ah ; 29/05/2017
49984                                  <1>       ;;mov  dx, 4D1h
49985                                  <1>       ;inc   dl
49986                                  <1>       ;out   dx, al       ; set level-triggered mode
49987                                  <1>
49988                                  <1>       ;xor   eax, eax ; 0
49989                                  <1>
49990                                  <1>       ;retn
49991                                  <1>
49992                                  <1> ; CODE for PCI
49993                                  <1>
49994                                  <1> pciFindDevice:
49995                                  <1>       ; 03/04/2017 ('pci.asm', 20/03/2017)
49996                                  <1>       ;
49997                                  <1>       ; scan through PCI space looking for a device+vendor ID
49998                                  <1>       ;
49999                                  <1>       ; Entry: EAX=Device+Vendor ID
50000                                  <1>       ;
50001                                  <1>       ; Exit: EAX=PCI address if device found
50002                                  <1>       ;      EDX=Device+Vendor ID
50003                                  <1>       ;       CY clear if found, set if not found. EAX invalid if CY set.
50004                                  <1>       ;
50005                                  <1>       ; Destroys: ebx, esi, edi, cl
50006                                  <1>       ;
50007                                  <1>
50008                                  <1>       ;push  ecx
50009 0001143B 50                      <1>       push   eax
50010                                  <1>       ;push  esi
50011                                  <1>       ;push  edi
50012                                  <1>
50013 0001143C 89C6                    <1>         mov     esi, eax                ; save off vend+device ID
50014 0001143E BF00FFFF7F              <1>         mov     edi, (80000000h - 100h) ; start with bus 0, dev 0 func 0
50015                                  <1>
50016                                  <1> nextPCIdevice:
50017 00011443 81C700010000            <1>         add     edi, 100h
50018 00011449 81FF00F8FF80            <1>         cmp     edi, 80FFF800h         ; scanned all devices?
50019 0001144F F9                      <1>         stc
50020 00011450 740C                    <1>         je      short PCIScanExit       ; not found
50021                                  <1>
50022 00011452 89F8                    <1>         mov     eax, edi                ; read PCI registers
50023 00011454 E86F000000              <1>         call    pciRegRead32
50024 00011459 39F2                    <1>         cmp     edx, esi                ; found device?
50025 0001145B 75E6                    <1>         jne     short nextPCIdevice
50026 0001145D F8                      <1>         clc
50027                                  <1>
50028                                  <1> PCIScanExit:
50029 0001145E 9C                      <1>       pushf
50030 0001145F B8FFFFFF7F              <1>       mov    eax, NOT_BIT31      ; 19/03/2017
50031 00011464 21F8                    <1>       and    eax, edi     ; return only bus/dev/fn #
50032 00011466 9D                      <1>       popf
50033                                  <1>
50034                                  <1>       ;pop   edi
50035                                  <1>       ;pop   esi
50036 00011467 5A                      <1>       pop    edx
50037                                  <1>       ;pop   ecx
50038 00011468 C3                      <1>       retn
50039                                  <1>
50040                                  <1> pciRegRead:
50041                                  <1>       ; 03/04/2017 ('pci.asm', 20/03/2017)
50042                                  <1>       ;
50043                                  <1>       ; 8/16/32bit PCI reader
50044                                  <1>       ;
50045                                  <1>       ; Entry: EAX=PCI Bus/Device/fn/register number
50046                                  <1>       ;           BIT30 set if 32 bit access requested
50047                                  <1>       ;           BIT29 set if 16 bit access requested
50048                                  <1>       ;           otherwise defaults to 8 bit read
```

```
50049                                  <1>      ;
50050                                  <1>      ; Exit:  DL,DX,EDX register data depending on requested read size
50051                                  <1>      ;
50052                                  <1>      ; Note1: this routine is meant to be called via pciRegRead8,
50053                                  <1>      ;      pciRegread16 or pciRegRead32, listed below.
50054                                  <1>      ;
50055                                  <1>      ; Note2: don't attempt to read 32 bits of data from a non dword
50056                                  <1>      ;      aligned reg number.  Likewise, don't do 16 bit reads from
50057                                  <1>      ;      non word aligned reg #
50058                                  <1>
50059 00011469 53                      <1>      push   ebx
50060 0001146A 51                      <1>      push   ecx
50061 0001146B 89C3                    <1>        mov     ebx, eax        ; save eax, dh
50062 0001146D 88F1                    <1>        mov     cl, dh
50063                                  <1>
50064 0001146F 25FFFFFF3F              <1>        and     eax, NOT_PCI32_PCI16   ; clear out data size request
50065 00011474 0D00000080              <1>        or      eax, BIT31             ; make a PCI access request
50066 00011479 24FC                    <1>        and     al, ~3 ; NOT 3         ; force index to be dword
50067                                  <1>
50068 0001147B 66BAF80C                <1>        mov     dx, PCI_INDEX_PORT
50069 0001147F EF                      <1>        out  dx, eax                   ; write PCI selector
50070                                  <1>
50071 00011480 66BAFC0C                <1>        mov     dx, PCI_DATA_PORT
50072 00011484 88D8                    <1>        mov     al, bl
50073 00011486 2403                    <1>        and     al, 3                  ; figure out which port to
50074 00011488 00C2                    <1>        add     dl, al                 ; read to
50075                                  <1>
50076 0001148A F7C3000000C0            <1>      test    ebx, PCI32+PCI16
50077 00011490 7507                    <1>        jnz     short _pregr0
50078 00011492 EC                      <1>      in    al, dx             ; return 8 bits of data
50079 00011493 88C2                    <1>        mov  dl, al
50080 00011495 88CE                    <1>      mov     dh, cl                 ; restore dh for 8 bit read
50081 00011497 EB12                    <1>      jmp    short _pregr2
50082                                  <1> _pregr0:
50083 00011499 F7C300000080            <1>      test    ebx, PCI32
50084 0001149F 7507                    <1>        jnz  short _pregr1
50085 000114A1 66ED                    <1>      in     ax, dx
50086 000114A3 6689C2                  <1>        mov     dx, ax                 ; return 16 bits of data
50087 000114A6 EB03                    <1>      jmp    short _pregr2
50088                                  <1> _pregr1:
50089 000114A8 ED                      <1>      in     eax, dx                 ; return 32 bits of data
50090 000114A9 89C2                    <1>      mov    edx, eax
50091                                  <1> _pregr2:
50092 000114AB 89D8                    <1>      mov     eax, ebx          ; restore eax
50093 000114AD 25FFFFFF3F              <1>        and     eax, NOT_PCI32_PCI16   ; clear out data size request
50094 000114B2 59                      <1>      pop    ecx
50095 000114B3 5B                      <1>      pop    ebx
50096 000114B4 C3                      <1>      retn
50097                                  <1>
50098                                  <1> pciRegRead8:
50099 000114B5 25FFFFFF3F              <1>        and     eax, NOT_PCI32_PCI16   ; set up 8 bit read size
50100 000114BA EBAD                    <1>        jmp     short pciRegRead; call generic PCI access
50101                                  <1>
50102                                  <1> pciRegRead16:
50103 000114BC 25FFFFFF3F              <1>        and     eax, NOT_PCI32_PCI16   ; set up 16 bit read size
50104 000114C1 0D00000040              <1>        or      eax, PCI16             ; call generic PCI access
50105 000114C6 EBA1                    <1>        jmp     short pciRegRead
50106                                  <1>
50107                                  <1> pciRegRead32:
50108 000114C8 25FFFFFF3F              <1>        and     eax, NOT_PCI32_PCI16   ; set up 32 bit read size
50109 000114CD 0D00000080              <1>        or      eax, PCI32             ; call generic PCI access
50110 000114D2 EB95                    <1>        jmp     pciRegRead
50111                                  <1>
50112                                  <1> pciRegWrite:
50113                                  <1>      ; 03/04/2017 ('pci.asm', 29/11/2016)
50114                                  <1>      ;
50115                                  <1>      ; 8/16/32bit PCI writer
50116                                  <1>      ;
50117                                  <1>      ; Entry: EAX=PCI Bus/Device/fn/register number
50118                                  <1>      ;          BIT31 set if 32 bit access requested
50119                                  <1>      ;          BIT30 set if 16 bit access requested
50120                                  <1>      ;          otherwise defaults to 8bit read
50121                                  <1>      ;        DL/DX/EDX data to write depending on size
50122                                  <1>      ;
50123                                  <1>      ; Note1: this routine is meant to be called via pciRegWrite8,
50124                                  <1>      ;      pciRegWrite16 or pciRegWrite32 as detailed below.
50125                                  <1>      ;
50126                                  <1>      ; Note2: don't attempt to write 32bits of data from a non dword
50127                                  <1>      ;      aligned reg number.  Likewise, don't do 16 bit writes from
50128                                  <1>      ;      non word aligned reg #
50129                                  <1>
50130 000114D4 53                      <1>      push   ebx
50131 000114D5 51                      <1>      push   ecx
50132 000114D6 89C3                    <1>        mov     ebx, eax        ; save eax, edx
50133 000114D8 89D1                    <1>        mov     ecx, edx
50134 000114DA 25FFFFFF3F              <1>      and     eax, NOT_PCI32_PCI16     ; clear out data size request
50135 000114DF 0D00000080              <1>        or      eax, BIT31             ; make a PCI access request
50136 000114E4 24FC                    <1>        and     al, ~3 ; NOT 3         ; force index to be dword
50137                                  <1>
50138 000114E6 66BAF80C                <1>        mov     dx, PCI_INDEX_PORT
50139 000114EA EF                      <1>        out  dx, eax                   ; write PCI selector
50140                                  <1>
50141 000114EB 66BAFC0C                <1>        mov     dx, PCI_DATA_PORT
50142 000114EF 88D8                    <1>        mov     al, bl
50143 000114F1 2403                    <1>        and     al, 3                  ; figure out which port to
50144 000114F3 00C2                    <1>        add     dl, al                 ; write to
50145                                  <1>
50146 000114F5 F7C3000000C0            <1>      test    ebx, PCI32+PCI16
50147 000114FB 7505                    <1>        jnz     short _pregw0
50148 000114FD 88C8                    <1>      mov    al, cl                  ; put data into al
50149 000114FF EE                      <1>      out    dx, al
50150 00011500 EB12                    <1>      jmp    short _pregw2
50151                                  <1> _pregw0:
```

```
50152 00011502 F7C300000080        <1>       test    ebx, PCI32
50153 00011508 7507                <1>         jnz     short _pregw1
50154 0001150A 6689C8              <1>       mov    ax, cx             ; put data into ax
50155 0001150D 66EF                <1>       out    dx, ax
50156 0001150F EB03                <1>       jmp    short _pregw2
50157                              <1> _pregw1:
50158 00011511 89C8                <1>       mov    eax, ecx           ; put data into eax
50159 00011513 EF                  <1>       out    dx, eax
50160                              <1> _pregw2:
50161 00011514 89D8                <1>         mov     eax, ebx        ; restore eax
50162 00011516 25FFFFFF3F          <1>         and     eax, NOT_PCI32_PCI16   ; clear out data size request
50163 0001151B 89CA                <1>         mov     edx, ecx        ; restore dx
50164 0001151D 59                  <1>       pop    ecx
50165 0001151E 5B                  <1>       pop    ebx
50166 0001151F C3                  <1>       retn
50167                              <1>
50168                              <1> pciRegWrite8:
50169 00011520 25FFFFFF3F          <1>         and     eax, NOT_PCI32_PCI16   ; set up 8 bit write size
50170 00011525 EBAD                <1>         jmp  short pciRegWrite   ; call generic PCI access
50171                              <1>
50172                              <1> pciRegWrite16:
50173 00011527 25FFFFFF3F          <1>         and     eax, NOT_PCI32_PCI16   ; set up 16 bit write size
50174 0001152C 0D00000040          <1>         or      eax, PCI16             ; call generic PCI access
50175 00011531 EBA1                <1>         jmp  short pciRegWrite
50176                              <1>
50177                              <1> pciRegWrite32:
50178 00011533 25FFFFFF3F          <1>         and     eax, NOT_PCI32_PCI16   ; set up 32 bit write size
50179 00011538 0D00000080          <1>         or      eax, PCI32             ; call generic PCI access
50180 0001153D EB95                <1>         jmp  pciRegWrite
50181                              <1>
50182                              <1> init_codec:
50183                              <1>       ; 05/06/2017
50184                              <1>       ; 28/05/2017 - Erdogan Tan (Ref: KolibriOS, vt823x.asm)
50185                              <1>       ;
50186 0001153F A1[AC650100]        <1>       mov    eax, [audio_dev_id]
50187 00011544 B041                <1>       mov    al, VIA_ACLINK_CTRL
50188 00011546 E86AFFFFFF          <1>       call   pciRegRead8
50189                              <1>       ; ?
50190 0001154B B040                <1>       mov    al, VIA_ACLINK_STAT
50191 0001154D E863FFFFFF          <1>       call   pciRegRead8
50192 00011552 F6C201              <1>       test   dl, VIA_ACLINK_C00_READY
50193 00011555 7508                <1>         jnz     short _codec_ready_1
50194 00011557 E80E000000          <1>       call   reset_codec
50195 0001155C 7306                <1>       jnc    short _codec_ready_2 ; eax = 1
50196 0001155E C3                  <1>       retn
50197                              <1> _codec_ready_1:
50198 0001155F B801000000          <1>       mov    eax, 1
50199                              <1> _codec_ready_2:
50200 00011564 E87A000000          <1>       call   codec_io_w16
50201                              <1> detect_codec:
50202 00011569 C3                  <1>       retn
50203                              <1>
50204                              <1> reset_codec:
50205                              <1>       ; 16/04/2017
50206                              <1>       ; 23/03/2017
50207                              <1>       ; ('codec.asm')
50208                              <1>       ; 12/11/2016 - Erdogan Tan (Ref: KolibriOS, vt823x.asm)
50209 0001156A A1[AC650100]        <1>       mov    eax, [audio_dev_id]
50210 0001156F B041                <1>       mov    al, VIA_ACLINK_CTRL
50211 00011571 B2E0                <1>              mov    dl, VIA_ACLINK_CTRL_ENABLE + VIA_ACLINK_CTRL_RESET + VIA_ACLINK_CTRL_SYNC
50212 00011573 E8A8FFFFFF          <1>       call   pciRegWrite8
50213                              <1>
50214 00011578 E83D000000          <1>       call   delay_100ms  ; wait 100 ms
50215                              <1> _rc_cold:
50216 0001157D E808000000          <1>         call    cold_reset
50217 00011582 7301                <1>         jnc     short _reset_codec_ok
50218                              <1>
50219                              <1>       ; 16/04/2017
50220                              <1>         ;xor     eax, eax         ; timeout error
50221                              <1>             ;stc
50222 00011584 C3                  <1>       retn
50223                              <1>
50224                              <1> _reset_codec_ok:
50225 00011585 31C0                <1>         xor     eax, eax
50226                              <1>         ;mov al, VIA_ACLINK_C00_READY ; 1
50227 00011587 FEC0                <1>         inc  al
50228 00011589 C3                  <1>       retn
50229                              <1>
50230                              <1> cold_reset:
50231                              <1>       ; 16/04/2017
50232                              <1>       ; 23/03/2017
50233                              <1>       ; ('codec.asm')
50234                              <1>       ; 12/11/2016 - Erdogan Tan (Ref: KolibriOS, vt823x.asm)
50235                              <1>       ;mov   eax, [audio_dev_id]
50236                              <1>       ;mov   al, VIA_ACLINK_CTRL
50237 0001158A 30D2                <1>       xor    dl, dl ; 0
50238 0001158C E88FFFFFFF          <1>       call   pciRegWrite8
50239                              <1>
50240 00011591 E824000000          <1>       call   delay_100ms  ; wait 100 ms
50241                              <1>
50242                              <1>       ;; ACLink on, deassert ACLink reset, VSR, SGD data out
50243                              <1>         ;; note - FM data out has trouble with non VRA codecs !!
50244                              <1>
50245                              <1>       ;mov   eax, [audio_dev_id]
50246                              <1>       ;mov   al, VIA_ACLINK_CTRL
50247 00011596 B2CC                <1>       mov    dl, VIA_ACLINK_CTRL_INIT
50248 00011598 E883FFFFFF          <1>       call   pciRegWrite8
50249                              <1>
50250 0001159D B910000000          <1>       mov    ecx, 16      ; total 2s
50251                              <1>
50252                              <1> _crst_wait:
50253                              <1>       ;mov   eax, [audio_dev_id]
50254 000115A2 B040                <1>       mov    al, VIA_ACLINK_STAT
```

```
50255 000115A4 E80CFFFFFF          <1>       call   pciRegRead8
50256                              <1>
50257 000115A9 F6C201              <1>         test    dl, VIA_ACLINK_C00_READY
50258 000115AC 750B                <1>         jnz     short _crst_ok
50259                              <1>
50260 000115AE 51                  <1>       push   ecx
50261 000115AF E806000000          <1>       call   delay_100ms
50262 000115B4 59                  <1>       pop    ecx
50263                              <1>
50264 000115B5 49                  <1>         dec     ecx
50265 000115B6 75EA                <1>         jnz     short _crst_wait
50266                              <1>
50267                              <1> _crst_fail:
50268 000115B8 F9                  <1>         stc
50269                              <1> _crst_ok:
50270 000115B9 C3                  <1>       retn
50271                              <1>
50272                              <1> delay_100ms:
50273                              <1>       ; 29/05/2017
50274                              <1>       ; 24/03/2017 ('codec.asm')
50275                              <1>       ; wait 100 ms
50276 000115BA B990010000          <1>       mov    ecx, 400  ; 400*0.25ms
50277                              <1> _delay_x_ms:
50278 000115BF E803000000          <1>       call   delay1_4ms
50279 000115C4 E2F9                <1>         loop _delay_x_ms
50280 000115C6 C3                  <1>       retn
50281                              <1>
50282                              <1> ;       delay1_4ms - Delay for 1/4 millisecond.
50283                              <1> ;         1mS = 1000us
50284                              <1> ;       Entry:
50285                              <1> ;         None
50286                              <1> ;       Exit:
50287                              <1> ;       None
50288                              <1> ;
50289                              <1> ;       Modified:
50290                              <1> ;         None
50291                              <1> ;
50292                              <1>
50293                              <1>       ; 29/05/2017
50294                              <1>       ; 23/04/2017
50295                              <1>       ; 05/03/2017 (TRDOS 386)
50296                              <1>       ; ('UTILS.ASM')
50297                              <1> delay1_4ms:
50298 000115C7 50                  <1>         push    eax
50299 000115C8 51                  <1>         push    ecx
50300 000115C9 B110                <1>         mov  cl, 16       ; close enough.
50301                              <1>
50302 000115CB E461                <1>       in     al, PORTB ; 61h
50303                              <1>
50304 000115CD 2410                <1>       and    al, REFRESH_STATUS ; 10h
50305 000115CF 88C5                <1>       mov    ch, al       ; Start toggle state
50306                              <1> _d4ms1:
50307 000115D1 E461                <1>       in     al, PORTB    ; Read system control port
50308                              <1>
50309 000115D3 2410                <1>       and    al, REFRESH_STATUS ; Refresh toggles 15.085 microseconds
50310 000115D5 38C5                <1>       cmp    ch, al
50311 000115D7 74F8                <1>       je     short _d4ms1 ; Wait for state change
50312                              <1>
50313 000115D9 88C5                <1>       mov    ch, al       ; Update with new state
50314 000115DB FEC9                <1>       dec    cl
50315 000115DD 75F2                <1>       jnz    short _d4ms1
50316                              <1>
50317 000115DF F8                  <1>       clc    ; 29/05/2017
50318                              <1>
50319 000115E0 59                  <1>         pop     ecx
50320 000115E1 58                  <1>         pop     eax
50321 000115E2 C3                  <1>         retn
50322                              <1>
50323                              <1> ; 10/04/2017 (TRDOS 386)
50324                              <1> ; 12/11/2016
50325                              <1>
50326                              <1> codec_io_w16: ;w32
50327                              <1>       ; ('codec.asm')
50328 000115E3 668B15[AA650100]    <1>         mov  dx, [audio_io_base]
50329 000115EA 6681C28000          <1>         add     dx, VIA_REG_AC97
50330 000115EF EF                  <1>       out    dx, eax
50331 000115F0 C3                  <1>         retn
50332                              <1>
50333                              <1> codec_io_r16: ;r32
50334                              <1>       ; ('codec.asm')
50335 000115F1 668B15[AA650100]    <1>         mov     dx, [audio_io_base]
50336 000115F8 6681C28000          <1>         add     dx, VIA_REG_AC97
50337 000115FD ED                  <1>         in   eax, dx
50338 000115FE C3                  <1>         retn
50339                              <1>
50340                              <1> ctrl_io_w8:
50341                              <1>       ; ('codec.asm')
50342 000115FF 660315[AA650100]    <1>         add     dx, [audio_io_base]
50343 00011606 EE                  <1>         out  dx, al
50344 00011607 C3                  <1>         retn
50345                              <1>
50346                              <1> ctrl_io_r8:
50347                              <1>       ; ('codec.asm')
50348 00011608 660315[AA650100]    <1>         add     dx, [audio_io_base]
50349 0001160F EC                  <1>         in   al, dx
50350 00011610 C3                  <1>         retn
50351                              <1>
50352                              <1> ctrl_io_w32:
50353                              <1>       ; ('codec.asm')
50354 00011611 660315[AA650100]    <1>         add     dx, [audio_io_base]
50355 00011618 EF                  <1>         out  dx, eax
50356 00011619 C3                  <1>         retn
50357                              <1>
```

```
50358                                  <1> ctrl_io_r32:
50359                                  <1>       ; ('codec.asm')
50360 0001161A 660315[AA650100]          <1>       add  dx, [audio_io_base]
50361 00011621 ED                        <1>       in     eax, dx
50362 00011622 C3                        <1>       retn
50363                                  <1>
50364                                  <1> codec_read:
50365                                  <1>       ; 12/11/2016 - Erdogan Tan (Ref: KolibriOS, vt823x.asm)
50366                                  <1>       ; Use only primary codec.
50367                                  <1>       ; eax = register
50368 00011623 C1E010                    <1>       shl     eax, VIA_REG_AC97_CMD_SHIFT
50369 00011626 0D00008002                <1>       or      eax, VIA_REG_AC97_PRIMARY_VALID + VIA_REG_AC97_READ
50370                                  <1>
50371 0001162B E8B3FFFFFF                <1>       call    codec_io_w16
50372                                  <1>
50373                                  <1>             ; codec_valid
50374 00011630 E831000000                <1>       call   codec_check_ready
50375 00011635 7301                      <1>       jnc  short _cr_ok
50376                                  <1>
50377 00011637 C3                        <1>       retn
50378                                  <1>
50379                                  <1> _cr_ok:
50380                                  <1>       ; wait 25 ms
50381 00011638 B950000000                <1>       mov    ecx, 80 ; (100*0.25 ms)
50382                                  <1> _cr_wloop:
50383 0001163D E885FFFFFF                <1>       call   delay1_4ms
50384 00011642 E2F9                      <1>       loop   _cr_wloop
50385                                  <1>
50386 00011644 E8A8FFFFFF                <1>       call    codec_io_r16
50387 00011649 25FFFF0000                <1>       and     eax, 0FFFFh
50388 0001164E C3                        <1>       retn
50389                                  <1>
50390                                  <1> codec_write:
50391                                  <1>       ; 12/11/2016 - Erdogan Tan (Ref: KolibriOS, vt823x.asm)
50392                                  <1>       ; Use only primary codec.
50393                                  <1>
50394                                  <1>       ; eax = data (volume)
50395                                  <1>       ; edx = register (mixer register)
50396                                  <1>
50397 0001164F C1E210                    <1>       shl     edx, VIA_REG_AC97_CMD_SHIFT
50398                                  <1>
50399 00011652 C1E000                    <1>       shl     eax, VIA_REG_AC97_DATA_SHIFT ; shl eax, 0
50400 00011655 09C2                      <1>       or      edx, eax
50401                                  <1>
50402 00011657 B800000000                <1>       mov     eax, VIA_REG_AC97_CODEC_ID_PRIMARY
50403 0001165C C1E01E                    <1>       shl     eax, VIA_REG_AC97_CODEC_ID_SHIFT
50404 0001165F 09D0                      <1>       or      eax, edx
50405                                  <1>
50406 00011661 E87DFFFFFF                <1>       call    codec_io_w16
50407                                  <1>       ;mov    [codec.regs+esi], ax
50408                                  <1>
50409                                  <1>       ;call      codec_check_ready
50410                                  <1>           ;retn
50411                                  <1>       ;jmp   short _codec_check_ready
50412                                  <1>
50413                                  <1> codec_check_ready:
50414                                  <1>       ; 12/11/2016 - Erdogan Tan (Ref: KolibriOS, vt823x.asm)
50415                                  <1>
50416                                  <1> _codec_check_ready:
50417 00011666 B914000000                <1>       mov    ecx, 20      ; total 2s
50418                                  <1> _ccr_wait:
50419 0001166B 51                        <1>       push   ecx
50420                                  <1>
50421 0001166C E880FFFFFF                <1>       call    codec_io_r16
50422 00011671 A900000001                <1>       test    eax, VIA_REG_AC97_BUSY
50423 00011676 740B                      <1>       jz      short _ccr_ok
50424                                  <1>
50425 00011678 E83DFFFFFF                <1>       call   delay_100ms
50426                                  <1>
50427 0001167D 59                        <1>       pop    ecx
50428                                  <1>
50429 0001167E 49                        <1>       dec     ecx
50430 0001167F 75EA                      <1>       jnz     short _ccr_wait
50431                                  <1>
50432 00011681 F9                        <1>       stc
50433 00011682 C3                        <1>       retn
50434                                  <1>
50435                                  <1> _ccr_ok:
50436 00011683 59                        <1>       pop    ecx
50437 00011684 25FFFF0000                <1>       and     eax, 0FFFFh
50438 00011689 C3                        <1>       retn
50439                                  <1>
50440                                  <1> codec_config:
50441                                  <1>       ; 10/06/2017
50442                                  <1>       ; 29/05/2017
50443                                  <1>       ; 24/04/2017
50444                                  <1>       ; 21/04/2017
50445                                  <1>       ; 16/04/2017 (TRDOS 386 Kernel)
50446                                  <1>       ; 15/11/2016 ('codec.asm', 'player.com')
50447                                  <1>       ; 14/11/2016
50448                                  <1>       ; 12/11/2016 - Erdogan Tan
50449                                  <1>       ;          (Ref: KolibriOS, 'setup_codec', codec.inc)
50450                                  <1>
50451 0001168A B802020000                <1>       mov     eax, 0202h
50452 0001168F 66A3[DA650100]            <1>       mov    [audio_master_volume], ax
50453 00011695 66B81F1F                  <1>       mov    ax, 1F1Fh ; 31,31
50454 00011699 BA02000000                <1>       mov    edx, CODEC_MASTER_VOL_REG ; 02h ; Line Out
50455 0001169E E8ACFFFFFF                <1>       call   codec_write
50456                                  <1>       ;jc    short cconfig_error
50457                                  <1>
50458                                  <1>       ;mov    eax, 0202h
50459 000116A3 66B80202                  <1>       mov     ax, 0202h
50460 000116A7 BA18000000                <1>       mov    edx, CODEC_PCM_OUT_REG ; 18h ; Wave Output (Stereo)
```

```
50461 000116AC E89EFFFFFF          <1>       call   codec_write
50462                              <1>       ;jc    short cconfig_error
50463                              <1>
50464                              <1>       ;mov    eax, 0202h
50465 000116B1 66B80202            <1>       mov    ax, 0202h
50466 000116B5 BA04000000          <1>       mov    edx, CODEC_AUX_VOL ; 04h ; CODEC_HP_VOL_REG ; HeadPhone
50467 000116BA E890FFFFFF          <1>       call   codec_write
50468                              <1>       ;jc    short cconfig_error
50469                              <1>
50470                              <1>       ;mov    eax, 08h
50471                              <1>         ;mov    ax,  08h
50472 000116BF 66B80880            <1>       mov    ax, 8008h ; Mute
50473 000116C3 BA0C000000          <1>       mov    edx, 0Ch  ; AC97_PHONE_VOL ; TAD Input (Mono)
50474 000116C8 E882FFFFFF          <1>       call   codec_write
50475                              <1>       ;jc    short cconfig_error
50476                              <1>
50477                              <1>       ;mov    eax, 0808h
50478 000116CD 66B80808            <1>         mov     ax,  0808h
50479 000116D1 BA10000000          <1>         mov  edx, CODEC_LINE_IN_VOL_REG ; 10h ; Line Input (Stereo)
50480 000116D6 E874FFFFFF          <1>       call   codec_write
50481                              <1>       ;jc    short cconfig_error
50482                              <1>
50483                              <1>       ;mov    eax, 0808h
50484 000116DB 66B80808            <1>       mov     ax,  0808h
50485 000116DF BA12000000          <1>         mov  edx, CODEC_CD_VOL_REG ; 12h ; CR Input (Stereo)
50486 000116E4 E866FFFFFF          <1>       call   codec_write
50487                              <1>       ;jc    short cconfig_error
50488                              <1>
50489                              <1>       ;mov    eax, 0808h
50490 000116E9 66B80808            <1>       mov     ax,  0808h
50491 000116ED BA16000000          <1>         mov  edx, CODEC_AUX_VOL_REG ; 16h ; Aux Input (Stereo)
50492                              <1>       ;call  codec_write
50493                              <1>       ;;jc   short cconfig_error
50494 000116F2 E958FFFFFF          <1>       jmp    codec_write ; 10/06/2017
50495                              <1>
50496                              <1> ;     ; Extended Audio Status (2Ah)
50497                              <1> ;     mov    eax, CODEC_EXT_AUDIO_CTRL_REG ; 2Ah
50498                              <1> ;     call   codec_read
50499                              <1> ;     and     eax, 0FFFFh - 2          ; clear DRA (BIT1)
50500                              <1> ;     ;or     eax, 1                   ; set VRA (BIT0)
50501                              <1> ;     or     eax, 5       ; VRA (BIT0) & S/PDIF (BIT2) ; 14/11/2016
50502                              <1> ;     mov    edx, CODEC_EXT_AUDIO_CTRL_REG
50503                              <1> ;     call   codec_write
50504                              <1> ;     ;jc    short cconfig_error
50505                              <1> ;
50506                              <1> ;set_sample_rate:
50507                              <1> ;     ;movzx eax, word [audio_freq]
50508                              <1> ;     mov    ax, [audio_freq]
50509                              <1> ;     mov    edx, CODEC_PCM_FRONT_DACRATE_REG ; 2Ch ; PCM Front DAC Rate
50510                              <1> ;     ;call  codec_write
50511                              <1> ;     ;retn
50512                              <1> ;     jmp    codec_write
50513                              <1>
50514                              <1> ;cconfig_error:
50515                              <1> ;     retn
50516                              <1>
50517                              <1> vt8233_int_handler:
50518                              <1>       ; Interrupt Handler for VIA VT8237R Audio Controller
50519                              <1>       ; Note: called by 'dev_IRQ_service'
50520                              <1>       ; 14/10/2017
50521                              <1>       ; 09/10/2017, 10/10/2017, 12/10/2017
50522                              <1>       ; 13/06/2017
50523                              <1>       ; 21/04/2017 (TRDOS 386 kernel, 'audio.s')
50524                              <1>       ; 24/03/2017 - 'PLAYER.COM' ('player.asm')
50525                              <1>
50526                              <1>       ;push  eax ; * must be saved !
50527                              <1>       ;push  edx
50528                              <1>       ;push  ecx
50529                              <1>       ;push  ebx ; * must be saved !
50530                              <1>       ;push  esi
50531                              <1>       ;push  edi
50532                              <1>
50533                              <1>       ;cmp   byte [audio_busy], 1
50534                              <1>       ;jnb   short _ih0 ; 09/10/2017
50535                              <1>
50536                              <1>       ;mov   byte [audio_flag_eol], 0
50537                              <1>
50538 000116F7 66BA0000            <1>         mov     dx, VIADEV_PLAYBACK + VIA_REG_OFFSET_STATUS
50539 000116FB E808FFFFFF          <1>         call    ctrl_io_r8
50540                              <1>
50541 00011700 A880                <1>       test    al, VIA_REG_STAT_ACTIVE
50542 00011702 7417                <1>         jz      short _ih0 ; 09/10/2017
50543                              <1>
50544 00011704 2407                <1>         and     al, VIA_REG_STAT_EOL + VIA_REG_STAT_FLAG + VIA_REG_STAT_STOPPED
50545 00011706 A2[D9650100]        <1>       mov    [audio_flag_eol], al
50546 0001170B 740E                <1>         jz   short _ih0 ; 09/10/2017
50547                              <1>
50548                              <1>       ; 09/10/2017
50549                              <1>       ;mov   byte [audio_busy], 1
50550                              <1>
50551 0001170D 803D[D8650100]01    <1>       cmp    byte [audio_play_cmd], 1
50552 00011714 7315                <1>       jnb    short _ih1 ; 10/10/2017
50553                              <1>
50554 00011716 E860000000          <1>       call   channel_reset
50555                              <1> _ih0:
50556                              <1>       ; 09/10/2017
50557 0001171B A0[D9650100]        <1>         mov     al, [audio_flag_eol]   ;; ack ;;
50558 00011720 66BA0000            <1>         mov     dx, VIADEV_PLAYBACK + VIA_REG_OFFSET_STATUS
50559 00011724 E8D6FEFFFF          <1>         call    ctrl_io_w8
50560 00011729 EB4F                <1>       jmp    short _ih4
50561                              <1> _ih1:
50562                              <1> vt8233_tuneLoop:
50563 0001172B A0[D9650100]        <1>         mov     al, [audio_flag_eol]   ;; ack ;;
```

```
50564 00011730 66BA0000            <1>         mov     dx, VIADEV_PLAYBACK + VIA_REG_OFFSET_STATUS
50565 00011734 E8C6FEFFFF          <1>         call    ctrl_io_w8
50566                              <1>
50567                              <1>       ; 12/10/2017
50568 00011739 C605[CC650100]00    <1>       mov    byte [audio_flag], 0 ; Reset
50569                              <1>
50570                              <1>       ; 10/10/2017
50571                              <1>       ; 09/10/2017
50572                              <1>       ;test  byte [audio_flag_eol], VIA_REG_STAT_FLAG
50573                              <1>       ;jz    short _ih2 ; EOL
50574                              <1>
50575                              <1>       ; 14/10/2017
50576 00011740 F605[D9650100]02    <1>       test   byte [audio_flag_eol], VIA_REG_STAT_EOL
50577 00011747 7506                <1>       jnz    short _ih2 ; EOL
50578                              <1>                       ; (Half Buffer 2 has been completed
50579                              <1>                       ; and Half Buffer 1 will be played.)
50580                              <1>       ; FLAG
50581                              <1>       ; (Half Buffer 1 has been completed
50582                              <1>       ;  and Half Buffer 2 will be played.)
50583                              <1>
50584                              <1>       ; 14/10/2017
50585                              <1>       ;; (Continue to play.)
50586                              <1>       ;mov   al, VIA_REG_CTRL_INT
50587                              <1>             ;or    al, VIA_REG_CTRL_START
50588                              <1>             ;mov   dx, VIADEV_PLAYBACK + VIA_REG_OFFSET_CONTROL
50589                              <1>         ;call      ctrl_io_w8
50590                              <1>       ; 12/10/2017
50591                              <1>       ;mov   byte [audio_flag], 1
50592 00011749 FE05[CC650100]      <1>       inc    byte [audio_flag] ; = 1
50593                              <1> _ih2:
50594                              <1>       ; 10/10/2017
50595 0001174F 8B3D[C4650100]      <1>       mov    edi, [audio_dma_buff]
50596 00011755 8B0D[C8650100]      <1>       mov    ecx, [audio_dmabuff_size]
50597 0001175B D1E9                <1>       shr    ecx, 1 ; dma buff size / 2 = half buffer size
50598                              <1>
50599                              <1>       ; 12/10/2017
50600 0001175D 803D[CC650100]00    <1>       cmp    byte [audio_flag], 0
50601 00011764 7702                <1>       ja     short _ih3 ; Playing Half Buffer 2 (Current: FLAG)
50602                              <1>       ; Playing Half Buffer 1 (Current: EOL)
50603 00011766 01CF                <1>       add    edi, ecx
50604                              <1> _ih3:
50605                              <1>       ; Update half buffer 2 while playing half buffer 1 (FLAG)
50606                              <1>       ; Update half buffer 1 while playing half buffer 2 (EOL)
50607                              <1>
50608 00011768 8B35[BC650100]      <1>       mov    esi, [audio_p_buffer] ; phy addr of audio buff
50609 0001176E C1E902              <1>       shr    ecx, 2 ; half buff size / 4
50610 00011771 F3A5                <1>       rep    movsd
50611                              <1>       ; switch flag value ;
50612 00011773 8035[CC650100]01    <1>       xor    byte [audio_flag], 1 ; 10/10/2017
50613                              <1>       ; 12/10/2017
50614                              <1>       ; [audio_flag] = 0 : Playing dma half buffer 2 (just after FLAG)
50615                              <1>                          ; Next buffer (to update) is dma half buff 1
50616                              <1>       ;             = 1 : Playing dma half buffer 1 (just after EOL)
50617                              <1>                            ; Next buffer (to update) is dma half buff 2
50618                              <1> _ih4:
50619                              <1>       ; 28/05/2017
50620                              <1>       ;mov   byte [audio_busy], 0 ; 09/10/2017
50621                              <1>       ;
50622                              <1>       ;pop   edi
50623                              <1>       ;pop   esi
50624                              <1>       ;pop   ebx ; * must be restored !
50625                              <1>       ;pop   ecx
50626                              <1>       ;pop   edx
50627                              <1>       ;pop   eax ; * must be restored !
50628                              <1>
50629 0001177A C3                  <1>       retn
50630                              <1>
50631                              <1> channel_reset:
50632                              <1>       ; 24/06/2017
50633                              <1>       ; 29/05/2017
50634                              <1>       ; 23/03/2017
50635                              <1>       ; 14/11/2016 - Erdogan Tan
50636                              <1>       ; 12/11/2016 - Erdogan Tan (Ref: KolibriOS, vt823x.asm)
50637 0001177B BA01000000          <1>         mov  edx, VIA_REG_OFFSET_CONTROL
50638                              <1>         ;mov eax, VIA_REG_CTRL_PAUSE + VIA_REG_CTRL_TERMINATE + VIA_REG_CTRL_RESET
50639 00011780 B848000000          <1>         mov  eax, VIA_REG_CTRL_PAUSE + VIA_REG_CTRL_TERMINATE ; 24/06/2017
50640 00011785 E875FEFFFF          <1>       call    ctrl_io_w8
50641                              <1>
50642                              <1>         ;mov edx, VIA_REG_OFFSET_CONTROL
50643                              <1>         ;call   ctrl_io_r8
50644                              <1>
50645                              <1>       ; wait for 50 ms
50646 0001178A B9A0000000          <1>       mov    ecx, 160 ; (200*0.25 ms) ; 29/05/2017
50647                              <1> _ch_rst_wait:
50648 0001178F E833FEFFFF          <1>       call   delay1_4ms
50649 00011794 49                  <1>       dec    ecx
50650 00011795 75F8                <1>       jnz    short _ch_rst_wait
50651                              <1>
50652                              <1>         ; disable interrupts
50653 00011797 BA01000000          <1>         mov  edx, VIA_REG_OFFSET_CONTROL
50654 0001179C 31C0                <1>         xor     eax, eax
50655 0001179E E85CFEFFFF          <1>         call    ctrl_io_w8
50656                              <1>
50657                              <1>         ; clear interrupts
50658 000117A3 BA00000000          <1>         mov  edx, VIA_REG_OFFSET_STATUS
50659 000117A8 B803000000          <1>       mov    eax, 3
50660 000117AD E84DFEFFFF          <1>         call ctrl_io_w8
50661                              <1>
50662                              <1>       ;mov   edx, VIA_REG_OFFSET_CURR_PTR
50663                              <1>       ;xor   eax, eax
50664                              <1>       ;call  ctrl_io_w32
50665                              <1>
50666 000117B2 C3                  <1>         retn
```

```
                                <1>
                                <1> vt8233_stop: ; 22/04/2017
000117B3 C605[D8650100]00       <1>       mov   byte [audio_play_cmd], 0 ; stop !
                                <1> _tlp2:
                                <1>       ; 24/06/2017
                                <1>         ; finished with song, stop everything
                                <1>       ;mov   al, VIA_REG_CTRL_INT
                                <1>         ;or  al, VIA_REG_CTRL_TERMINATE
                                <1>       ;mov   dx, VIADEV_PLAYBACK + VIA_REG_OFFSET_CONTROL
                                <1>         ;call      ctrl_io_w8
                                <1>
                                <1>         ;call      channel_reset
                                <1>       ;retn
000117BA EBBF                   <1>       jmp   short channel_reset
                                <1>
                                <1> set_vt8233_bdl: ; Set VT8237R Buffer Descriptor List
                                <1>       ; 28/05/2017
                                <1>       ; 21/04/2017 (TRDOS 386 kernel, 'audio.s')
                                <1>       ; 24/03/2017 - 'PLAYER.COM' ('via_wav.asm' - 29/11/2016)
                                <1>
                                <1>       ; eax = dma buffer address = [audio_DMA_buff]
                                <1>       ; ecx = dma buffer buffer size = [audio_dmabuff_size]
                                <1>
000117BC D1E9                   <1>       shr   ecx, 1 ; dma half buffer size
000117BE 89CE                   <1>       mov   esi, ecx
                                <1>
000117C0 BF[E0650100]           <1>         mov     edi, audio_bdl_buff    ; get BDL address
000117C5 B910000000             <1>         mov     ecx, 32 / 2            ; make 32 entries in BDL
                                <1>
000117CA EB05                   <1>       jmp   short s_vt8233_bdl1
                                <1>
                                <1> s_vt8233_bdl0:
                                <1>       ; set buffer descriptor 0 to start of data file in memory
                                <1>
000117CC A1[C4650100]           <1>       mov   eax, [audio_dma_buff]     ; Physical address of DMA buffer
                                <1>
                                <1> s_vt8233_bdl1:
000117D1 AB                     <1>       stosd                     ; store dmabuffer1 address
                                <1>
000117D2 89C2                   <1>       mov   edx, eax
                                <1>
                                <1> ; VIA VT8235.PDF: (Page 110) (Erdogan Tan, 29/11/2016)
                                <1>       ;
                                <1>       ;     Audio SGD Table Format
                                <1>       ;     -------------------------------
                                <1>       ;     63   62    61-56    55-32  31-0
                                <1>       ;     --   --   --------  -----  ----
                                <1>       ;     EOL FLAG -reserved- Base   Base
                                <1>       ;                       Count  Address
                                <1>       ;                        [23:0] [31:0]
                                <1>       ;     EOL: End Of Link.
                                <1>       ;          1 indicates this block is the last of the link.
                                <1>       ;          If the channel “Interrupt on EOL” bit is set, then
                                <1>       ;          an interrupt is generated at the end of the transfer.
                                <1>       ;
                                <1>       ;     FLAG: Block Flag. If set, transfer pauses at the end of this
                                <1>       ;           block. If the channel “Interrupt on FLAG” bit is set,
                                <1>       ;           then an interrupt is generated at the end of this block.
                                <1>
000117D4 89F0                   <1>       mov   eax, esi ; DMA half buffer size
000117D6 01C2                   <1>       add   edx, eax
000117D8 0D00000040             <1>       or    eax, FLAG
                                <1>       ;or   eax, EOL
000117DD AB                     <1>       stosd
                                <1>
                                <1> ; 2nd buffer:
                                <1>
000117DE 89D0                   <1>         mov eax, edx ; Physical address of the 2nd half of DMA buffer
000117E0 AB                     <1>       stosd         ; store dmabuffer2 address
                                <1>
                                <1> ; set length to [audio_dmabuff_size]/2
                                <1> ; Set control (bits 31:16) to BUP, bits 15:0=number of samples
                                <1> ;
000117E1 89F0                   <1>       mov   eax, esi ; DMA half buffer size
000117E3 0D00000080             <1>       or    eax, EOL
                                <1>       ;or   eax, FLAG
000117E8 AB                     <1>       stosd
                                <1>
000117E9 E2E1                   <1>       loop    s_vt8233_bdl0
                                <1>
000117EB C3                     <1>       retn
                                <1>
                                <1> vt8233_start_play:
                                <1>       ; start to play audio data via VT8233 audio controller
                                <1>       ; 13/06/2017
                                <1>       ; 10/06/2017
                                <1>       ; 24/04/2017
                                <1>       ; 21/04/2017 (TRDOS 386 kernel, 'audio.s')
                                <1>       ; 24/03/2017 - 'PLAYER.COM' ('via_wav.asm' - 29/11/2016)
                                <1>       ; write buffer descriptor list address
                                <1>       ;
                                <1>
                                <1>       ; Extended Audio Status (2Ah)
000117EC B82A000000             <1>       mov   eax, CODEC_EXT_AUDIO_CTRL_REG ; 2Ah
000117F1 E82DFEFFFF             <1>       call  codec_read
000117F6 25FDFF0000             <1>       and     eax, 0FFFFh - 2          ; clear DRA (BIT1)
                                <1>       ;or     eax, 1                   ; set VRA (BIT0)
000117FB 83C805                 <1>       or    eax, 5       ; VRA (BIT0) & S/PDIF (BIT2) ; 14/11/2016
000117FE BA2A000000             <1>       mov   edx, CODEC_EXT_AUDIO_CTRL_REG
00011803 E847FEFFFF             <1>       call  codec_write
                                <1>       ;jc   short cconfig_error
                                <1>
                                <1> set_sample_rate:
```

```
50770                                  <1>       ;movzx eax, word [audio_freq]
50771 00011808 66A1[D6650100]          <1>       mov   ax, [audio_freq]
50772 0001180E BA2C000000              <1>       mov   edx, CODEC_PCM_FRONT_DACRATE_REG ; 2Ch ; PCM Front DAC Rate
50773 00011813 E837FEFFFF              <1>       call  codec_write
50774                                  <1>
50775 00011818 B8[E0650100]            <1>         mov  eax, audio_bdl_buff
50776                                  <1>
50777                                  <1>       ; 12/11/2016 - Erdogan Tan
50778                                  <1>       ; (Ref: KolibriOS, vt823x.asm, 'create_primary_buff')
50779 0001181D BA04000000              <1>       mov   edx, VIADEV_PLAYBACK + VIA_REG_OFFSET_TABLE_PTR
50780 00011822 E8EAFDFFFF              <1>         call ctrl_io_w32
50781                                  <1>
50782                                  <1>       ;call  codec_check_ready
50783                                  <1>
50784 00011827 66BA0200                <1>       mov   dx, VIADEV_PLAYBACK + VIA_REG_OFS_PLAYBACK_VOLUME_L
50785                                  <1>         ;mov eax, 2 ; 31
50786 0001182B B01F                    <1>       mov   al, 31
50787 0001182D 2A05[DA650100]          <1>         sub  al, [audio_master_volume_l]
50788 00011833 E8C7FDFFFF              <1>       call  ctrl_io_w8
50789                                  <1>
50790                                  <1>       ;call  codec_check_ready
50791                                  <1>
50792 00011838 66BA0300                <1>         mov     dx, VIADEV_PLAYBACK + VIA_REG_OFS_PLAYBACK_VOLUME_R
50793                                  <1>         ;mov ax, 2  ; 31
50794 0001183C B01F                    <1>       mov   al, 31
50795 0001183E 2A05[DB650100]          <1>         sub  al, [audio_master_volume_r]
50796 00011844 E8B6FDFFFF              <1>       call    ctrl_io_w8
50797                                  <1>
50798                                  <1>       ;call  codec_check_ready
50799                                  <1> ;
50800                                  <1> ;
50801                                  <1> ; All set.  Let's play some music.
50802                                  <1> ;
50803                                  <1> ;
50804                                  <1>             ;mov    dx, VIADEV_PLAYBACK + VIA_REG_OFFSET_STOP_IDX
50805                                  <1>         ;mov    ax, VIA8233_REG_TYPE_16BIT or VIA8233_REG_TYPE_STEREO or 0xfffff or 0xff000000
50806                                  <1>         ;call   ctrl_io_w32
50807                                  <1>
50808                                  <1>       ;call  codec_check_ready
50809                                  <1>
50810                                  <1>       ; 08/12/2016
50811                                  <1>       ; 07/10/2016
50812                                  <1>         ;mov    al, 1
50813 00011849 B01F                    <1>         mov al, 31
50814 0001184B E815000000              <1>       call    set_VT8233_LastValidIndex
50815                                  <1>
50816 00011850 C605[D8650100]01        <1>       mov   byte [audio_play_cmd], 1 ; play command (do not stop) !
50817                                  <1>
50818                                  <1> vt8233_play: ; continue to play
50819                                  <1>       ; 22/04/2017
50820 00011857 B023                    <1>         mov al, VIA_REG_CTRL_INT
50821 00011859 0C80                    <1>             or    al, VIA_REG_CTRL_START
50822                                  <1>         ;mov al, VIA_REG_CTRL_AUTOSTART + VIA_REG_CTRL_START
50823                                  <1>       ;mov  al, VIA_REG_CTRL_AUTOSTART + VIA_REG_CTRL_START + VIA_REG_CTRL_INT_FLAG
50824 0001185B 66BA0100                <1>       mov     dx, VIADEV_PLAYBACK + VIA_REG_OFFSET_CONTROL
50825 0001185F E89BFDFFFF              <1>         call    ctrl_io_w8
50826                                  <1>       ;call  codec_check_ready
50827                                  <1>       ;retn
50828                                  <1>       ;jmp  codec_check_ready
50829 00011864 C3                      <1>       retn
50830                                  <1>
50831                                  <1> ;input AL = index # to stop on
50832                                  <1> set_VT8233_LastValidIndex:
50833                                  <1>       ; 10/06/2017
50834                                  <1>       ; 21/04/2017 (TRDOS 386 kernel, 'audio.s')
50835                                  <1>       ; 24/03/2017 - 'PLAYER.COM' ('via_wav.asm' - 29/11/2016)
50836                                  <1>       ; 19/11/2016
50837                                  <1>       ; 14/11/2016 - Erdogan Tan (Ref: VIA VT8235.PDF, Page 110)
50838                                  <1>       ; 12/11/2016 - Erdogan Tan
50839                                  <1>       ; (Ref: KolibriOS, vt823x.asm, 'create_primary_buff')
50840                                  <1>       ;push  edx
50841 00011865 6650                    <1>       push  ax
50842                                  <1>       ;push  ecx
50843 00011867 0FB705[D6650100]        <1>       movzx  eax, word [audio_freq] ; Hertz
50844 0001186E BA00001000              <1>       mov   edx, 100000h ; 2^20 = 1048576
50845 00011873 F7E2                    <1>       mul   edx
50846 00011875 B980BB0000              <1>       mov   ecx, 48000
50847 0001187A F7F1                    <1>       div   ecx
50848                                  <1>       ;and  eax, 0FFFFFh
50849                                  <1>       ;pop  ecx
50850 0001187C 665A                    <1>       pop   dx
50851 0001187E C1E218                  <1>       shl   edx, 24  ; STOP Index Setting: Bit 24 to 31
50852 00011881 09D0                    <1>       or    eax, edx
50853                                  <1>       ; 19/11/2016
50854 00011883 803D[D4650100]10        <1>       cmp   byte [audio_bps], 16
50855 0001188A 7505                    <1>       jne   short sLVI_1
50856 0001188C 0D00002000              <1>       or    eax, VIA8233_REG_TYPE_16BIT
50857                                  <1> sLVI_1:
50858 00011891 803D[D5650100]02        <1>       cmp   byte [audio_stmo], 2
50859 00011898 7505                    <1>       jne   short sLVI_2
50860 0001189A 0D00001000              <1>       or    eax, VIA8233_REG_TYPE_STEREO
50861                                  <1> sLVI_2:
50862 0001189F BA08000000              <1>       mov     edx, VIADEV_PLAYBACK + VIA_REG_OFFSET_STOP_IDX
50863 000118A4 E868FDFFFF              <1>         call    ctrl_io_w32
50864                                  <1>       ;call  codec_check_ready
50865                                  <1>       ;pop  edx
50866 000118A9 C3                      <1>       retn
50867                                  <1>
50868                                  <1> vt8233_pause: ; pause
50869                                  <1>       ; 10/06/2017
50870                                  <1>       ; 22/04/2017
50871 000118AA B023                    <1>       mov     al, VIA_REG_CTRL_INT
50872 000118AC 0C08                    <1>         or      al, VIA_REG_CTRL_PAUSE
```

```
50873 000118AE 66BA0100            <1>       mov     dx, VIADEV_PLAYBACK + VIA_REG_OFFSET_CONTROL
50874 000118B2 E848FDFFFF          <1>         call    ctrl_io_w8
50875                              <1>       ;call  codec_check_ready
50876                              <1>       ;retn
50877                              <1>       ;jmp   codec_check_ready
50878 000118B7 C3                  <1>       retn
50879                              <1>
50880                              <1> vt8233_reset:
50881                              <1>       ; 22/04/2017
50882                              <1>       ; reset VT8237R (vt8233) Audio Controller
50883                              <1>       ;cmp   byte [audio_play_cmd], 1
50884                              <1>       ;jna   short vt8233_rst_0
50885 000118B8 C605[D8650100]00    <1>       mov    byte [audio_play_cmd], 0 ; stop !
50886                              <1> vt8233_rst_0:
50887 000118BF E8A6FCFFFF          <1>       call   reset_codec
50888 000118C4 720A                <1>       jc     short vt8233_rst_1 ; codec error !
50889                              <1>       ; eax = 1
50890 000118C6 E818FDFFFF          <1>       call   codec_io_w16 ; w32
50891 000118CB E8ABFEFFFF          <1>       call   channel_reset
50892                              <1> vt8233_rst_1:
50893 000118D0 C3                  <1>       retn
50894                              <1>
50895                              <1> vt8233_volume:
50896                              <1>       ; set VT8237R (vt8233) sound volume level
50897                              <1>       ; 24/04/2017
50898                              <1>       ; 22/04/2017
50899                              <1>       ; bl = component (0 = master/playback/lineout volume)
50900                              <1>       ; cl = left channel volume level (0 to 31)
50901                              <1>       ; ch = right channel volume level (0 to 31)
50902                              <1>
50903 000118D1 08DB                <1>       or     bl, bl
50904 000118D3 7520                <1>       jnz    short vt8233_vol_1 ; temporary !
50905 000118D5 66B81F1F            <1>       mov    ax, 1F1Fh ; 31,31
50906 000118D9 38C1                <1>       cmp    cl, al
50907 000118DB 7718                <1>       ja     short vt8233_vol_1 ; temporary !
50908 000118DD 38E5                <1>       cmp    ch, ah
50909 000118DF 7714                <1>       ja     short vt8233_vol_1 ; temporary !
50910 000118E1 66890D[DA650100]    <1>       mov    [audio_master_volume], cx
50911 000118E8 6629C8              <1>       sub    ax, cx
50912 000118EB BA02000000          <1>       mov    edx, CODEC_MASTER_VOL_REG ; 02h ; Line Out
50913 000118F0 E85AFDFFFF          <1>       call   codec_write
50914                              <1> vt8233_vol_1:
50915 000118F5 C3                  <1>       retn
50916                              <1>
50917                              <1> ; CODE for SOUND BLASTER 16
50918                              <1>
50919                              <1> DetectSB:
50920                              <1>       ; 24/04/2017
50921                              <1>       ;pushad
50922                              <1> ScanPort:
50923 000118F6 66BB1002            <1>       mov     bx, 210h    ; start scanning ports
50924                              <1>                           ; 210h, 220h, .. 260h
50925                              <1> ResetDSP:
50926 000118FA 6689DA              <1>       mov     dx, bx            ; try to reset the DSP.
50927 000118FD 6683C206            <1>       add     dx, 06h
50928 00011901 B001                <1>       mov    al, 1
50929 00011903 EE                  <1>       out    dx, al
50930                              <1>
50931 00011904 EC                  <1>       in     al, dx
50932 00011905 EC                  <1>       in     al, dx
50933 00011906 EC                  <1>       in     al, dx
50934 00011907 EC                  <1>       in     al, dx
50935                              <1>
50936 00011908 30C0                <1>       xor     al, al
50937 0001190A EE                  <1>       out    dx, al
50938                              <1>
50939 0001190B 6683C208            <1>       add     dx, 08h
50940 0001190F 66B96400            <1>       mov    cx, 100
50941                              <1> WaitID:
50942 00011913 EC                  <1>       in     al, dx
50943 00011914 08C0                <1>       or      al, al
50944 00011916 7804                <1>       js      short GetID
50945 00011918 E2F9                <1>       loop    WaitID
50946 0001191A EB0F                <1>       jmp     short NextPort
50947                              <1> GetID:
50948 0001191C 6683EA04            <1>       sub     dx, 04h
50949 00011920 EC                  <1>       in     al, dx
50950 00011921 3CAA                <1>       cmp     al, 0AAh
50951 00011923 7413                <1>       je      short Found
50952 00011925 6683C204            <1>       add     dx, 04h
50953 00011929 E2E8                <1>       loop    WaitID
50954                              <1> NextPort:
50955 0001192B 6683C310            <1>       add     bx, 10h           ; if not response,
50956 0001192F 6681FB6002          <1>       cmp     bx, 260h    ; try the next port.
50957 00011934 76C4                <1>       jbe     short ResetDSP
50958 00011936 F9                  <1>       stc
50959 00011937 C3                  <1>       retn
50960                              <1> Found:
50961 00011938 66891D[AA650100]    <1>       mov     [audio_io_base], bx      ; SB Port Address Found!
50962                              <1> ScanIRQ:
50963                              <1> SetIrqs:
50964 0001193F 28C0                <1>       sub    al, al ; 0
50965 00011941 A2[A2650100]        <1>       mov    [IRQnum], al ; reset
50966 00011946 A2[A7650100]        <1>       mov    [audio_intr], al ; reset
50967                              <1>
50968                              <1>       ; ah > 0 -> set IRQ vector
50969                              <1>       ; al = IRQ number
50970                              <1>       ;mov   ax, 103h ; IRQ 3
50971                              <1>       ;call  set_hardware_int_vector
50972                              <1>       ;mov   ax, 104h ; IRQ 4
50973                              <1>       ;call  set_hardware_int_vector
50974 0001194B 66B80501            <1>       mov    ax, 105h ; IRQ 5
50975 0001194F E8F0DDFFFF          <1>       call   set_hardware_int_vector
```

```
50976 00011954 66B80701            <1>       mov    ax, 107h ; IRQ 7
50977 00011958 E8E7DDFFFF          <1>       call   set_hardware_int_vector
50978                              <1>
50979 0001195D 668B15[AA650100]    <1>       mov     dx, [audio_io_base] ; tells to the SB to
50980 00011964 6683C20C            <1>       add     dx, 0Ch               ; generate a IRQ!
50981                              <1> WaitSb:
50982 00011968 EC                  <1>       in     al, dx
50983 00011969 08C0                <1>       or      al, al
50984 0001196B 78FB                <1>       js      short WaitSb
50985 0001196D B0F2                <1>       mov     al, 0F2h
50986 0001196F EE                  <1>       out    dx, al
50987                              <1>
50988 00011970 31C9                <1>       xor     ecx, ecx    ; wait until IRQ level
50989                              <1> WaitIRQ:
50990 00011972 A0[A2650100]        <1>       mov    al, [IRQnum]
50991 00011977 3C00                <1>       cmp     al, 0 ; is changed or timeout.
50992 00011979 7706                <1>       ja     short IrqOk
50993 0001197B 6649                <1>       dec    cx
50994 0001197D 75F3                <1>       jnz    short WaitIRQ
50995 0001197F EB15                <1>       jmp    short RestoreIrqs
50996                              <1> IrqOk:
50997 00011981 A2[A7650100]        <1>       mov    [audio_intr], al ; set
50998 00011986 668B15[AA650100]    <1>       mov     dx, [audio_io_base]
50999 0001198D 6683C20E            <1>       add     dx, 0Eh
51000 00011991 EC                  <1>       in     al, dx ; SB acknowledge.
51001 00011992 B020                <1>       mov    al, 20h
51002 00011994 E620                <1>       out    20h, al      ; Hardware acknowledge.
51003                              <1>
51004                              <1> RestoreIrqs:
51005                              <1>       ; ah = 0 -> reset IRQ vector
51006                              <1>       ; al = IRQ number
51007                              <1>       ;mov   ax, 3 ; IRQ 3
51008                              <1>       ;call  set_hardware_int_vector
51009                              <1>       ;mov   ax, 4 ; IRQ 4
51010                              <1>       ;call  set_hardware_int_vector
51011 00011996 66B80500            <1>       mov    ax, 5 ; IRQ 5
51012 0001199A E8A5DDFFFF          <1>       call   set_hardware_int_vector
51013 0001199F 66B80700            <1>       mov    ax, 7 ; IRQ 7
51014 000119A3 E89CDDFFFF          <1>       call   set_hardware_int_vector
51015                              <1>
51016 000119A8 31D2                <1>       xor    edx, edx
51017 000119AA 8915[AC650100]      <1>       mov    [audio_dev_id], edx ; 0
51018 000119B0 8915[B0650100]      <1>       mov    [audio_vendor], edx ; 0
51019 000119B6 8915[B4650100]      <1>       mov    [audio_stats_cmd], edx ; 0
51020                              <1>
51021                              <1>       ;popad
51022                              <1>
51023 000119BC 803D[A7650100]01    <1>       cmp     byte [audio_intr], 1 ; IRQ level was changed?
51024                              <1>
51025 000119C3 C3                  <1>       retn
51026                              <1>
51027                              <1> %macro      SbOut  1
51028                              <1> %%Wait:
51029                              <1>       in     al, dx
51030                              <1>       or     al, al
51031                              <1>       js     short %%Wait
51032                              <1>       mov    al, %1
51033                              <1>       out    dx, al
51034                              <1> %endmacro
51035                              <1>
51036                              <1> SbInit_play:
51037                              <1>       ; 22/10/2017
51038                              <1>       ; 20/10/2017
51039                              <1>       ; 06/10/2017
51040                              <1>       ; 13/07/2017, 09/08/2017
51041                              <1>       ; 24/04/2017, 15/05/2017, 24/06/2017
51042                              <1>       ;pushad
51043                              <1> SetBuffer:
51044                              <1>       ;mov   byte [DmaFlag], 0
51045                              <1>
51046 000119C4 8B1D[C4650100]      <1>       mov    ebx, [audio_dma_buff] ; physical addr of DMA buff
51047 000119CA 89DF                <1>       mov    edi, ebx
51048 000119CC 8B0D[C8650100]      <1>       mov     ecx, [audio_dmabuff_size]
51049                              <1>
51050 000119D2 803D[D4650100]10    <1>       cmp    byte [audio_bps], 16
51051 000119D9 7531                <1>       jne    short sbInit_0 ; set 8 bit DMA buffer
51052                              <1>
51053                              <1>       ; 09/08/2017
51054                              <1>       ; convert byte count to word count
51055 000119DB D1E9                <1>       shr    ecx, 1
51056 000119DD 49                  <1>       dec    ecx ; word count - 1
51057                              <1>       ; convert byte offset to word offset
51058 000119DE D1EB                <1>       shr    ebx, 1
51059                              <1>
51060                              <1>       ; 16 bit DMA buffer setting (DMA channel 5)
51061 000119E0 B005                <1>       mov     al, 05h  ; set mask bit for channel 5  (4+1)
51062 000119E2 E6D4                <1>       out    0D4h, al
51063                              <1>
51064 000119E4 30C0                <1>       xor     al, al   ; stops all DMA processes on selected channel
51065 000119E6 E6D8                <1>       out    0D8h, al ; clear selected channel register
51066                              <1>
51067 000119E8 88D8                <1>       mov     al, bl       ; byte 0 of DMA buffer offset in words (physical)
51068 000119EA E6C4                <1>       out    0C4h, al ; DMA channel 5 port number
51069                              <1>
51070 000119EC 88F8                <1>       mov     al, bh   ; byte 1 of DMA buffer offset in words (physical)
51071 000119EE E6C4                <1>       out    0C4h, al
51072                              <1>
51073                              <1>       ; 09/08/2017
51074 000119F0 C1EB0F              <1>       shr    ebx, 15       ; complete 16 bit shift
51075 000119F3 80E3FE              <1>       and    bl, 0FEh ; clear bit 0 (not necessary, it will be ignored)
51076                              <1>
51077 000119F6 88D8                <1>       mov     al, bl   ; byte 2 of DMA buffer address (physical)
51078 000119F8 E68B                <1>       out    8Bh, al  ; page register port addr for channel 5 ; 13/07/2017
```

```
51079                                  <1>
51080 000119FA 88C8                    <1>       mov     al, cl   ; low byte of DMA count - 1
51081 000119FC E6C6                    <1>       out    0C6h, al ; count register port addr for channel 1
51082                                  <1>
51083 000119FE 88E8                    <1>       mov     al, ch   ; high byte of DMA count - 1
51084 00011A00 E6C6                    <1>       out    0C6h, al
51085                                  <1>
51086                                  <1>       ; channel 5, read, autoinitialized, single mode
51087                                  <1>       ;mov   al, 49h
51088 00011A02 B059                    <1>       mov    al, 59h  ; 06/10/2017
51089 00011A04 E6D6                    <1>       out    0D6h, al ; DMA mode register port address
51090                                  <1>
51091 00011A06 B001                    <1>       mov     al, 01h  ; clear mask bit for channel 1
51092 00011A08 E6D4                    <1>       out    0D4h, al ; DMA mask register port address
51093                                  <1>
51094 00011A0A EB28                    <1>       jmp    short ClearBuffer
51095                                  <1>
51096                                  <1> sbInit_0:
51097 00011A0C 49                      <1>       dec     ecx ; 09/08/2017
51098                                  <1>
51099                                  <1>       ; 8 bit DMA buffer setting (DMA channel 1)
51100 00011A0D B005                    <1>       mov     al, 05h ; set mask bit for channel 1  (4+1)
51101 00011A0F E60A                    <1>       out    0Ah, al ; DMA mask register
51102                                  <1>
51103 00011A11 30C0                    <1>       xor     al, al  ; stops all DMA processes on selected channel
51104 00011A13 E60C                    <1>       out    0Ch, al ; clear selected channel register
51105                                  <1>
51106 00011A15 88D8                    <1>       mov     al, bl      ; byte 0 of DMA buffer address (physical)
51107 00011A17 E602                    <1>       out    02h, al ; DMA channel 1 port number
51108                                  <1>
51109 00011A19 88F8                    <1>       mov     al, bh  ; byte 1 of DMA buffer address (physical)
51110 00011A1B E602                    <1>       out    02h, al
51111                                  <1>
51112 00011A1D C1EB10                  <1>       shr    ebx, 16
51113                                  <1>
51114 00011A20 88D8                    <1>       mov     al, bl  ; byte 2 of DMA buffer address (physical)
51115 00011A22 E683                    <1>       out    83h, al ; page register port addr for channel 1
51116                                  <1>
51117 00011A24 88C8                    <1>       mov     al, cl  ; low byte of DMA count - 1
51118 00011A26 E603                    <1>       out    03h, al ; count register port addr for channel 1
51119                                  <1>
51120 00011A28 88E8                    <1>       mov     al, ch  ; high byte of DMA count - 1
51121 00011A2A E603                    <1>       out    03h, al
51122                                  <1>
51123                                  <1>       ; channel 1, read, autoinitialized, single mode
51124                                  <1>       ;mov   al, 49h
51125 00011A2C B059                    <1>       mov    al, 59h ; 06/10/2017
51126 00011A2E E60B                    <1>       out    0Bh, al ; DMA mode register port address
51127                                  <1>
51128 00011A30 B001                    <1>       mov     al, 01h ; clear mask bit for channel 1
51129 00011A32 E60A                    <1>       out    0Ah, al ; DMA mask register port address
51130                                  <1>
51131                                  <1> ClearBuffer:
51132                                  <1>       ;;mov  edi, [audio_dma_buff]
51133                                  <1>       ;;mov  ecx, [audio_dmabuff_size]
51134                                  <1>       ;inc   ecx
51135                                  <1>       ;mov     al, 80h
51136                                  <1>       ;;cld
51137                                  <1>       ;rep     stosb
51138                                  <1> SetIrq:
51139                                  <1>       ;mov   ebx, SbIrqhandler
51140                                  <1>       ;mov   al, [audio_intr] ; IRQ number
51141                                  <1>       ;call  set_dev_IRQ_service
51142                                  <1>       ;; SETUP (audio) INTERRUPT CALLBACK SERVICE
51143                                  <1>       ;mov   bl, [audio_intr] ; IRQ number
51144                                  <1>       ;mov   bh, [audio_cb_mode]
51145                                  <1>       ;inc   bh  ; 1 = Signal Response Byte method (fixed value)
51146                                  <1>       ;          ; 2 = Callback service method
51147                                  <1>       ;          ; 3 = Auto Increment S.R.B. method
51148                                  <1>       ;mov   cl, [audio_srb]
51149                                  <1>       ;mov   edx, [audio_cb_addr]
51150                                  <1>       ;mov   al, [audio_user]
51151                                  <1>       ;call  set_irq_callback_service
51152                                  <1> ResetDsp:
51153 00011A34 668B15[AA650100]        <1>       mov     dx, [audio_io_base]
51154 00011A3B 6683C206                <1>       add     dx, 06h
51155 00011A3F B001                    <1>       mov     al, 1
51156 00011A41 EE                      <1>       out    dx, al
51157                                  <1>
51158 00011A42 EC                      <1>       in     al, dx
51159 00011A43 EC                      <1>       in     al, dx
51160 00011A44 EC                      <1>       in     al, dx
51161 00011A45 EC                      <1>       in     al, dx
51162                                  <1>
51163 00011A46 30C0                    <1>       xor     al, al
51164 00011A48 EE                      <1>       out    dx, al
51165                                  <1>
51166 00011A49 66B96400                <1>       mov     cx, 100
51167 00011A4D 28E4                    <1>       sub    ah, ah ; 0
51168                                  <1> WaitId:
51169 00011A4F 668B15[AA650100]        <1>       mov     dx, [audio_io_base]
51170 00011A56 6683C20E                <1>       add     dx, 0Eh
51171 00011A5A EC                      <1>       in     al, dx
51172 00011A5B 08C0                    <1>       or      al, al
51173 00011A5D 7807                    <1>       js      short sb_GetId
51174 00011A5F E2EE                    <1>       loop    WaitId
51175 00011A61 E9B4000000              <1>       jmp     sb_Exit
51176                                  <1> sb_GetId:
51177 00011A66 668B15[AA650100]        <1>       mov     dx, [audio_io_base]
51178 00011A6D 6683C20A                <1>       add     dx, 0Ah
51179 00011A71 EC                      <1>       in     al, dx
51180 00011A72 3CAA                    <1>       cmp     al, 0AAh
51181 00011A74 7407                    <1>       je      short SbOk
```

```
51182 00011A76 E2D7                <1>       loop    WaitId
51183 00011A78 E99D000000          <1>       jmp    sb_Exit
51184                              <1> SbOk:
51185 00011A7D 668B15[AA650100]    <1>       mov     dx, [audio_io_base]
51186 00011A84 6683C20C            <1>       add     dx, 0Ch
51187                              <1>       SbOut   0D1h ; Turn on speaker
51188                              <2> %%Wait:
51189 00011A88 EC                  <2>  in al, dx
51190 00011A89 08C0                <2>  or al, al
51191 00011A8B 78FB                <2>  js short %%Wait
51192 00011A8D B0D1                <2>  mov al, %1
51193 00011A8F EE                  <2>  out dx, al
51194                              <1>       SbOut   41h ; 8 bit or 16 bit transfer
51195                              <2> %%Wait:
51196 00011A90 EC                  <2>  in al, dx
51197 00011A91 08C0                <2>  or al, al
51198 00011A93 78FB                <2>  js short %%Wait
51199 00011A95 B041                <2>  mov al, %1
51200 00011A97 EE                  <2>  out dx, al
51201 00011A98 668B1D[D6650100]    <1>       mov    bx, [audio_freq] ; sampling rate (Hz)
51202                              <1>       SbOut  bh ; sampling rate high byte
51203                              <2> %%Wait:
51204 00011A9F EC                  <2>  in al, dx
51205 00011AA0 08C0                <2>  or al, al
51206 00011AA2 78FB                <2>  js short %%Wait
51207 00011AA4 88F8                <2>  mov al, %1
51208 00011AA6 EE                  <2>  out dx, al
51209                              <1>       SbOut  bl ; sampling rate low byte
51210                              <2> %%Wait:
51211 00011AA7 EC                  <2>  in al, dx
51212 00011AA8 08C0                <2>  or al, al
51213 00011AAA 78FB                <2>  js short %%Wait
51214 00011AAC 88D8                <2>  mov al, %1
51215 00011AAE EE                  <2>  out dx, al
51216                              <1>
51217                              <1>       ; 22/05/2017
51218 00011AAF E8C0000000          <1>       call   sb16_volume_initial ; 15/05/2017
51219                              <1>       ; 20/05/2017
51220                              <1>       ;call  sb16_volume
51221                              <1>
51222                              <1> StartDma:
51223                              <1>       ; autoinitialized mode
51224 00011AB4 803D[D4650100]10    <1>       cmp    byte [audio_bps], 16 ; 16 bit samples
51225 00011ABB 7411                <1>       je     short sb_play_1
51226                              <1>       ; 8 bit samples
51227 00011ABD 66BBC600            <1>       mov    bx, 0C6h ; 8 bit output (0C6h)
51228 00011AC1 803D[D5650100]02    <1>       cmp    byte [audio_stmo], 2 ; 1 = mono, 2 = stereo
51229 00011AC8 7214                <1>       jb     short sb_play_2
51230 00011ACA B720                <1>       mov    bh, 20h      ; 8 bit stereo (20h)
51231 00011ACC EB10                <1>       jmp    short sb_play_2
51232                              <1> sb_play_1:
51233                              <1>       ; 16 bit samples
51234 00011ACE 66BBB610            <1>       mov    bx, 10B6h ; 16 bit output (0B6h)
51235 00011AD2 803D[D5650100]02    <1>       cmp    byte [audio_stmo], 2 ; 1 = mono, 2 = stereo
51236 00011AD9 7203                <1>       jb     short sb_play_2
51237 00011ADB 80C720              <1>       add    bh, 20h      ; 16 bit stereo (30h)
51238                              <1> sb_play_2:
51239                              <1>       ; PCM output (8/16 bit mono autoinitialized transfer)
51240                              <1>       SbOut   bl ; bCommand
51241                              <2> %%Wait:
51242 00011ADE EC                  <2>  in al, dx
51243 00011ADF 08C0                <2>  or al, al
51244 00011AE1 78FB                <2>  js short %%Wait
51245 00011AE3 88D8                <2>  mov al, %1
51246 00011AE5 EE                  <2>  out dx, al
51247                              <1>       SbOut  bh ; bMode
51248                              <2> %%Wait:
51249 00011AE6 EC                  <2>  in al, dx
51250 00011AE7 08C0                <2>  or al, al
51251 00011AE9 78FB                <2>  js short %%Wait
51252 00011AEB 88F8                <2>  mov al, %1
51253 00011AED EE                  <2>  out dx, al
51254 00011AEE 8B1D[C8650100]      <1>       mov    ebx, [audio_dmabuff_size]  ; 15/05/2017
51255 00011AF4 D1EB                <1>       shr    ebx, 1 ; half buffer size
51256                              <1>       ; 20/10/2017
51257 00011AF6 803D[D4650100]10    <1>       cmp    byte [audio_bps], 16 ; 16 bit DMA
51258 00011AFD 7502                <1>       jne    short sb_play_3
51259 00011AFF D1EB                <1>       shr    ebx, 1 ; byte count to word count
51260                              <1> sb_play_3:
51261 00011B01 664B                <1>       dec    bx  ; wBlkSize is one less than the actual size
51262                              <1>       SbOut   bl
51263                              <2> %%Wait:
51264 00011B03 EC                  <2>  in al, dx
51265 00011B04 08C0                <2>  or al, al
51266 00011B06 78FB                <2>  js short %%Wait
51267 00011B08 88D8                <2>  mov al, %1
51268 00011B0A EE                  <2>  out dx, al
51269                              <1>       SbOut   bh
51270                              <2> %%Wait:
51271 00011B0B EC                  <2>  in al, dx
51272 00011B0C 08C0                <2>  or al, al
51273 00011B0E 78FB                <2>  js short %%Wait
51274 00011B10 88F8                <2>  mov al, %1
51275 00011B12 EE                  <2>  out dx, al
51276                              <1>
51277 00011B13 C605[D8650100]01    <1>       mov    byte [audio_play_cmd], 1 ; playing !
51278                              <1>
51279                              <1>       ;; Set Voice and master volumes
51280                              <1>       ;mov   dx, [audio_io_base]
51281                              <1>       ;add   dl, 4 ; Mixer chip Register Address Port
51282                              <1>       ;SbOut 30h   ; select Master Volume Register (L)
51283                              <1>       ;inc   dl    ; Mixer chip Register Data Port
51284                              <1>       ;SbOut 0F8h  ; Max. volume value is 31 (31*8)
```

```
51285                                  <1>       ;dec   dl
51286                                  <1>       ;SbOut 31h   ; select Master Volume Register (R)
51287                                  <1>       ;inc   dl
51288                                  <1>       ;SbOut 0F8h  ; Max. volume value is 31 (31*8)
51289                                  <1>       ;dec   dl
51290                                  <1>       ;SbOut 32h   ; select Voice Volume Register (L)
51291                                  <1>       ;inc   dl
51292                                  <1>       ;SbOut 0F8h  ; Max. volume value is 31 (31*8)
51293                                  <1>       ;dec   dl
51294                                  <1>       ;SbOut 33h   ; select Voice Volume Register (R)
51295                                  <1>       ;inc   dl
51296                                  <1>       ;SbOut 0F8h  ; Max. volume value is 31 (31*8)
51297                                  <1>       ;;
51298                                  <1>       ;dec   dl
51299                                  <1>       ;SbOut 44h   ; select Treble Register (L)
51300                                  <1>       ;inc   dl
51301                                  <1>       ;SbOut 0F0h  ; Max. Treble value is 15 (15*16)
51302                                  <1>       ;dec   dl
51303                                  <1>       ;SbOut 45h   ; select Treble Register (R)
51304                                  <1>       ;inc   dl
51305                                  <1>       ;SbOut 0F0h  ; Max. Treble value is 15 (15*16)
51306                                  <1>       ;dec   dl
51307                                  <1>       ;SbOut 46h   ; select Bass Register (L)
51308                                  <1>       ;inc   dl
51309                                  <1>       ;SbOut 0F0h  ; Max. Bass value is 15 (15*16)
51310                                  <1>       ;dec   dl
51311                                  <1>       ;SbOut 47h   ; select Bass Register (R)
51312                                  <1>       ;inc   dl
51313                                  <1>       ;SbOut 0F0h  ; Max. Bass value is 15 (15*16)
51314                                  <1>
51315                                  <1> sb_Exit:
51316                                  <1>       ;popad
51317 00011B1A C3                      <1>       retn
51318                                  <1>
51319                                  <1> sb16_int_handler:
51320                                  <1>       ; Interrupt Handler for Sound Blaster 16 Audio Card
51321                                  <1>       ; Note: called by 'dev_IRQ_service'
51322                                  <1>       ; 20/10/2017
51323                                  <1>       ; 12/10/2017
51324                                  <1>       ; 10/10/2017
51325                                  <1>       ; 12/05/2017, 09/10/2017
51326                                  <1>       ; 24/04/2017 (TRDOS 386 kernel, 'audio.s')
51327                                  <1>       ; 10/03/2017 - 'PLAYWAV.PRG' ('playwav.s')
51328                                  <1>
51329                                  <1>       ;push  eax ; * must be saved !
51330                                  <1>       ;push  ebx ; * must be saved !
51331                                  <1>       ;push  ecx
51332                                  <1>       ;push  edx
51333                                  <1>       ;push  esi
51334                                  <1>       ;push  edi
51335                                  <1>
51336 00011B1B 668B15[AA650100]        <1>       mov     dx, [audio_io_base]
51337                                  <1>       ; 20/10/2017
51338 00011B22 80C20F                  <1>       add     dl, 0Fh ; 2xFh (DSP 16 bit intr ack)
51339 00011B25 803D[D4650100]10        <1>       cmp    byte [audio_bps], 16
51340 00011B2C 7402                    <1>       je     short sb_irq_16bit_ack
51341                                  <1> sb_irq_8bit_ack:
51342 00011B2E FECA                    <1>       dec    dl  ; 2xEh (DSP 8 bit intr ack)
51343                                  <1> sb_irq_16bit_ack:
51344 00011B30 EC                      <1>       in     al, dx
51345                                  <1>
51346                                  <1>       ;cmp   byte [audio_busy], 0
51347                                  <1>       ;ja    short sb_irq_h3
51348                                  <1>
51349                                  <1>       ;mov   byte [audio_busy], 1
51350                                  <1>
51351 00011B31 803D[D8650100]01        <1>       cmp    byte [audio_play_cmd], 1
51352 00011B38 7307                    <1>       jnb    short sb_irq_h1
51353                                  <1> sb_irq_h0:
51354 00011B3A E8A9000000              <1>       call   sb16_stop
51355 00011B3F EB2B                    <1>       jmp    short sb_irq_h3
51356                                  <1> sb_irq_h1:
51357                                  <1>       ;call  sb16_tuneloop
51358                                  <1>       ; 09/10/2017
51359                                  <1> sb16_tuneloop:
51360 00011B41 8B3D[C4650100]          <1>       mov    edi, [audio_dma_buff]
51361 00011B47 8B0D[C8650100]          <1>       mov    ecx, [audio_dmabuff_size]
51362 00011B4D D1E9                    <1>       shr    ecx, 1 ; dma buff size / 2 = half buffer size
51363                                  <1>
51364                                  <1>       ; 22/05/2017
51365 00011B4F F605[CC650100]01        <1>       test   byte [audio_flag], 1  ; Current flag value
51366 00011B56 7402                    <1>       jz     short sb_tlp1 ; EOL (Half Buffer 1 must be filled)
51367                                  <1>       ; FLAG (Half Buffer 2 must be filled)
51368 00011B58 01CF                    <1>       add    edi, ecx
51369                                  <1>       ; 15/05/2017
51370                                  <1> sb_tlp1:
51371 00011B5A 8B35[BC650100]          <1>       mov    esi, [audio_p_buffer] ; phy addr of audio buff
51372                                  <1>       ;rep   movsb
51373 00011B60 C1E902                  <1>       shr    ecx, 2 ; half buff size / 4
51374 00011B63 F3A5                    <1>       rep    movsd
51375                                  <1>       ;retn
51376                                  <1>
51377                                  <1>       ; 10/10/2017
51378                                  <1>       ; switch flag value
51379 00011B65 8035[CC650100]01        <1>       xor    byte [audio_flag], 1
51380                                  <1>
51381                                  <1>       ; 12/10/2017
51382                                  <1>       ; [audio_flag] = 0 : Playing dma half buffer 2 (odd intr count)
51383                                  <1>                          ; Next buffer (to update) is dma half buff 1
51384                                  <1>       ;             = 1 : Playing dma half buffer 1 (even intr count)
51385                                  <1>                          ; Next buffer (to update) is dma half buff 2
51386                                  <1>
51387                                  <1> sb_irq_h3:
```

```
51388                                  <1>       ;mov   byte [audio_busy], 0
51389                                  <1>
51390                                  <1>       ;pop   edi
51391                                  <1>       ;pop   esi
51392                                  <1>       ;pop   edx
51393                                  <1>       ;pop   ecx
51394                                  <1>       ;pop   ebx ; * must be restored !
51395                                  <1>       ;pop   eax ; * must be restored !
51396                                  <1>
51397 00011B6C C3                      <1>       retn
51398                                  <1>
51399                                  <1> sb16_volume:
51400                                  <1>       ; 22/10/2017
51401                                  <1>       ; mov [audio_master_volume_l], cl
51402                                  <1>       ; mov [audio_master_volume_h], ch
51403 00011B6D 66890D[DA650100]        <1>       mov    [audio_master_volume], cx
51404                                  <1> sb16_volume_initial:
51405 00011B74 6652                    <1>       push   dx ; DX (port address) must be saved
51406 00011B76 668B15[AA650100]        <1>       mov    dx, [audio_io_base]
51407 00011B7D 6683C204                <1>       add    dx, 4 ; Mixer chip address port
51408 00011B81 B022                    <1>       mov    al, 22h ; master volume
51409 00011B83 EE                      <1>       out    dx, al
51410 00011B84 6642                    <1>       inc    dx
51411 00011B86 8A25[DA650100]          <1>       mov    ah, [audio_master_volume_l]
51412 00011B8C C0EC02                  <1>       shr    ah, 2 ; 32 -> 8 level
51413 00011B8F C0E405                  <1>       shl    ah, 5 ; bit 5 to 7
51414 00011B92 A0[DB650100]            <1>       mov    al, [audio_master_volume_r]
51415 00011B97 C0E802                  <1>       shr    al, 2 ; 32 -> 8 level
51416                                  <1>       ;and   al, 0Fh
51417 00011B9A D0E0                    <1>       shl    al, 1 ; bit 1 to 3
51418 00011B9C 08E0                    <1>       or     al, ah
51419 00011B9E EE                      <1>       out    dx, al
51420 00011B9F 665A                    <1>       pop    dx ; DX (port address) must be restored
51421 00011BA1 C3                      <1>       retn
51422                                  <1>
51423                                  <1> sb16_pause:
51424 00011BA2 668B15[AA650100]        <1>       mov    dx, [audio_io_base]
51425 00011BA9 6683C20C                <1>       add    dx, 0Ch ; Command & Data Port
51426 00011BAD 803D[D4650100]10        <1>       cmp    byte [audio_bps], 16 ; 16 bit samples
51427 00011BB4 7404                    <1>       je     short sb_pause_1
51428                                  <1>       ; 8 bit samples
51429 00011BB6 B3D0                    <1>       mov    bl, 0D0h ; 8 bit DMA mode
51430 00011BB8 EB02                    <1>       jmp    short sb_pause_2
51431                                  <1> sb_pause_1:
51432                                  <1>       ; 16 bit samples
51433 00011BBA B3D5                    <1>       mov    bl, 0D5h ; 16 bit DMA mode
51434                                  <1> sb_pause_2:
51435                                  <1>       SbOut   bl ; bCommand
51436                                  <2> %%Wait:
51437 00011BBC EC                      <2>  in al, dx
51438 00011BBD 08C0                    <2>  or al, al
51439 00011BBF 78FB                    <2>  js short %%Wait
51440 00011BC1 88D8                    <2>  mov al, %1
51441 00011BC3 EE                      <2>  out dx, al
51442                                  <1> sb_pause_3:
51443 00011BC4 C3                      <1>       retn
51444                                  <1>
51445                                  <1> sb16_continue:
51446 00011BC5 668B15[AA650100]        <1>       mov    dx, [audio_io_base]
51447 00011BCC 6683C20C                <1>       add    dx, 0Ch ; Command & Data Port
51448 00011BD0 803D[D4650100]10        <1>       cmp    byte [audio_bps], 16 ; 16 bit samples
51449 00011BD7 7404                    <1>       je     short sb_cont_1
51450                                  <1>       ; 8 bit samples
51451 00011BD9 B3D4                    <1>       mov    bl, 0D4h ; 8 bit DMA mode
51452 00011BDB EB02                    <1>       jmp    short sb_cont_2
51453                                  <1> sb_cont_1:
51454                                  <1>       ; 16 bit samples
51455 00011BDD B3D6                    <1>       mov    bl, 0D6h ; 16 bit DMA mode
51456                                  <1> sb_cont_2:
51457                                  <1>       SbOut   bl ; bCommand
51458                                  <2> %%Wait:
51459 00011BDF EC                      <2>  in al, dx
51460 00011BE0 08C0                    <2>  or al, al
51461 00011BE2 78FB                    <2>  js short %%Wait
51462 00011BE4 88D8                    <2>  mov al, %1
51463 00011BE6 EE                      <2>  out dx, al
51464                                  <1> sb_cont_3:
51465 00011BE7 C3                      <1>       retn
51466                                  <1>
51467                                  <1> sb16_stop:
51468                                  <1>       ; 24/04/2017
51469 00011BE8 803D[D8650100]00        <1>       cmp    byte [audio_play_cmd], 0
51470 00011BEF 7648                    <1>       jna    short sb16_stop_4
51471                                  <1>
51472                                  <1>       ; 22/05/2017
51473 00011BF1 668B15[AA650100]        <1>       mov    dx, [audio_io_base]
51474 00011BF8 6683C20C                <1>       add    dx, 0Ch
51475                                  <1>
51476 00011BFC B3D9                    <1>       mov    bl, 0D9h ; exit auto-initialize 16 bit transfer
51477                                  <1>       ; stop  autoinitialized DMA transfer mode
51478 00011BFE 803D[D4650100]10        <1>       cmp    byte [audio_bps], 16 ; 16 bit samples
51479 00011C05 7402                    <1>       je     short sb16_stop_1
51480                                  <1>       ;mov   bl, 0DAh ; exit auto-initialize 8 bit transfer
51481 00011C07 FEC3                    <1>       inc    bl
51482                                  <1> sb16_stop_1:
51483                                  <1>       SbOut  bl ; exit auto-initialize transfer command
51484                                  <2> %%Wait:
51485 00011C09 EC                      <2>  in al, dx
51486 00011C0A 08C0                    <2>  or al, al
51487 00011C0C 78FB                    <2>  js short %%Wait
51488 00011C0E 88D8                    <2>  mov al, %1
51489 00011C10 EE                      <2>  out dx, al
51490                                  <1>
```

```
51491 00011C11 30C0                <1>       xor     al, al ; stops all DMA processes on selected channel
51492                              <1>
51493 00011C13 803D[D4650100]10    <1>       cmp    byte [audio_bps], 16 ; 16 bit samples
51494 00011C1A 7404                <1>       je     short sb16_stop_2
51495 00011C1C E60C                <1>       out    0Ch, al ; clear selected channel register
51496 00011C1E EB02                <1>       jmp    short sb16_stop_3
51497                              <1>
51498                              <1> sb16_stop_2:
51499 00011C20 E6D8                <1>       out    0D8h, al ; clear selected channel register
51500                              <1>
51501                              <1> sb16_stop_3:
51502 00011C22 C605[D8650100]00    <1>       mov    byte [audio_play_cmd], 0 ; stop !
51503                              <1> SbDone:
51504                              <1>       ;mov   dx, [audio_io_base]
51505                              <1>       ;add   dx, 0Ch
51506                              <1>       SbOut   0D0h
51507                              <2> %%Wait:
51508 00011C29 EC                  <2>  in al, dx
51509 00011C2A 08C0                <2>  or al, al
51510 00011C2C 78FB                <2>  js short %%Wait
51511 00011C2E B0D0                <2>  mov al, %1
51512 00011C30 EE                  <2>  out dx, al
51513                              <1>       SbOut   0D3h
51514                              <2> %%Wait:
51515 00011C31 EC                  <2>  in al, dx
51516 00011C32 08C0                <2>  or al, al
51517 00011C34 78FB                <2>  js short %%Wait
51518 00011C36 B0D3                <2>  mov al, %1
51519 00011C38 EE                  <2>  out dx, al
51520                              <1> sb16_stop_4:
51521 00011C39 C3                  <1>       retn
51522                              <1>
51523                              <1> sb16_reset:
51524                              <1>       ; 24/04/2017
51525 00011C3A 668B15[AA650100]    <1>       mov     dx, [audio_io_base] ; try to reset the DSP.
51526 00011C41 6683C206            <1>       add     dx, 06h
51527 00011C45 B001                <1>       mov    al, 1
51528 00011C47 EE                  <1>       out    dx, al
51529                              <1>
51530 00011C48 EC                  <1>       in     al, dx
51531 00011C49 EC                  <1>       in     al, dx
51532 00011C4A EC                  <1>       in     al, dx
51533 00011C4B EC                  <1>       in     al, dx
51534                              <1>
51535 00011C4C 30C0                <1>       xor     al, al
51536 00011C4E EE                  <1>       out    dx, al
51537                              <1>
51538 00011C4F 6683C208            <1>       add     dx, 08h
51539 00011C53 66B96400            <1>       mov    cx, 100
51540                              <1> sbrstWaitID:
51541 00011C57 EC                  <1>       in     al, dx
51542 00011C58 08C0                <1>       or      al, al
51543 00011C5A 7804                <1>       js      short sbrstGetID
51544 00011C5C E2F9                <1>       loop    sbrstWaitID
51545 00011C5E F9                  <1>       stc
51546 00011C5F C3                  <1>       retn
51547                              <1> sbrstGetID:
51548 00011C60 6683EA04            <1>       sub     dx, 04h
51549 00011C64 EC                  <1>       in     al, dx
51550 00011C65 3CAA                <1>       cmp     al, 0AAh
51551 00011C67 7406                <1>       je      short sb_rst_retn
51552 00011C69 6683C204            <1>       add     dx, 04h
51553 00011C6D E2E8                <1>       loop    sbrstWaitID
51554                              <1> sb_rst_retn:
51555 00011C6F C3                  <1>       retn
51556                              <1>
51557                              <1> ac97_codec_config:
51558                              <1>       ; 10/06/2017
51559                              <1>       ; 05/06/2017
51560                              <1>       ; 29/05/2017
51561                              <1>       ; 28/05/2017 (TRDOS 386, 'audio.s')
51562                              <1>       ; 07/11/2016 (Erdogan Tan)
51563                              <1>       ; Derived from 'codecConfig' procedure in 'CODEC.ASM'
51564                              <1>       ; .wav player for DOS by Jeff Leyda (02/09/2002)
51565                              <1>
51566                              <1>       ;; 'PLAYER.ASM'
51567                              <1>       ;; get ICH base address regs for mixer and bus master
51568                              <1>
51569                              <1> init_ac97_controller: ; 10/06/2017
51570 00011C70 A1[AC650100]        <1>       mov    eax, [audio_dev_id]
51571                              <1>         ;mov al, NAMBAR_REG
51572                              <1>         ;;call  pciRegRead16                ; read PCI registers 10-11
51573                              <1>         ;call      pciRegRead32
51574                              <1>       ;and   dx, IO_ADDR_MASK          ; mask off BIT0
51575                              <1>       ;;and  edx, IO_ADDR_MASK
51576                              <1>
51577                              <1>         ;mov [NAMBAR], dx              ; save audio mixer base addr
51578                              <1>
51579                              <1>         ;mov    al, NABMBAR_REG
51580                              <1>         ;;call     pciRegRead16
51581                              <1>         ;call      pciRegRead32
51582                              <1>       ;and   dx, 0FFC0h ; IO_ADDR_MASK
51583                              <1>       ;;and  edx, 0FFC0h
51584                              <1>
51585                              <1>         ;mov    [NABMBAR], dx               ; save bus master base addr
51586                              <1>
51587                              <1>       ;mov   eax, [audio_dev_id]
51588 00011C75 B004                <1>         mov     al, PCI_CMD_REG
51589                              <1>         ;call      pciRegRead8              ; read PCI command register
51590 00011C77 E840F8FFFF          <1>         call pciRegRead16
51591 00011C7C 80CA05              <1>       or      dl, IO_ENA+BM_ENA             ; enable IO and bus master
51592                              <1>         ;call      pciRegWrite8
51593 00011C7F E8A3F8FFFF          <1>       call   pciRegWrite16
```

```
51594                                  <1>
51595                                  <1>      ; 'CODEC.ASM'
51596                                  <1>
51597                                  <1>      ; enable codec, unmute stuff, set output rate
51598                                  <1> ;    ; entry: [audio_freq] = desired sample rate
51599                                  <1>
51600                                  <1> ;    mov         dx, [NAMBAR]
51601                                  <1> ;    add         dx, CODEC_EXT_AUDIO_CTRL_REG     ; 2Ah
51602                                  <1> ;    in          ax, dx
51603                                  <1> ;    or     ax, 1
51604                                  <1> ;    out    dx, ax                           ; Enable variable rate audio
51605                                  <1>
51606                                  <1> ;      ;call    delay1_4ms
51607                                  <1> ;      ;call    delay1_4ms
51608                                  <1> ;      ;call    delay1_4ms
51609                                  <1> ;      ;call    delay1_4ms
51610                                  <1>
51611                                  <1> ;    mov    ax, [audio_freq]          ; sample rate
51612                                  <1>
51613                                  <1> ;    mov         dx, [NAMBAR]
51614                                  <1> ;    add         dx, CODEC_PCM_FRONT_DACRATE_REG  ; 2Ch
51615                                  <1> ;    out    dx, ax                           ; out sample rate
51616                                  <1>
51617                                  <1> ;      ;call     delay1_4ms
51618                                  <1> ;      ;call     delay1_4ms
51619                                  <1> ;      ;call     delay1_4ms
51620                                  <1> ;      ;call     delay1_4ms
51621                                  <1>
51622                                  <1>      ;mov   dx, [NAMBAR]              ; mixer base address
51623                                  <1>        ;add dx, CODEC_RESET_REG              ; reset register
51624                                  <1>        ;mov ax, 42
51625                                  <1>      ;out   dx, ax                          ; reset
51626                                  <1>
51627                                  <1>      ;mov    dx, [NABMBAR]                  ; bus master base address
51628                                  <1>        ;add dx, GLOB_STS_REG
51629                                  <1>        ;mov ax, 2
51630                                  <1>      ;out   dx, ax
51631                                  <1>
51632 00011C84 E831F9FFFF             <1>        call    delay_100ms ; 29/05/2017
51633                                  <1>
51634                                  <1> init_ac97_codec:
51635                                  <1>      ; 10/06/2017
51636                                  <1>      ; 29/05/2017
51637                                  <1>      ; 28/05/2017 - Erdogan Tan (Ref: KolibriOS, intelac97.asm)
51638                                  <1>      ;
51639 00011C89 66BA2C00               <1>      mov    dx, GLOB_CNT_REG ; 2Ch
51640 00011C8D 660315[DE650100]       <1>      add    dx, [NABMBAR]
51641 00011C94 ED                     <1>      in     eax, dx
51642                                  <1>      ; ?
51643 00011C95 66BA3000               <1>      mov    dx, GLOB_STS_REG ; 30h
51644 00011C99 660315[DE650100]       <1>      add    dx, [NABMBAR]
51645 00011CA0 ED                     <1>      in     eax, dx
51646                                  <1>
51647 00011CA1 83F8FF                 <1>      cmp    eax, 0FFFFFFFFh ; -1
51648 00011CA4 744B                   <1>      je     short init_ac97_codec_err1
51649                                  <1>
51650 00011CA6 A900030010             <1>      test   eax, CTRL_ST_CREADY
51651 00011CAB 7507                   <1>      jnz    short _ac97_codec_ready
51652                                  <1>
51653 00011CAD E8EF020000             <1>      call   reset_ac97_codec
51654 00011CB2 723E                   <1>      jc     short init_ac97_codec_err2
51655                                  <1>
51656                                  <1> _ac97_codec_ready:
51657 00011CB4 668B15[DC650100]       <1>      mov    dx, [NAMBAR]
51658                                  <1>      ;add   dx, 0 ; ac_reg_0 ; reset register
51659 00011CBB 66EF                   <1>      out    dx, ax
51660                                  <1>
51661 00011CBD 31C0                   <1>      xor    eax, eax ; 0
51662 00011CBF 668B15[DC650100]       <1>      mov    dx, [NAMBAR]
51663 00011CC6 6683C226               <1>      add    dx, CODEC_REG_POWERDOWN
51664 00011CCA 66EF                   <1>      out    dx, ax
51665                                  <1>
51666                                  <1>      ; 10/06/2017
51667                                  <1>      ; 29/05/2017
51668                                  <1>      ; wait for 1 second
51669 00011CCC B9E8030000             <1>      mov    ecx, 1000 ; 1000*0.25ms = 1s
51670                                  <1> _ac97_codec_rloop:
51671 00011CD1 E8F1F8FFFF             <1>      call   delay1_4ms
51672 00011CD6 E8ECF8FFFF             <1>      call   delay1_4ms
51673 00011CDB E8E7F8FFFF             <1>      call   delay1_4ms
51674 00011CE0 E8E2F8FFFF             <1>      call   delay1_4ms
51675                                  <1>      ;mov   dx, [NAMBAR]
51676                                  <1>      ;add   dx, CODEC_REG_POWERDOWN
51677 00011CE5 66ED                   <1>      in     ax, dx
51678 00011CE7 6683E00F               <1>      and    ax, 0Fh
51679 00011CEB 3C0F                   <1>      cmp    al, 0Fh
51680 00011CED 7404                   <1>      je     short _ac97_codec_init_ok
51681 00011CEF E2E0                   <1>      loop   _ac97_codec_rloop
51682                                  <1>
51683                                  <1> init_ac97_codec_err1:
51684 00011CF1 F9                     <1>      stc
51685                                  <1> init_ac97_codec_err2:
51686 00011CF2 C3                     <1>      retn
51687                                  <1>
51688                                  <1> _ac97_codec_init_ok:
51689 00011CF3 B002                   <1>        mov     al, 2 ; force set 16-bit 2-channel PCM
51690 00011CF5 66BA2C00               <1>      mov    dx, GLOB_CNT_REG ; 2Ch
51691 00011CF9 660315[DE650100]       <1>      add    dx, [NABMBAR]
51692 00011D00 EF                     <1>      out    dx, eax
51693                                  <1>
51694                                  <1>      ;call  delay1_4ms
51695                                  <1>
51696                                  <1>      ; 10/06/2017
```

```
51697 00011D01 E849020000          <1>       call   reset_ac97_controller
51698                              <1>
51699                              <1> ;     call   setup_ac97_codec
51700                              <1> ;
51701                              <1> ;detect_ac97_codec:
51702                              <1> ;     retn
51703                              <1>
51704                              <1> setup_ac97_codec:
51705                              <1>       ; 10/06/2017
51706                              <1>       ; 29/05/2017
51707 00011D06 B802020000          <1>       mov     eax, 0202h
51708 00011D0B 66A3[DA650100]      <1>       mov    [audio_master_volume], ax
51709 00011D11 66B81F1F            <1>       mov    ax, 1F1Fh ; 31, 31
51710                              <1>
51711 00011D15 668B15[DC650100]    <1>       mov     dx, [NAMBAR]
51712 00011D1C 6683C202            <1>       add     dx, CODEC_MASTER_VOL_REG        ;02h
51713 00011D20 6631C0              <1>       xor    ax, ax       ; volume attenuation = 0 (max. volume)
51714 00011D23 66EF                <1>       out     dx, ax
51715                              <1>
51716 00011D25 668B15[DC650100]    <1>       mov     dx, [NAMBAR]
51717 00011D2C 6683C206            <1>       add     dx, CODEC_MASTER_MONO_VOL_REG   ;06h
51718                              <1>       ;xor   ax, ax
51719 00011D30 66EF                <1>       out     dx, ax
51720                              <1>
51721 00011D32 668B15[DC650100]    <1>       mov     dx, [NAMBAR]
51722 00011D39 6683C20A            <1>       add     dx, CODEC_PCBEEP_VOL_REG        ;0Ah
51723                              <1>       ;xor    ax, ax
51724 00011D3D 66EF                <1>       out     dx, ax
51725                              <1>
51726 00011D3F 668B15[DC650100]    <1>       mov     dx, [NAMBAR]
51727 00011D46 6683C218            <1>       add     dx, CODEC_PCM_OUT_REG          ;18h
51728                              <1>       ;xor    ax, ax
51729 00011D4A 66EF                <1>       out     dx, ax
51730                              <1>
51731 00011D4C 66B80880            <1>       mov     ax,  8008h ; Mute
51732 00011D50 668B15[DC650100]    <1>       mov     dx, [NAMBAR]
51733 00011D57 6683C20C            <1>       add    dx, 0Ch       ; AC97_PHONE_VOL ; TAD Input (Mono)
51734 00011D5B 66EF                <1>       out     dx, ax
51735                              <1>
51736 00011D5D 66B80808            <1>         mov  ax,  0808h
51737 00011D61 668B15[DC650100]    <1>       mov     dx, [NAMBAR]
51738 00011D68 6683C210            <1>         add  dx, CODEC_LINE_IN_VOL_REG ; 10h ; Line Input (Stereo)
51739 00011D6C 66EF                <1>       out     dx, ax
51740                              <1>
51741                              <1>       ;mov   ax,  0808h
51742 00011D6E 668B15[DC650100]    <1>       mov     dx, [NAMBAR]
51743 00011D75 6683C212            <1>         add  dx, CODEC_CD_VOL_REG ; 12h ; CR Input (Stereo)
51744 00011D79 66EF                <1>       out     dx, ax
51745                              <1>
51746                              <1>       ;mov   ax,  0808h
51747 00011D7B 668B15[DC650100]    <1>       mov     dx, [NAMBAR]
51748 00011D82 6683C216            <1>         add  dx, CODEC_AUX_VOL_REG ; 16h ; Aux Input (Stereo)
51749 00011D86 66EF                <1>       out     dx, ax
51750                              <1>
51751                              <1>         ;call    delay1_4ms
51752                              <1>         ;call    delay1_4ms
51753                              <1>         ;call    delay1_4ms
51754                              <1>         ;call    delay1_4ms
51755                              <1>
51756                              <1> detect_ac97_codec:
51757 00011D88 C3                  <1>         retn
51758                              <1>
51759                              <1> set_ac97_bdl: ; Set AC97 (ICH) Buffer Descriptor List
51760                              <1>       ; 17/06/2017
51761                              <1>       ; 11/06/2017
51762                              <1>       ; 28/05/2017
51763                              <1>       ; eax = dma buffer address = [audio_DMA_buff]
51764                              <1>       ; ecx = dma buffer buffer size = [audio_dmabuff_size]
51765                              <1>
51766 00011D89 D1E9                <1>       shr    ecx, 1 ; dma half buffer size
51767 00011D8B 89CE                <1>       mov    esi, ecx
51768                              <1>
51769 00011D8D BF[E0650100]        <1>         mov     edi, audio_bdl_buff    ; get BDL address
51770 00011D92 B910000000          <1>         mov     ecx, 32 / 2            ; make 32 entries in BDL
51771                              <1>
51772 00011D97 EB05                <1>       jmp    short s_ac97_bdl1
51773                              <1>
51774                              <1> s_ac97_bdl0:
51775                              <1>       ; set buffer descriptor 0 to start of data file in memory
51776                              <1>
51777 00011D99 A1[C4650100]        <1>       mov    eax, [audio_dma_buff]     ; Physical address of DMA buffer
51778                              <1>
51779                              <1> s_ac97_bdl1:
51780 00011D9E AB                  <1>       stosd                     ; store dmabuffer1 address
51781                              <1>
51782 00011D9F 89C2                <1>       mov    edx, eax
51783                              <1>
51784                              <1> ;
51785                              <1> ; Buffer Descriptors List
51786                              <1> ; As stated earlier, each buffer descriptor list is a set of (up to) 32
51787                              <1> ; descriptors, each 8 bytes in length.  Bytes 0-3 of a descriptor entry point
51788                              <1> ; to a chunk of memory to either play from or record to.  Bytes 4-7 of an
51789                              <1> ; entry describe various control things detailed below.
51790                              <1> ;
51791                              <1> ; Buffer pointers must always be aligned on a Dword boundry.
51792                              <1> ;
51793                              <1> ;
51794                              <1>
51795                              <1> ;IOC                    equ     BIT31 ; Fire an interrupt whenever this
51796                              <1>                                       ; buffer is complete.
51797                              <1>
51798                              <1> ;BUP                    equ     BIT30  ; Buffer Underrun Policy.
51799                              <1>                                       ; if this buffer is the last buffer
```

```
                                        <1>                                     ; in a playback, fill the remaining
                                        <1>                                     ; samples with 0 (silence) or not.
                                        <1>                                     ; It's a good idea to set this to 1
                                        <1>                                     ; for the last buffer in playback,
                                        <1>                                     ; otherwise you're likely to get a lot
                                        <1>                                     ; of noise at the end of the sound.
                                        <1>
                                        <1> ;
                                        <1> ; Bits 15:0 contain the length of the buffer, in number of samples, which
                                        <1> ; are 16 bits each, coupled in left and right pairs, or 32bits each.
                                        <1> ; Luckily for us, that's the same format as .wav files.
                                        <1> ;
                                        <1> ; A value of FFFF is 65536 samples.  Running at 44.1Khz, that's just about
                                        <1> ; 1.5 seconds of sample time.  FFFF * 32bits is 1FFFFh bytes or 128k of data.
                                        <1> ;
                                        <1> ; A value of 0 in these bits means play no samples.
                                        <1> ;
                                        <1>
00011DA1 89F0                           <1>     mov    eax, esi ; DMA half buffer size
00011DA3 01C2                           <1>     add    edx, eax
00011DA5 D1E8                           <1>     shr    eax, 1 ; count of 16 bit samples
                                        <1>     ;or    eax, IOC+BUP
00011DA7 0D00000080                     <1>     or     eax, IOC ; 11/06/2017
00011DAC AB                             <1>     stosd
                                        <1>
                                        <1> ; 2nd buffer:
                                        <1>
00011DAD 89D0                           <1>       mov  eax, edx ; Physical address of the 2nd half of DMA buffer
00011DAF AB                             <1>     stosd         ; store dmabuffer2 address
                                        <1>
                                        <1> ; set length to [audio_dmabuff_size]/2
                                        <1> ; Set control (bits 31:16) to BUP, bits 15:0=number of samples
                                        <1> ;
00011DB0 89F0                           <1>     mov    eax, esi ; DMA half buffer size
00011DB2 D1E8                           <1>     shr    eax, 1 ; count of 16 bit samples
                                        <1>     ;or    eax, IOC+BUP
00011DB4 0D00000080                     <1>     or     eax, IOC ; 11/06/2017
00011DB9 AB                             <1>     stosd
                                        <1>
00011DBA E2DD                           <1>     loop    s_ac97_bdl0
                                        <1>
00011DBC C3                             <1>     retn
                                        <1>
                                        <1> ac97_start_play:
                                        <1>     ; 28/05/2017
                                        <1>     ; Derived from 'playWav' procedure in 'ICHWAV.ASM'
                                        <1>     ; .wav player for DOS by Jeff Leyda (02/09/2002)
                                        <1>
                                        <1>     ; set output rate
                                        <1>     ; entry: [audio_freq] = desired sample rate
                                        <1>
00011DBD 668B15[DC650100]               <1>     mov         dx, [NAMBAR]
00011DC4 6683C22A                       <1>     add         dx, CODEC_EXT_AUDIO_CTRL_REG     ; 2Ah
00011DC8 66ED                           <1>     in          ax, dx
00011DCA 6683C801                       <1>     or     ax, 1
00011DCE 66EF                           <1>     out    dx, ax                           ; Enable variable rate audio
                                        <1>
                                        <1>      ;call    delay1_4ms
                                        <1>      ;call    delay1_4ms
                                        <1>      ;call    delay1_4ms
                                        <1>      ;call    delay1_4ms
                                        <1>
00011DD0 66A1[D6650100]                 <1>     mov    ax, [audio_freq]          ; sample rate
                                        <1>
00011DD6 668B15[DC650100]               <1>     mov         dx, [NAMBAR]
00011DDD 6683C22C                       <1>     add         dx, CODEC_PCM_FRONT_DACRATE_REG  ; 2Ch
00011DE1 66EF                           <1>     out    dx, ax                           ; out sample rate
                                        <1>
                                        <1>      ;call    delay1_4ms
                                        <1>      ;call    delay1_4ms
                                        <1>      ;call    delay1_4ms
                                        <1>      ;call    delay1_4ms
                                        <1>
                                        <1> ;
                                        <1> ; register reset the DMA engine.  This may cause a pop noise on the output
                                        <1> ; lines when the device is reset.  Prolly a better idea to mute output, then
                                        <1> ; reset.
                                        <1> ;
00011DE3 668B15[DE650100]               <1>      mov     dx, [NABMBAR]
00011DEA 6683C21B                       <1>      add     dx, PO_CR_REG                   ; set pointer to Cntl reg
00011DEE B002                           <1>      mov     al, RR                          ; set reset
00011DF0 EE                             <1>      out     dx, al                          ; self clearing bit
                                        <1> ;
                                        <1> ;     mov    edi, audio_bdl_buff
                                        <1> ;     mov    edx, [audio_dmabuff_size]
                                        <1> ;     shr    edx, 1
                                        <1> ;     mov    ecx, 32/2
                                        <1> ;ac97_set_bdl_buffer:
                                        <1> ;     ; 1st half of DMA buffer
                                        <1> ;     mov    eax, [audio_dma_buff]
                                        <1> ;     push   eax
                                        <1> ;     stosd
                                        <1> ;     mov    eax, edx ; dma buffer size / 2
                                        <1> ;     or     eax, IOC+BUP
                                        <1> ;     stosd
                                        <1> ;     pop    eax
                                        <1> ;     ; 2nd half of DMA buffer
                                        <1> ;     add    eax, edx
                                        <1> ;     stosd
                                        <1> ;     mov    eax, edx ; dma buffer size / 2
                                        <1> ;     or     eax, IOC+BUP
                                        <1> ;     stosd
                                        <1> ;     loop   ac97_set_bdl_buffer
```

```
                                <1>
                                <1> ; tell the DMA engine where to find our list of Buffer Descriptors.
                                <1> ; this 32bit value is a flat mode memory offset (ie no segment:offset)
                                <1> ;
                                <1> ; write NABMBAR+10h with offset of buffer descriptor list
                                <1> ;
00011DF1 B8[E0650100]           <1>         mov  eax, audio_bdl_buff
00011DF6 668B15[DE650100]       <1>       mov    dx, [NABMBAR]
00011DFD 6683C210               <1>       add    dx, PO_BDBAR_REG
00011E01 EF                     <1>       out    dx, eax
                                <1> ;
                                <1> ; All set.  Let's play some music.
                                <1> ;
                                <1> ;
00011E02 B81F000000             <1>       mov    eax, 31
00011E07 E816000000             <1>       call    set_ac97_LastValidIndex
                                <1>
00011E0C C605[D8650100]01       <1>       mov    byte [audio_play_cmd], 1 ; play command (do not stop) !
                                <1>
                                <1> ac97_play: ; continue to play (after pause)
                                <1>       ; 11/06/2017
                                <1>       ; 29/05/2017
                                <1>       ; 28/05/2017
00011E13 668B15[DE650100]       <1>         mov     dx, [NABMBAR]
00011E1A 6683C21B               <1>         add     dx, PO_CR_REG          ; PCM out control register
00011E1E B011                   <1>         mov  al, IOCE+RPBM ; 29/05/2017
                                <1>         ;mov al, 1Dh ; (Ref: KolibriOS, intelac97.asm, 'play:')
00011E20 EE                     <1>       out     dx, al                   ; set start!
                                <1>
                                <1>       ;mov   byte [audio_play_cmd], 1 ; play command (do not stop) !
                                <1>
00011E21 C3                     <1>       retn
                                <1>
                                <1> ;input AL = index # to stop on
                                <1> set_ac97_LastValidIndex:
                                <1>       ; 28/05/2017
                                <1>       ; Derived from 'setLastValidIndex' procedure in 'ICHWAV.ASM'
                                <1>       ; .wav player for DOS by Jeff Leyda (02/09/2002)
00011E22 668B15[DE650100]       <1>       mov    dx, [NABMBAR]
00011E29 6683C215               <1>       add    dx, PO_LVI_REG
00011E2D EE                     <1>         out     dx, al
                                <1>       ;mov   [audio_lvi], al ; for ac97_int_handler
00011E2E C3                     <1>       retn
                                <1>
                                <1> ac97_volume:
                                <1>       ; 28/05/2017
                                <1>       ; bl = component (0 = master/playback/lineout volume)
                                <1>       ; cl = left channel volume level (0 to 31)
                                <1>       ; ch = right channel volume level (0 to 31)
                                <1>
00011E2F 08DB                   <1>       or     bl, bl
00011E31 7523                   <1>       jnz    short ac97_vol_1 ; temporary !
00011E33 66B81F1F               <1>       mov    ax, 1F1Fh ; 31,31
00011E37 38C1                   <1>       cmp    cl, al
00011E39 771B                   <1>       ja     short ac97_vol_1 ; temporary !
00011E3B 38E5                   <1>       cmp    ch, ah
00011E3D 7717                   <1>       ja     short ac97_vol_1 ; temporary !
00011E3F 66890D[DA650100]       <1>       mov    [audio_master_volume], cx
00011E46 6629C8                 <1>       sub    ax, cx
00011E49 668B15[DC650100]       <1>       mov     dx, [NAMBAR]
00011E50 6683C202               <1>       add     dx, CODEC_MASTER_VOL_REG  ; 02h ; Line Out
00011E54 66EF                   <1>       out     dx, ax
                                <1> ac97_vol_1:
00011E56 C3                     <1>       retn
                                <1>
                                <1> ac97_int_handler:
                                <1>       ; 12/10/2017
                                <1>       ; 10/10/2017
                                <1>       ; 09/10/2017
                                <1>       ; 13/06/2017, 13/06/2017
                                <1>       ; 10/06/2017, 11/06/2017
                                <1>       ; Interrupt Handler for AC97 (ICH) Audio Controller
                                <1>       ; Note: called by 'dev_IRQ_service'
                                <1>       ; 28/05/2017
                                <1>
                                <1>       ;push  eax ; * must be saved !
                                <1>       ;push  edx
                                <1>       ;push  ecx
                                <1>       ;push  ebx ; * must be saved !
                                <1>       ;push  esi
                                <1>       ;push  edi
                                <1>
                                <1>       ;cmp   byte [audio_busy], 1
                                <1>       ;jnb   _ac97_ih2    ; busy !
                                <1>
00011E57 66BA3000               <1>         mov     dx, GLOB_STS_REG
00011E5B 660315[DE650100]       <1>         add  dx, [NABMBAR]
00011E62 ED                     <1>       in     eax, dx
                                <1>
00011E63 83F8FF                 <1>         cmp     eax, 0FFFFFFFFh ; -1
00011E66 0F849A000000           <1>         je      _ac97_ih3 ; exit
                                <1>
00011E6C A940000000             <1>         test    eax, 40h ; PCM Out Interrupt
00011E71 750E                   <1>         jnz     short _ac97_ih0
                                <1>
00011E73 85C0                   <1>       test   eax, eax
00011E75 0F848B000000           <1>       jz     _ac97_ih3 ; exit
                                <1>
                                <1>       ;mov   dx, GLOB_STS_REG
                                <1>         ;add dx, [NABMBAR]
00011E7B EF                     <1>       out    dx, eax
                                <1>
00011E7C E985000000             <1>       jmp    _ac97_ih3 ; exit
```

```
52006                                  <1>
52007                                  <1> _ac97_ih0:
52008 00011E81 50                      <1>         push   eax
52009                                  <1>         ; 09/10/2017
52010 00011E82 803D[D8650100]01        <1>         cmp    byte [audio_play_cmd], 1
52011 00011E89 727C                    <1>         jb     short _ac97_ih4 ; stop command !
52012                                  <1>
52013                                  <1>         ;mov   byte [audio_busy], 1
52014                                  <1>
52015                                  <1>         ;mov   al, 10h
52016                                  <1>         ;mov   dx, PO_CR_REG
52017                                  <1>           ;add dx, [NABMBAR]
52018                                  <1>         ;out   dx, al
52019                                  <1>
52020 00011E8B 66B81C00                <1>         mov    ax, 1Ch ; FIFOE(=16)+BCIS(=8)+LVBCI(=4)
52021 00011E8F 66BA1600                <1>         mov    dx, PO_SR_REG
52022 00011E93 660315[DE650100]        <1>           add  dx, [NABMBAR]
52023 00011E9A 66EF                    <1>         out    dx, ax
52024                                  <1>
52025 00011E9C 66BA1400                <1>         mov    dx, PO_CIV_REG
52026 00011EA0 660315[DE650100]        <1>           add  dx, [NABMBAR]
52027 00011EA7 EC                      <1>         in     al, dx
52028                                  <1>
52029                                  <1>         ;cmp   al, [audio_civ] ; [audio_flag]
52030                                  <1>         ;je    short _ac97_ih2
52031                                  <1>
52032 00011EA8 A2[D9650100]            <1>         mov    [audio_civ], al
52033 00011EAD FEC8                    <1>         dec    al
52034                                  <1>         ;inc   al ; 11/06/2017
52035 00011EAF 241F                    <1>         and    al, 1Fh
52036                                  <1>
52037 00011EB1 66BA1500                <1>           mov     dx, PO_LVI_REG
52038 00011EB5 660315[DE650100]        <1>           add  dx, [NABMBAR]
52039 00011EBC EE                      <1>           out  dx, al
52040                                  <1>
52041                                  <1>         ; 12/10/2017
52042 00011EBD A0[D9650100]            <1>         mov    al, [audio_civ]
52043 00011EC2 FEC0                    <1>         inc    al
52044 00011EC4 2401                    <1>         and    al, 1
52045 00011EC6 A2[CC650100]            <1>         mov    [audio_flag], al
52046                                  <1>         ;; [audio_flag] : 0 = Buffer 1, 1 = Buffer 2
52047                                  <1>         ;
52048 00011ECB 58                      <1>         pop    eax
52049                                  <1>         ;
52050 00011ECC 83E040                  <1>         and     eax, 40h
52051 00011ECF 668B15[DE650100]        <1>           mov  dx, [NABMBAR]
52052 00011ED6 6683C230                <1>         add    dx, GLOB_STS_REG
52053 00011EDA EF                      <1>         out    dx, eax
52054                                  <1>
52055                                  <1>         ;; 13/06/2017
52056                                  <1>         ;mov   al, 11h ; IOCE + RPBM
52057                                  <1>         ;mov   dx, PO_CR_REG
52058                                  <1>           ;add dx, [NABMBAR]
52059                                  <1>         ;out   dx, al
52060                                  <1>
52061                                  <1> ac97_tuneloop:
52062                                  <1>         ; 09/10/2017
52063 00011EDB 8B3D[C4650100]          <1>         mov    edi, [audio_dma_buff]
52064 00011EE1 8B0D[C8650100]          <1>         mov    ecx, [audio_dmabuff_size]
52065 00011EE7 D1E9                    <1>         shr    ecx, 1 ; dma buff size / 2 = half buffer size
52066                                  <1>
52067                                  <1>         ; 12/10/2017
52068 00011EE9 803D[CC650100]00        <1>         cmp    byte [audio_flag], 0
52069 00011EF0 7702                    <1>         ja     short _ac97_ih1  ; Playing Half Buffer 2 (Current: FLAG)
52070                                  <1>         ; Playing Half Buffer 1 (Current: EOL)
52071 00011EF2 01CF                    <1>         add    edi, ecx
52072                                  <1> _ac97_ih1:
52073                                  <1>         ; Update half buffer 2 while playing half buffer 1 (next: FLAG)
52074                                  <1>         ; Update half buffer 1 while playing half buffer 2 (next: EOL)
52075                                  <1>
52076 00011EF4 8B35[BC650100]          <1>         mov    esi, [audio_p_buffer] ; phy addr of audio buff
52077 00011EFA C1E902                  <1>         shr    ecx, 2 ; half buff size / 4
52078 00011EFD F3A5                    <1>         rep    movsd
52079                                  <1>
52080                                  <1>         ; 10/10/2017
52081                                  <1>         ; switch flag value
52082 00011EFF 8035[CC650100]01        <1>         xor    byte [audio_flag], 1
52083                                  <1>         ; 12/10/2017
52084                                  <1>         ; [audio_flag] = 0 : Playing dma half buffer 2 (even index value)
52085                                  <1>                              ; Next buffer (to update) is dma half buff 1
52086                                  <1>         ;             = 1 : Playing dma half buffer 1 (odd index value)
52087                                  <1>                              ; Next buffer (to update) is dma half buff 2
52088                                  <1>
52089                                  <1> _ac97_ih2:
52090                                  <1>         ;mov   byte [audio_busy], 0
52091                                  <1> _ac97_ih3:
52092                                  <1>         ;pop   edi
52093                                  <1>         ;pop   esi
52094                                  <1>         ;pop   ebx ; * must be restored !
52095                                  <1>         ;pop   ecx
52096                                  <1>         ;pop   edx
52097                                  <1>         ;pop   eax ; * must be restored !
52098                                  <1>
52099 00011F06 C3                      <1>         retn
52100                                  <1>
52101                                  <1> _ac97_ih4:
52102                                  <1>         ; 09/10/2017
52103 00011F07 E818000000              <1>         call   _ac97_stop
52104                                  <1>         ;
52105 00011F0C 58                      <1>         pop    eax
52106                                  <1>         ;
52107 00011F0D 83E040                  <1>         and     eax, 40h
52108 00011F10 668B15[DE650100]        <1>           mov  dx, [NABMBAR]
```

```
52109 00011F17 6683C230            <1>       add    dx, GLOB_STS_REG
52110 00011F1B EF                  <1>       out    dx, eax
52111                              <1>
52112                              <1>       ;; 13/06/2017
52113                              <1>       ;mov   al, 11h ; IOCE + RPBM
52114                              <1>       ;dx, PO_CR_REG
52115                              <1>         ;add dx, [NABMBAR]
52116                              <1>       ;out   dx, al
52117                              <1>
52118                              <1>       ; 10/10/2017
52119                              <1>       ;jmp   short _ac97_ih3  ; exit
52120 00011F1C C3                  <1>       retn
52121                              <1>
52122                              <1> ac97_stop:
52123                              <1>       ; 28/05/2017
52124 00011F1D C605[D8650100]00    <1>       mov    byte [audio_play_cmd], 0 ; stop !
52125                              <1> _ac97_stop: ; 09/10/2017
52126                              <1>       ; 29/05/2017
52127                              <1>       ;mov   dx, [NABMBAR]
52128                              <1>       ;add   dx, PO_CR_REG
52129                              <1>       ;mov   al, 0
52130                              <1>       ;out   dx, al
52131                              <1>
52132                              <1>       ; 11/06/2017
52133 00011F24 30C0                <1>       xor    al, al ; 0
52134 00011F26 E813000000          <1>       call   ac97_po_cmd
52135                              <1>
52136                              <1>       ; (Ref: KolibriOS, intelac97.asm, 'stop:')
52137                              <1>       ; Clear FIFOE, BCIS, LVBCI (Ref: Intel ICH hub manual)
52138 00011F2B 66B81C00            <1>       mov     ax, 1Ch
52139 00011F2F 668B15[DE650100]    <1>       mov     dx, [NABMBAR]
52140 00011F36 6683C216            <1>       add     dx, PO_SR_REG
52141 00011F3A 66EF                <1>       out     dx, ax
52142                              <1>
52143                              <1>       ;retn
52144                              <1>
52145                              <1>       ; 11/06/2017
52146 00011F3C B002                <1>       mov     al, RR
52147                              <1> ac97_po_cmd:
52148                              <1>        ;11/06/2017
52149                              <1>       ; 29/05/2017
52150 00011F3E 668B15[DE650100]    <1>       mov     dx, [NABMBAR]
52151 00011F45 6683C21B            <1>         add     dx, PO_CR_REG          ; PCM out control register
52152 00011F49 EE                  <1>       out    dx, al
52153 00011F4A C3                  <1>       retn
52154                              <1>
52155                              <1> ac97_pause:
52156                              <1>       ; 11/06/2017
52157                              <1>       ; 29/05/2017
52158 00011F4B B010                <1>       mov    al, IOCE
52159 00011F4D EBEF                <1>       jmp    short ac97_po_cmd
52160                              <1>
52161                              <1> reset_ac97_controller:
52162                              <1>       ; 10/06/2017
52163                              <1>       ; 29/05/2017
52164                              <1>       ; 28/05/2017
52165                              <1>       ; reset AC97 audio controller registers
52166 00011F4F 31C0                <1>       xor     eax, eax
52167 00011F51 66BA0B00            <1>         mov  dx, PI_CR_REG
52168 00011F55 660315[DE650100]    <1>       add    dx, [NABMBAR]
52169 00011F5C EE                  <1>       out     dx, al
52170                              <1>
52171 00011F5D 66BA1B00            <1>         mov     dx, PO_CR_REG
52172 00011F61 660315[DE650100]    <1>       add    dx, [NABMBAR]
52173 00011F68 EE                  <1>       out     dx, al
52174                              <1>
52175 00011F69 66BA2B00            <1>         mov     dx, MC_CR_REG
52176 00011F6D 660315[DE650100]    <1>       add    dx, [NABMBAR]
52177 00011F74 EE                  <1>       out     dx, al
52178                              <1>
52179 00011F75 B002                <1>         mov     al, RR
52180 00011F77 66BA0B00            <1>         mov     dx, PI_CR_REG
52181 00011F7B 660315[DE650100]    <1>       add    dx, [NABMBAR]
52182 00011F82 EE                  <1>       out     dx, al
52183                              <1>
52184 00011F83 66BA1B00            <1>         mov     dx, PO_CR_REG
52185 00011F87 660315[DE650100]    <1>       add    dx, [NABMBAR]
52186 00011F8E EE                  <1>       out     dx, al
52187                              <1>
52188 00011F8F 66BA2B00            <1>         mov     dx, MC_CR_REG
52189 00011F93 660315[DE650100]    <1>       add    dx, [NABMBAR]
52190 00011F9A EE                  <1>       out     dx, al
52191                              <1>
52192 00011F9B C3                  <1>       retn
52193                              <1>
52194                              <1> ac97_reset:
52195                              <1>       ; 10/06/2017
52196                              <1>       ; 29/05/2017
52197                              <1>       ; 28/05/2017
52198 00011F9C E8AEFFFFFF          <1>       call   reset_ac97_controller
52199                              <1>       ; 29/05/2017
52200                              <1>       ;jmp   reset_ac97_codec
52201                              <1> reset_ac97_codec:
52202                              <1>       ; 28/05/2017 - Erdogan Tan (Ref: KolibriOS, intelac97.asm)
52203 00011FA1 66BA2C00            <1>       mov    dx, GLOB_CNT_REG ; 2Ch
52204 00011FA5 660315[DE650100]    <1>       add    dx, [NABMBAR]
52205 00011FAC ED                  <1>       in     eax, dx
52206                              <1>
52207 00011FAD A902000000          <1>       test   eax, 2
52208 00011FB2 7407                <1>       jz     short _r_ac97codec_cold
52209                              <1>
52210 00011FB4 E80F000000          <1>       call   warm_ac97codec_reset
52211 00011FB9 7308                <1>       jnc    short _r_ac97codec_ok
```

```
                                <1> _r_ac97codec_cold:
00011FBB E83D000000             <1>         call    cold_ac97codec_reset
00011FC0 7301                   <1>         jnc     short _r_ac97codec_ok
                                <1>
                                <1>       ; 16/04/2017
                                <1>         ;xor     eax, eax         ; timeout error
                                <1>             ;stc
00011FC2 C3                     <1>       retn
                                <1>
                                <1> _r_ac97codec_ok:
00011FC3 31C0                   <1>         xor     eax, eax
                                <1>         ;mov al, VIA_ACLINK_C00_READY ; 1
00011FC5 FEC0                   <1>         inc  al
00011FC7 C3                     <1>       retn
                                <1>
                                <1> warm_ac97codec_reset:
                                <1>       ; 28/05/2017 - Erdogan Tan (Ref: KolibriOS, intelac97.asm)
00011FC8 B806000000             <1>         mov     eax, 6
00011FCD 66BA2C00               <1>       mov    dx, GLOB_CNT_REG ; 2Ch
00011FD1 660315[DE650100]       <1>       add    dx, [NABMBAR]
00011FD8 EF                     <1>       out    dx, eax
                                <1>
00011FD9 B90A000000             <1>       mov    ecx, 10      ; total 1s
                                <1> _warm_ac97c_rst_wait:
00011FDE 51                     <1>       push   ecx
00011FDF E8D6F5FFFF             <1>       call   delay_100ms
00011FE4 59                     <1>       pop    ecx
                                <1>
00011FE5 66BA3000               <1>       mov    dx, GLOB_STS_REG ; 30h
00011FE9 660315[DE650100]       <1>       add    dx, [NABMBAR]
00011FF0 ED                     <1>       in     eax, dx
                                <1>
00011FF1 A900030010             <1>       test   eax, CTRL_ST_CREADY
00011FF6 7504                   <1>       jnz    short _warm_ac97c_rst_ok
                                <1>
00011FF8 49                     <1>         dec     ecx
00011FF9 75E3                   <1>         jnz     short _warm_ac97c_rst_wait
                                <1>
                                <1> _warm_ac97c_rst_fail:
00011FFB F9                     <1>         stc
                                <1> _warm_ac97c_rst_ok:
00011FFC C3                     <1>       retn
                                <1>
                                <1> cold_ac97codec_reset:
                                <1>       ; 28/05/2017 - Erdogan Tan (Ref: KolibriOS, intelac97.asm)
00011FFD B802000000             <1>         mov     eax, 2
00012002 66BA2C00               <1>       mov    dx, GLOB_CNT_REG ; 2Ch
00012006 660315[DE650100]       <1>       add    dx, [NABMBAR]
0001200D EF                     <1>       out    dx, eax
                                <1>
0001200E E8A7F5FFFF             <1>       call   delay_100ms  ; wait 100 ms
00012013 E8A2F5FFFF             <1>       call   delay_100ms  ; wait 100 ms
00012018 E89DF5FFFF             <1>       call   delay_100ms  ; wait 100 ms
0001201D E898F5FFFF             <1>       call   delay_100ms  ; wait 100 ms
                                <1>
00012022 B910000000             <1>       mov    ecx, 16      ; total 20*100 ms = 2s
                                <1> _cold_ac97c_rst_wait:
00012027 66BA3000               <1>       mov    dx, GLOB_STS_REG ; 30h
0001202B 660315[DE650100]       <1>       add    dx, [NABMBAR]
00012032 ED                     <1>       in     eax, dx
                                <1>
00012033 A900030010             <1>       test   eax, CTRL_ST_CREADY
00012038 750B                   <1>       jnz    short _cold_ac97c_rst_ok
                                <1>
0001203A 51                     <1>       push   ecx
0001203B E87AF5FFFF             <1>       call   delay_100ms
00012040 59                     <1>       pop    ecx
                                <1>
00012041 49                     <1>         dec     ecx
00012042 75E3                   <1>         jnz     short _cold_ac97c_rst_wait
                                <1>
                                <1> _cold_ac97c_rst_fail:
00012044 F9                     <1>         stc
                                <1> _cold_ac97c_rst_ok:
00012045 C3                     <1>       retn
                                <1>
                                <1> sb16_current_sound_data:
                                <1>       ; 20/08/2017
                                <1>       ; 24/06/2017
                                <1>       ; 22/06/2017
                                <1>       ; get current sound (PCM out) data for graphics
                                <1>       ; (for Sound Blaster 16)
                                <1>       ; ebx = Physical address (on page boundary)
                                <1>       ; ecx = Byte count
                                <1>       ; [audio_buff_size]
                                <1>
                                <1>       ;;mov  edi, [audio_buff_size]
                                <1>       ;mov   edi, [audio_dmabuff_size]
                                <1>       ;mov   esi, [audio_dma_buff]
00012046 39CF                   <1>       cmp    edi, ecx
00012048 7302                   <1>       jnb    short sb16_gcd_0
0001204A 89F9                   <1>       mov    ecx, edi
                                <1> sb16_gcd_0:
                                <1>       ; 20/08/2017
0001204C 803D[D4650100]10       <1>       cmp    byte [audio_bps], 16
00012053 750F                   <1>       jne    short sb16_gcd_1 ; 8 bit DMA channel
00012055 E4C6                   <1>       in     al, 0C6h ; DMA channel 5 count register
00012057 88C2                   <1>       mov    dl, al
00012059 E4C6                   <1>       in     al, 0C6h
0001205B 88C6                   <1>       mov     dh, al
0001205D 0FB7C2                 <1>       movzx  eax, dx
00012060 D1E0                   <1>       shl    eax, 1 ; word count -> byte count
00012062 EB4E                   <1>       jmp    short sb16_gcd_2
```

```
                                <1> sb16_gcd_1:
00012064 E403                   <1>       in     al, 03h ; DMA channel 1 count register
00012066 88C2                   <1>       mov    dl, al
00012068 E403                   <1>       in     al, 03h
0001206A 88C6                   <1>       mov     dh, al
0001206C 0FB7C2                 <1>       movzx  eax, dx
0001206F EB41                   <1>       jmp    short sb16_gcd_2
                                <1> ;sb16_gcd_2:
                                <1> ;     cmp    eax, ecx
                                <1> ;     jnb    short sb16_gcd_3
                                <1> ;     ; remain count < graphics bytes
                                <1> ;     mov    eax, ecx ; fix remain count to data size
                                <1> ;sb16_gcd_3:
                                <1> ;     sub    edi, eax
                                <1> ;     jna    short sb16_gcd_4
                                <1> ;     add    esi, edi ; dma buffer offset
                                <1> ;sb16_gcd_4:
                                <1> ;     mov    edi, ebx ; buffer address (for graphics)
                                <1> ;     mov    [u.r0], ecx
                                <1> ;     rep    movsb
                                <1> ;     retn
                                <1>
                                <1> get_current_sound_data:
                                <1>       ; 24/06/2017
                                <1>       ; 22/06/2017
                                <1>       ; get current sound (PCM out) data for graphics
                                <1>       ;
                                <1>       ; ebx = Physical address (on page boundary)
                                <1>       ; ecx = Byte count
                                <1>       ; [audio_buff_size]
                                <1>
                                <1>       ;mov   edi, [audio_buff_size]
00012071 8B3D[C8650100]         <1>       mov    edi, [audio_dmabuff_size]
00012077 8B35[C4650100]         <1>       mov    esi, [audio_dma_buff]
0001207D 803D[A5650100]02       <1>       cmp    byte [audio_device], 2
00012084 72C0                   <1>       jb     short sb16_current_sound_data ; = 1
00012086 D1EF                   <1>       shr    edi, 1
00012088 39CF                   <1>       cmp    edi, ecx
0001208A 7302                   <1>       jnb    short gcd_0
0001208C 89F9                   <1>       mov    ecx, edi
                                <1> gcd_0:
0001208E 803D[A5650100]03       <1>       cmp    byte [audio_device], 3
00012095 7232                   <1>       jb     short ac97_current_sound_data ; = 2
                                <1>       ; = 3
                                <1> vt8233_current_sound_data:
                                <1>       ; 22/06/2017
                                <1>       ; 21/06/2017
                                <1>       ; get current sound (PCM out) data for graphics
                                <1>       ; (for VT 8233, VT 8237R)
                                <1>       ; ebx = Physical address (on page boundary)
                                <1>       ; ecx = Byte count
                                <1>       ; [audio_buff_size]
                                <1>
                                <1>       ;;mov  edi, [audio_buff_size]
                                <1>       ;mov   edi, [audio_dmabuff_size]
                                <1>       ;mov   esi, [audio_dma_buff]
                                <1>       ;shr   edi, 1
                                <1>       ;cmp   edi, ecx
                                <1>       ;jnb   short vt8233_gcd_1
                                <1>       ;mov   ecx, edi
                                <1> vt8233_gcd_1:
00012097 BA0C000000             <1>       mov    edx, VIA_REG_OFFSET_CURR_COUNT
0001209C E879F5FFFF             <1>       call   ctrl_io_r32
000120A1 89C2                   <1>       mov    edx, eax ; remain count (bits 23-0),
                                <1>                       ; SGD index (bits 31-24)
000120A3 81E200000001           <1>       and    edx, 1000000h ; SGD index (0 = 1st half)
000120A9 7402                   <1>       jz     short vt8233_gcd_2
                                <1>       ; the second half of DMA buffer
000120AB 01FE                   <1>       add    esi, edi
                                <1> vt8233_gcd_2:
000120AD 25FFFFFF00             <1>       and    eax, 0FFFFFFh ; bits 23-0
                                <1> ac97_gcd_2:
                                <1> sb16_gcd_2:
000120B2 39C8                   <1>       cmp    eax, ecx
000120B4 7302                   <1>       jnb    short vt8233_gcd_3
                                <1>       ; remain count < graphics bytes
000120B6 89C8                   <1>       mov    eax, ecx ; fix remain count to data size
                                <1> vt8233_gcd_3:
000120B8 29C7                   <1>       sub    edi, eax
000120BA 7602                   <1>       jna    short vt8233_gcd_4
000120BC 01FE                   <1>       add    esi, edi ; dma buffer offset
                                <1> vt8233_gcd_4:
000120BE 89DF                   <1>       mov    edi, ebx ; buffer address (for graphics)
000120C0 890D[64030300]         <1>       mov    [u.r0], ecx
000120C6 F3A4                   <1>       rep    movsb
                                <1> vt8233_gcd_5:
000120C8 C3                     <1>       retn
                                <1>
                                <1> ac97_current_sound_data:
                                <1>       ; 23/06/2017
                                <1>       ; 22/06/2017
                                <1>       ; get current sound (PCM out) data for graphics
                                <1>       ; (for AC'97, ICH)
                                <1>       ; ebx = Physical address (on page boundary)
                                <1>       ; ecx = Byte count
                                <1>       ; [audio_buff_size]
                                <1>
                                <1>       ;;mov  edi, [audio_buff_size]
                                <1>       ;mov   edi, [audio_dmabuff_size]
                                <1>       ;mov   esi, [audio_dma_buff]
                                <1>       ;shr   edi, 1
                                <1>       ;cmp   edi, ecx
                                <1>       ;jnb   short ac97_gcd_0
```

```
52418                                  <1>       ;mov   ecx, edi
52419                                  <1> ac97_gcd_0:
52420 000120C9 66BA1400                <1>       mov    dx, PO_CIV_REG ; Position In Current Buff Reg
52421 000120CD 660315[DE650100]        <1>       add    dx, [NABMBAR]
52422 000120D4 EC                      <1>       in     al, dx ; current index value
52423 000120D5 A801                    <1>       test   al, 1
52424 000120D7 7402                    <1>       jz     short ac97_gcd_1
52425 000120D9 01FE                    <1>       add    esi, edi
52426                                  <1> ac97_gcd_1:
52427 000120DB 31C0                    <1>       xor    eax, eax
52428 000120DD 66BA1800                <1>       mov    dx, PO_PICB_REG ; Position In Current Buff Reg
52429 000120E1 660315[DE650100]        <1>       add    dx, [NABMBAR]
52430 000120E8 66ED                    <1>       in     ax, dx ; remain dwords
52431 000120EA C1E002                  <1>       shl    eax, 2 ; remain bytes ; 23/06/2017
52432 000120ED EBC3                    <1>       jmp    short ac97_gcd_2
52433                                  <1> ;      cmp    eax, ecx
52434                                  <1> ;      jnb    short ac97_gcd_2
52435                                  <1> ;      ; remain count < graphics bytes
52436                                  <1> ;      mov    eax, ecx ; fix remain count to data size
52437                                  <1> ;ac97_gcd_2:
52438                                  <1> ;      sub    edi, eax
52439                                  <1> ;      jna    short ac97_gcd_3
52440                                  <1> ;      add    esi, edi ; dma buffer offset
52441                                  <1> ;ac97_gcd_3:
52442                                  <1> ;      mov    edi, ebx ; buffer address (for graphics)
52443                                  <1> ;      mov    [u.r0], ecx
52444                                  <1> ;      rep    movsb
52445                                  <1> ;      retn
52446                                  <1>
52447                                  <1> sb16_get_dma_buff_off:
52448                                  <1>       ; 24/06/2017
52449                                  <1>       ; 22/06/2017
52450                                  <1>       ; get current (PCM OUT DMA buffer) pointer
52451                                  <1>       ; (for Sound Blaster 16)
52452                                  <1>
52453                                  <1>       ;mov   ecx, [audio_dmabuff_size]
52454                                  <1>       ;xor   ebx, ebx
52455                                  <1>       ;shr   ecx, 1
52456                                  <1> sb16_gdmabo_0:
52457 000120EF E403                    <1>       in     al, 03h
52458 000120F1 88C2                    <1>       mov    dl, al
52459 000120F3 E403                    <1>       in     al, 03h
52460 000120F5 88C6                    <1>       mov     dh, al
52461 000120F7 0FB7C2                  <1>       movzx  eax, dx
52462 000120FA EB30                    <1>       jmp    short sb16_gdmabo_1
52463                                  <1>
52464                                  <1> get_dma_buffer_offset:
52465                                  <1>       ; 24/06/2017
52466                                  <1>       ; 22/06/2017
52467                                  <1>       ; get current sound (PCM out) data for graphics
52468                                  <1>       ;
52469                                  <1>       ; ebx = Physical address (on page boundary)
52470                                  <1>       ; ecx = Byte count
52471                                  <1>       ; [audio_buff_size]
52472                                  <1>
52473 000120FC 8B0D[C8650100]          <1>       mov    ecx, [audio_dmabuff_size]
52474 00012102 31DB                    <1>       xor    ebx, ebx
52475                                  <1> gdmabo_0:
52476 00012104 803D[A5650100]02        <1>       cmp    byte [audio_device], 2
52477 0001210B 72E2                    <1>       jb     short sb16_get_dma_buff_off
52478 0001210D 742A                    <1>       je     short ac97_get_dma_buff_off
52479                                  <1>
52480                                  <1> vt8233_get_dma_buff_off:
52481                                  <1>       ; 24/06/2017
52482                                  <1>       ; 22/06/2017
52483                                  <1>       ; get current (PCM OUT DMA buffer) pointer
52484                                  <1>       ; (for VT 8233, VT 8237R)
52485                                  <1>
52486                                  <1>       ;mov   ecx, [audio_dmabuff_size]
52487                                  <1>       ;xor   ebx, ebx
52488 0001210F D1E9                    <1>       shr    ecx, 1
52489                                  <1> vt8233_gdmabo_0:
52490 00012111 BA0C000000              <1>       mov    edx, VIA_REG_OFFSET_CURR_COUNT
52491 00012116 E8FFF4FFFF              <1>       call   ctrl_io_r32
52492 0001211B 89C2                    <1>       mov    edx, eax ; remain count (bits 23-0),
52493                                  <1>                       ; SGD index (bits 31-24)
52494 0001211D 81E200000001            <1>       and    edx, 1000000h ; SGD index (0 = 1st half)
52495 00012123 7402                    <1>       jz     short vt8233_gdmabo_1
52496                                  <1>       ; the second half of DMA buffer
52497 00012125 89CB                    <1>       mov    ebx, ecx
52498                                  <1> vt8233_gdmabo_1:
52499 00012127 25FFFFFF00              <1>       and    eax, 0FFFFFFh ; bits 23-0
52500                                  <1> sb16_gdmabo_1:
52501                                  <1> ac97_gdmabo_2:
52502 0001212C 29C1                    <1>       sub    ecx, eax
52503 0001212E 7602                    <1>       jna    short vt8233_gdmabo_2
52504 00012130 01CB                    <1>       add    ebx, ecx ; dma buffer offset
52505                                  <1> vt8233_gdmabo_2:
52506 00012132 891D[64030300]          <1>       mov    [u.r0], ebx
52507 00012138 C3                      <1>       retn
52508                                  <1>
52509                                  <1> ac97_get_dma_buff_off:
52510                                  <1>       ; 24/06/2017
52511                                  <1>       ; 22/06/2017
52512                                  <1>       ; get current (PCM OUT DMA buffer) pointer
52513                                  <1>       ; (for AC'97, ICH)
52514                                  <1>       ; ebx = Physical address (on page boundary)
52515                                  <1>       ; ecx = Byte count
52516                                  <1>       ; [audio_buff_size]
52517                                  <1>
52518                                  <1>       ;mov   ecx, [audio_dmabuff_size]
52519                                  <1>       ;xor   ebx, ebx
52520 00012139 D1E9                    <1>       shr    ecx, 1
```

```
 52521                                  <1> ac97_gdmabo_0:
 52522 0001213B 66BA1400                <1>         mov    dx, PO_CIV_REG ; Position In Current Buff Reg
 52523 0001213F 660315[DE650100]        <1>         add    dx, [NABMBAR]
 52524 00012146 EC                      <1>         in     al, dx ; current index value
 52525 00012147 A801                    <1>         test   al, 1
 52526 00012149 7402                    <1>         jz     short ac97_gdmabo_1
 52527 0001214B 89CB                    <1>         mov    ebx, ecx
 52528                                  <1> ac97_gdmabo_1:
 52529 0001214D 31C0                    <1>         xor    eax, eax
 52530 0001214F 66BA1800                <1>         mov    dx, PO_PICB_REG ; Position In Current Buff Reg
 52531 00012153 660315[DE650100]        <1>         add    dx, [NABMBAR]
 52532 0001215A 66ED                    <1>         in     ax, dx ; remain dwords
 52533 0001215C EBCE                    <1>         jmp    short ac97_gdmabo_2
 52534
 52535 0001215E 90<rept>                    align 4
 52536
 52537                                      %include 'vgadata.s' ; 04/07/2016
 52538                                  <1> ;
*****************************************************************************************
 52539                                  <1> ; TRDOS386.ASM (TRDOS 386 Kernel) - v2.0.0 - vgadata.s  (palette and fond data)
 52540                                  <1> ; ----------------------------------------------------------------------------
 52541                                  <1> ; Last Update: 04/07/2016
 52542                                  <1> ; ----------------------------------------------------------------------------
 52543                                  <1> ; Beginning: 16/01/2016
 52544                                  <1> ; ----------------------------------------------------------------------------
 52545                                  <1> ; Assembler: NASM version 2.11 (trdos386.s)
 52546                                  <1> ; ----------------------------------------------------------------------------
 52547                                  <1> ; Turkish Rational DOS
 52548                                  <1> ; Operating System Project v2.0 by ERDOGAN TAN (Beginning: 04/01/2016)
 52549                                  <1> ;
 52550                                  <1> ; Derived from 'Plex86/Bochs VGABios' source code, vgabios-0.7a (2011)
 52551                                  <1> ; by the LGPL VGABios Developers Team (2001-2008), 'vgatables.h'
 52552                                  <1> ;
 52553                                  <1> ; Oracle VirtualBox 5.0.24 VGABios Source Code
 52554                                  <1> ; ('vgabios.c', 'vgatables.h', 'vgafonts.h', 'vgarom.asm')
 52555                                  <1> ;
 52556                                  <1> ; Palette and font data in assembly language format:
 52557                                  <1> ; 'VBoxVgaBiosAlternative.asm'
 52558                                  <1>
 52559                                  <1> ;
*****************************************************************************************
 52560                                  <1>
 52561                                  <1> ; 04/07/2016
 52562                                  <1> ; COLOR DATA
 52563                                  <1>
 52564                                  <1> palette0:
 52565 00012160 000000000000000000-     <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 52566 00012169 00000000000000          <1>
 52567 00012170 00000000000000002A-     <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah,
02ah
 52568 00012179 2A2A2A2A2A2A2A          <1>
 52569 00012180 2A2A2A2A2A2A2A2A2A-     <1>     db  02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah,
02ah
 52570 00012189 2A2A2A2A2A2A2A          <1>
 52571 00012190 2A2A2A2A2A2A2A2A2A-     <1>     db  02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah,
02ah
 52572 00012199 2A2A2A2A2A2A2A          <1>
 52573 000121A0 2A2A2A2A2A2A2A2A3F-     <1>     db  02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh,
03fh
 52574 000121A9 3F3F3F3F3F3F3F          <1>
 52575 000121B0 3F3F3F3F3F3F3F3F3F-     <1>     db  03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh,
03fh
 52576 000121B9 3F3F3F3F3F3F3F          <1>
 52577 000121C0 000000000000000000-     <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 52578 000121C9 00000000000000          <1>
 52579 000121D0 00000000000000002A-     <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah,
02ah
 52580 000121D9 2A2A2A2A2A2A2A          <1>
 52581 000121E0 2A2A2A2A2A2A2A2A2A-     <1>     db  02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah,
02ah
 52582 000121E9 2A2A2A2A2A2A2A          <1>
 52583 000121F0 2A2A2A2A2A2A2A2A2A-     <1>     db  02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah,
02ah
 52584 000121F9 2A2A2A2A2A2A2A          <1>
 52585 00012200 2A2A2A2A2A2A2A2A3F-     <1>     db  02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh,
03fh
 52586 00012209 3F3F3F3F3F3F3F          <1>
 52587 00012210 3F3F3F3F3F3F3F3F3F-     <1>     db  03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh,
03fh
 52588 00012219 3F3F3F3F3F3F3F          <1>
 52589                                  <1> palette1:
 52590 00012220 00000000002A002A00-     <1>     db  000h, 000h, 000h, 000h, 000h, 02ah, 000h, 02ah, 000h, 000h, 02ah, 02ah, 02ah, 000h, 000h,
02ah
 52591 00012229 002A2A2A00002A          <1>
 52592 00012230 002A2A15002A2A2A00-     <1>     db  000h, 02ah, 02ah, 015h, 000h, 02ah, 02ah, 02ah, 000h, 000h, 000h, 000h, 000h, 02ah, 000h,
02ah
 52593 00012239 000000002A002A          <1>
 52594 00012240 00002A2A2A00002A00-     <1>     db  000h, 000h, 02ah, 02ah, 02ah, 000h, 000h, 02ah, 000h, 02ah, 02ah, 015h, 000h, 02ah, 02ah,
02ah
 52595 00012249 2A2A15002A2A2A          <1>
 52596 00012250 15151515153F153F15-     <1>     db  015h, 015h, 015h, 015h, 015h, 03fh, 015h, 03fh, 015h, 015h, 03fh, 03fh, 03fh, 015h, 015h,
03fh
 52597 00012259 153F3F3F15153F          <1>
 52598 00012260 153F3F3F153F3F3F15-     <1>     db  015h, 03fh, 03fh, 03fh, 015h, 03fh, 03fh, 03fh, 015h, 015h, 015h, 015h, 015h, 03fh, 015h,
03fh
 52599 00012269 151515153F153F          <1>
 52600 00012270 15153F3F3F15153F15-     <1>     db  015h, 015h, 03fh, 03fh, 03fh, 015h, 015h, 03fh, 015h, 03fh, 03fh, 03fh, 015h, 03fh, 03fh,
03fh
 52601 00012279 3F3F3F153F3F3F          <1>
 52602 00012280 00000000002A002A00-     <1>     db  000h, 000h, 000h, 000h, 000h, 02ah, 000h, 02ah, 000h, 000h, 02ah, 02ah, 02ah, 000h, 000h,
02ah
```

```
 52603 00012289 002A2A2A00002A      <1>
 52604 00012290 002A2A15002A2A2A00- <1>     db  000h, 02ah, 02ah, 015h, 000h, 02ah, 02ah, 02ah, 000h, 000h, 000h, 000h, 000h, 02ah, 000h,
02ah
 52605 00012299 000000002A002A      <1>
 52606 000122A0 00002A2A2A00002A00- <1>     db  000h, 000h, 02ah, 02ah, 02ah, 000h, 000h, 02ah, 000h, 02ah, 02ah, 015h, 000h, 02ah, 02ah,
02ah
 52607 000122A9 2A2A15002A2A2A      <1>
 52608 000122B0 15151515153F153F15- <1>     db  015h, 015h, 015h, 015h, 015h, 03fh, 015h, 03fh, 015h, 015h, 03fh, 03fh, 03fh, 015h, 015h,
03fh
 52609 000122B9 153F3F3F15153F      <1>
 52610 000122C0 153F3F3F153F3F3F15- <1>     db  015h, 03fh, 03fh, 03fh, 015h, 03fh, 03fh, 03fh, 015h, 015h, 015h, 015h, 015h, 03fh, 015h,
03fh
 52611 000122C9 151515153F153F      <1>
 52612 000122D0 15153F3F3F15153F15- <1>     db  015h, 015h, 03fh, 03fh, 03fh, 015h, 015h, 03fh, 015h, 03fh, 03fh, 03fh, 015h, 03fh, 03fh,
03fh
 52613 000122D9 3F3F3F153F3F3F      <1>
 52614                              <1> palette2:
 52615 000122E0 00000000002A002A00- <1>     db  000h, 000h, 000h, 000h, 000h, 02ah, 000h, 02ah, 000h, 000h, 02ah, 02ah, 02ah, 000h, 000h,
02ah
 52616 000122E9 002A2A2A00002A      <1>
 52617 000122F0 002A2A2A002A2A2A00- <1>     db  000h, 02ah, 02ah, 02ah, 000h, 02ah, 02ah, 02ah, 000h, 000h, 015h, 000h, 000h, 03fh, 000h,
02ah
 52618 000122F9 001500003F002A      <1>
 52619 00012300 15002A3F2A00152A00- <1>     db  015h, 000h, 02ah, 03fh, 02ah, 000h, 015h, 02ah, 000h, 03fh, 02ah, 02ah, 015h, 02ah, 02ah,
03fh
 52620 00012309 3F2A2A152A2A3F      <1>
 52621 00012310 00150000152A003F00- <1>     db  000h, 015h, 000h, 000h, 015h, 02ah, 000h, 03fh, 000h, 000h, 03fh, 02ah, 02ah, 015h, 000h,
02ah
 52622 00012319 003F2A2A15002A      <1>
 52623 00012320 152A2A3F002A3F2A00- <1>     db  015h, 02ah, 02ah, 03fh, 000h, 02ah, 03fh, 02ah, 000h, 015h, 015h, 000h, 015h, 03fh, 000h,
03fh
 52624 00012329 151500153F003F      <1>
 52625 00012330 15003F3F2A15152A15- <1>     db  015h, 000h, 03fh, 03fh, 02ah, 015h, 015h, 02ah, 015h, 03fh, 02ah, 03fh, 015h, 02ah, 03fh,
03fh
 52626 00012339 3F2A3F152A3F3F      <1>
 52627 00012340 15000015002A152A00- <1>     db  015h, 000h, 000h, 015h, 000h, 02ah, 015h, 02ah, 000h, 015h, 02ah, 02ah, 03fh, 000h, 000h,
03fh
 52628 00012349 152A2A3F00003F      <1>
 52629 00012350 002A3F2A003F2A2A15- <1>     db  000h, 02ah, 03fh, 02ah, 000h, 03fh, 02ah, 02ah, 015h, 000h, 015h, 015h, 000h, 03fh, 015h,
02ah
 52630 00012359 001515003F152A      <1>
 52631 00012360 15152A3F3F00153F00- <1>     db  015h, 015h, 02ah, 03fh, 03fh, 000h, 015h, 03fh, 000h, 03fh, 03fh, 02ah, 015h, 03fh, 02ah,
03fh
 52632 00012369 3F3F2A153F2A3F      <1>
 52633 00012370 15150015152A153F00- <1>     db  015h, 015h, 000h, 015h, 015h, 02ah, 015h, 03fh, 000h, 015h, 03fh, 02ah, 03fh, 015h, 000h,
03fh
 52634 00012379 153F2A3F15003F      <1>
 52635 00012380 152A3F3F003F3F2A15- <1>     db  015h, 02ah, 03fh, 03fh, 000h, 03fh, 03fh, 02ah, 015h, 015h, 015h, 015h, 015h, 03fh, 015h,
03fh
 52636 00012389 151515153F153F      <1>
 52637 00012390 15153F3F3F15153F15- <1>     db  015h, 015h, 03fh, 03fh, 03fh, 015h, 015h, 03fh, 015h, 03fh, 03fh, 03fh, 015h, 03fh, 03fh,
03fh
 52638 00012399 3F3F3F153F3F3F      <1>
 52639                              <1> palette3:
 52640 000123A0 00000000002A002A00- <1>     db  000h, 000h, 000h, 000h, 000h, 02ah, 000h, 02ah, 000h, 000h, 02ah, 02ah, 02ah, 000h, 000h,
02ah
 52641 000123A9 002A2A2A00002A      <1>
 52642 000123B0 002A2A15002A2A2A15- <1>     db  000h, 02ah, 02ah, 015h, 000h, 02ah, 02ah, 02ah, 015h, 015h, 015h, 015h, 015h, 03fh, 015h,
03fh
 52643 000123B9 151515153F153F      <1>
 52644 000123C0 15153F3F3F15153F15- <1>     db  015h, 015h, 03fh, 03fh, 03fh, 015h, 015h, 03fh, 015h, 03fh, 03fh, 03fh, 015h, 03fh, 03fh,
03fh
 52645 000123C9 3F3F3F153F3F3F      <1>
 52646 000123D0 000000050505080808- <1>     db  000h, 000h, 000h, 005h, 005h, 005h, 008h, 008h, 008h, 00bh, 00bh, 00bh, 00eh, 00eh, 00eh,
011h
 52647 000123D9 0B0B0B0E0E0E11      <1>
 52648 000123E0 111114141418181C- <1>     db  011h, 011h, 014h, 014h, 014h, 018h, 018h, 018h, 01ch, 01ch, 01ch, 020h, 020h, 020h, 024h,
024h
 52649 000123E9 1C1C2020202424      <1>
 52650 000123F0 242828282D2D2D3232- <1>     db  024h, 028h, 028h, 028h, 02dh, 02dh, 02dh, 032h, 032h, 032h, 038h, 038h, 038h, 03fh, 03fh,
03fh
 52651 000123F9 323838383F3F3F      <1>
 52652 00012400 00003F10003F1F003F- <1>     db  000h, 000h, 03fh, 010h, 000h, 03fh, 01fh, 000h, 03fh, 02fh, 000h, 03fh, 03fh, 000h, 03fh,
03fh
 52653 00012409 2F003F3F003F3F      <1>
 52654 00012410 002F3F001F3F00103F- <1>     db  000h, 02fh, 03fh, 000h, 01fh, 03fh, 000h, 010h, 03fh, 000h, 000h, 03fh, 010h, 000h, 03fh,
01fh
 52655 00012419 00003F10003F1F      <1>
 52656 00012420 003F2F003F3F002F3F- <1>     db  000h, 03fh, 02fh, 000h, 03fh, 03fh, 000h, 02fh, 03fh, 000h, 01fh, 03fh, 000h, 010h, 03fh,
000h
 52657 00012429 001F3F00103F00      <1>
 52658 00012430 003F00003F10003F1F- <1>     db  000h, 03fh, 000h, 000h, 03fh, 010h, 000h, 03fh, 01fh, 000h, 03fh, 02fh, 000h, 03fh, 03fh,
000h
 52659 00012439 003F2F003F3F00      <1>
 52660 00012440 2F3F001F3F00103F1F- <1>     db  02fh, 03fh, 000h, 01fh, 03fh, 000h, 010h, 03fh, 01fh, 01fh, 03fh, 027h, 01fh, 03fh, 02fh,
01fh
 52661 00012449 1F3F271F3F2F1F      <1>
 52662 00012450 3F371F3F3F1F3F3F1F- <1>     db  03fh, 037h, 01fh, 03fh, 03fh, 01fh, 03fh, 03fh, 01fh, 037h, 03fh, 01fh, 02fh, 03fh, 01fh,
027h
 52663 00012459 373F1F2F3F1F27      <1>
 52664 00012460 3F1F1F3F271F3F2F1F- <1>     db  03fh, 01fh, 01fh, 03fh, 027h, 01fh, 03fh, 02fh, 01fh, 03fh, 037h, 01fh, 03fh, 03fh, 01fh,
037h
 52665 00012469 3F371F3F3F1F37      <1>
 52666 00012470 3F1F2F3F1F273F1F1F- <1>     db  03fh, 01fh, 02fh, 03fh, 01fh, 027h, 03fh, 01fh, 01fh, 03fh, 01fh, 01fh, 03fh, 027h, 01fh,
03fh
 52667 00012479 3F1F1F3F271F3F      <1>
 52668 00012480 2F1F3F371F3F3F1F37- <1>     db  02fh, 01fh, 03fh, 037h, 01fh, 03fh, 03fh, 01fh, 037h, 03fh, 01fh, 02fh, 03fh, 01fh, 027h,
03fh
 52669 00012489 3F1F2F3F1F273F      <1>
 52670 00012490 2D2D3F312D3F362D3F- <1>     db  02dh, 02dh, 03fh, 031h, 02dh, 03fh, 036h, 02dh, 03fh, 03ah, 02dh, 03fh, 03fh, 02dh, 03fh,
03fh
 52671 00012499 3A2D3F3F2D3F3F      <1>
```

```
 52672 000124A0 2D3A3F2D363F2D313F- <1>     db  02dh, 03ah, 03fh, 02dh, 036h, 03fh, 02dh, 031h, 03fh, 02dh, 02dh, 03fh, 031h, 02dh, 03fh,
036h
 52673 000124A9 2D2D3F312D3F36      <1>
 52674 000124B0 2D3F3A2D3F3F2D3A3F- <1>     db  02dh, 03fh, 03ah, 02dh, 03fh, 03fh, 02dh, 03ah, 03fh, 02dh, 036h, 03fh, 02dh, 031h, 03fh,
02dh
 52675 000124B9 2D363F2D313F2D      <1>
 52676 000124C0 2D3F2D2D3F312D3F36- <1>     db  02dh, 03fh, 02dh, 02dh, 03fh, 031h, 02dh, 03fh, 036h, 02dh, 03fh, 03ah, 02dh, 03fh, 03fh,
02dh
 52677 000124C9 2D3F3A2D3F3F2D      <1>
 52678 000124D0 3A3F2D363F2D313F00- <1>     db  03ah, 03fh, 02dh, 036h, 03fh, 02dh, 031h, 03fh, 000h, 000h, 01ch, 007h, 000h, 01ch, 00eh,
000h
 52679 000124D9 001C07001C0E00      <1>
 52680 000124E0 1C15001C1C001C1C00- <1>     db  01ch, 015h, 000h, 01ch, 01ch, 000h, 01ch, 01ch, 000h, 015h, 01ch, 000h, 00eh, 01ch, 000h,
007h
 52681 000124E9 151C000E1C0007      <1>
 52682 000124F0 1C00001C07001C0E00- <1>     db  01ch, 000h, 000h, 01ch, 007h, 000h, 01ch, 00eh, 000h, 01ch, 015h, 000h, 01ch, 01ch, 000h,
015h
 52683 000124F9 1C15001C1C0015      <1>
 52684 00012500 1C000E1C00071C0000- <1>     db  01ch, 000h, 00eh, 01ch, 000h, 007h, 01ch, 000h, 000h, 01ch, 000h, 000h, 01ch, 007h, 000h,
01ch
 52685 00012509 1C00001C07001C      <1>
 52686 00012510 0E001C15001C1C0015- <1>     db  00eh, 000h, 01ch, 015h, 000h, 01ch, 01ch, 000h, 015h, 01ch, 000h, 00eh, 01ch, 000h, 007h,
01ch
 52687 00012519 1C000E1C00071C      <1>
 52688 00012520 0E0E1C110E1C150E1C- <1>     db  00eh, 00eh, 01ch, 011h, 00eh, 01ch, 015h, 00eh, 01ch, 018h, 00eh, 01ch, 01ch, 00eh, 01ch,
01ch
 52689 00012529 180E1C1C0E1C1C      <1>
 52690 00012530 0E181C0E151C0E111C- <1>     db  00eh, 018h, 01ch, 00eh, 015h, 01ch, 00eh, 011h, 01ch, 00eh, 00eh, 01ch, 011h, 00eh, 01ch,
015h
 52691 00012539 0E0E1C110E1C15      <1>
 52692 00012540 0E1C180E1C1C0E181C- <1>     db  00eh, 01ch, 018h, 00eh, 01ch, 01ch, 00eh, 018h, 01ch, 00eh, 015h, 01ch, 00eh, 011h, 01ch,
00eh
 52693 00012549 0E151C0E111C0E      <1>
 52694 00012550 0E1C0E0E1C110E1C15- <1>     db  00eh, 01ch, 00eh, 00eh, 01ch, 011h, 00eh, 01ch, 015h, 00eh, 01ch, 018h, 00eh, 01ch, 01ch,
00eh
 52695 00012559 0E1C180E1C1C0E      <1>
 52696 00012560 181C0E151C0E111C14- <1>     db  018h, 01ch, 00eh, 015h, 01ch, 00eh, 011h, 01ch, 014h, 014h, 01ch, 016h, 014h, 01ch, 018h,
014h
 52697 00012569 141C16141C1814      <1>
 52698 00012570 1C1A141C1C141C1C14- <1>     db  01ch, 01ah, 014h, 01ch, 01ch, 014h, 01ch, 01ch, 014h, 01ah, 01ch, 014h, 018h, 01ch, 014h,
016h
 52699 00012579 1A1C14181C1416      <1>
 52700 00012580 1C14141C16141C1814- <1>     db  01ch, 014h, 014h, 01ch, 016h, 014h, 01ch, 018h, 014h, 01ch, 01ah, 014h, 01ch, 01ch, 014h,
01ah
 52701 00012589 1C1A141C1C141A      <1>
 52702 00012590 1C14181C14161C1414- <1>     db  01ch, 014h, 018h, 01ch, 014h, 016h, 01ch, 014h, 014h, 01ch, 014h, 014h, 01ch, 016h, 014h,
01ch
 52703 00012599 1C14141C16141C      <1>
 52704 000125A0 18141C1A141C1C141A- <1>     db  018h, 014h, 01ch, 01ah, 014h, 01ch, 01ch, 014h, 01ah, 01ch, 014h, 018h, 01ch, 014h, 016h,
01ch
 52705 000125A9 1C14181C14161C      <1>
 52706 000125B0 0000100400100800 10- <1>     db  000h, 000h, 010h, 004h, 000h, 010h, 008h, 000h, 010h, 00ch, 000h, 010h, 010h, 000h, 010h,
010h
 52707 000125B9 0C001010001010      <1>
 52708 000125C0 000C10000810000410- <1>     db  000h, 00ch, 010h, 000h, 008h, 010h, 000h, 004h, 010h, 000h, 000h, 010h, 004h, 000h, 010h,
008h
 52709 000125C9 00001004001008      <1>
 52710 000125D0 00100C001010000C10- <1>     db  000h, 010h, 00ch, 000h, 010h, 010h, 000h, 00ch, 010h, 000h, 008h, 010h, 000h, 004h, 010h,
000h
 52711 000125D9 00081000041000      <1>
 52712 000125E0 0010000010040010 08- <1>     db  000h, 010h, 000h, 000h, 010h, 004h, 000h, 010h, 008h, 000h, 010h, 00ch, 000h, 010h, 010h,
000h
 52713 000125E9 00100C00101000      <1>
 52714 000125F0 0C1000081000041008- <1>     db  00ch, 010h, 000h, 008h, 010h, 000h, 004h, 010h, 008h, 008h, 010h, 00ah, 008h, 010h, 00ch,
008h
 52715 000125F9 08100A08100C08      <1>
 52716 00012600 100E08101008101008- <1>     db  010h, 00eh, 008h, 010h, 010h, 008h, 010h, 010h, 008h, 00eh, 010h, 008h, 00ch, 010h, 008h,
00ah
 52717 00012609 0E10080C10080A      <1>
 52718 00012610 100808100A08100C08- <1>     db  010h, 008h, 008h, 010h, 00ah, 008h, 010h, 00ch, 008h, 010h, 00eh, 008h, 010h, 010h, 008h,
00eh
 52719 00012619 100E081010080E      <1>
 52720 00012620 10080C10080A100808- <1>     db  010h, 008h, 00ch, 010h, 008h, 00ah, 010h, 008h, 008h, 010h, 008h, 008h, 010h, 00ah, 008h,
010h
 52721 00012629 100808100A0810      <1>
 52722 00012630 0C08100E081010080E- <1>     db  00ch, 008h, 010h, 00eh, 008h, 010h, 010h, 008h, 00eh, 010h, 008h, 00ch, 010h, 008h, 00ah,
010h
 52723 00012639 10080C10080A10      <1>
 52724 00012640 0B0B100C0B100D0B10- <1>     db  00bh, 00bh, 010h, 00ch, 00bh, 010h, 00dh, 00bh, 010h, 00fh, 00bh, 010h, 010h, 00bh, 010h,
010h
 52725 00012649 0F0B10100B1010      <1>
 52726 00012650 0B0F100B0D100B0C10- <1>     db  00bh, 00fh, 010h, 00bh, 00dh, 010h, 00bh, 00ch, 010h, 00bh, 00bh, 010h, 00ch, 00bh, 010h,
00dh
 52727 00012659 0B0B100C0B100D      <1>
 52728 00012660 0B100F0B10100B0F10- <1>     db  00bh, 010h, 00fh, 00bh, 010h, 010h, 00bh, 00fh, 010h, 00bh, 00dh, 010h, 00bh, 00ch, 010h,
00bh
 52729 00012669 0B0D100B0C100B      <1>
 52730 00012670 0B100B0B100C0B100D- <1>     db  00bh, 010h, 00bh, 00bh, 010h, 00ch, 00bh, 010h, 00dh, 00bh, 010h, 00fh, 00bh, 010h, 010h,
00bh
 52731 00012679 0B100F0B10100B      <1>
 52732 00012680 0F100B0D100B0C1000- <1>     db  00fh, 010h, 00bh, 00dh, 010h, 00bh, 00ch, 010h, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 52733 00012689 00000000000000      <1>
 52734 00012690 0000000000000000 00- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 52735 00012699 00000000000000      <1>
 52736                              <1>
 52737                              <1>
 52738                              <1> ; 04/07/2016
 52739                              <1> ; FONT DATA
 52740                              <1>
 52741                              <1> CRT_CHAR_GEN:
 52742                              <1> vgafont8:
```

```
 52743 000126A0 00000000000000007E- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 07eh, 081h, 0a5h, 081h, 0bdh, 099h, 081h,
07eh
 52744 000126A9 81A581BD99817E      <1>
 52745 000126B0 7EFFDBFFC3E7FF7E6C- <1>     db  07eh, 0ffh, 0dbh, 0ffh, 0c3h, 0e7h, 0ffh, 07eh, 06ch, 0feh, 0feh, 0feh, 07ch, 038h, 010h,
000h
 52746 000126B9 FEFEFE7C381000      <1>
 52747 000126C0 10387CFE7C38100038- <1>     db  010h, 038h, 07ch, 0feh, 07ch, 038h, 010h, 000h, 038h, 07ch, 038h, 0feh, 0feh, 07ch, 038h,
07ch
 52748 000126C9 7C38FEFE7C387C      <1>
 52749 000126D0 1010387CFE7C387C00- <1>     db  010h, 010h, 038h, 07ch, 0feh, 07ch, 038h, 07ch, 000h, 000h, 018h, 03ch, 03ch, 018h, 000h,
000h
 52750 000126D9 00183C3C180000      <1>
 52751 000126E0 FFFFE7C3C3E7FFFF00- <1>     db  0ffh, 0ffh, 0e7h, 0c3h, 0c3h, 0e7h, 0ffh, 0ffh, 000h, 03ch, 066h, 042h, 042h, 066h, 03ch,
000h
 52752 000126E9 3C664242663C00      <1>
 52753 000126F0 FFC399BDBD99C3FF0F- <1>     db  0ffh, 0c3h, 099h, 0bdh, 0bdh, 099h, 0c3h, 0ffh, 00fh, 007h, 00fh, 07dh, 0cch, 0cch, 0cch,
078h
 52754 000126F9 070F7DCCCCCC78      <1>
 52755 00012700 3C6666663C187E183F- <1>     db  03ch, 066h, 066h, 066h, 03ch, 018h, 07eh, 018h, 03fh, 033h, 03fh, 030h, 030h, 070h, 0f0h,
0e0h
 52756 00012709 333F303070F0E0      <1>
 52757 00012710 7F637F636367E6C099- <1>     db  07fh, 063h, 07fh, 063h, 063h, 067h, 0e6h, 0c0h, 099h, 05ah, 03ch, 0e7h, 0e7h, 03ch, 05ah,
099h
 52758 00012719 5A3CE7E73C5A99      <1>
 52759 00012720 80E0F8FEF8E0800002- <1>     db  080h, 0e0h, 0f8h, 0feh, 0f8h, 0e0h, 080h, 000h, 002h, 00eh, 03eh, 0feh, 03eh, 00eh, 002h,
000h
 52760 00012729 0E3EFE3E0E0200      <1>
 52761 00012730 183C7E18187E3C1866- <1>     db  018h, 03ch, 07eh, 018h, 018h, 07eh, 03ch, 018h, 066h, 066h, 066h, 066h, 066h, 000h, 066h,
000h
 52762 00012739 66666666006600      <1>
 52763 00012740 7FDBDB7B1B1B1B003E- <1>     db  07fh, 0dbh, 0dbh, 07bh, 01bh, 01bh, 01bh, 000h, 03eh, 063h, 038h, 06ch, 06ch, 038h, 0cch,
078h
 52764 00012749 63386C6C38CC78      <1>
 52765 00012750 000000007E7E7E0018- <1>     db  000h, 000h, 000h, 000h, 07eh, 07eh, 07eh, 000h, 018h, 03ch, 07eh, 018h, 07eh, 03ch, 018h,
0ffh
 52766 00012759 3C7E187E3C18FF      <1>
 52767 00012760 183C7E181818180018- <1>     db  018h, 03ch, 07eh, 018h, 018h, 018h, 018h, 000h, 018h, 018h, 018h, 018h, 07eh, 03ch, 018h,
000h
 52768 00012769 1818187E3C1800      <1>
 52769 00012770 00180CFE0C18000000- <1>     db  000h, 018h, 00ch, 0feh, 00ch, 018h, 000h, 000h, 000h, 030h, 060h, 0feh, 060h, 030h, 000h,
000h
 52770 00012779 3060FE60300000      <1>
 52771 00012780 0000C0C0C0FE000000- <1>     db  000h, 000h, 0c0h, 0c0h, 0c0h, 0feh, 000h, 000h, 000h, 024h, 066h, 0ffh, 066h, 024h, 000h,
000h
 52772 00012789 2466FF66240000      <1>
 52773 00012790 00183C7EFFFF000000- <1>     db  000h, 018h, 03ch, 07eh, 0ffh, 0ffh, 000h, 000h, 000h, 0ffh, 0ffh, 07eh, 03ch, 018h, 000h,
000h
 52774 00012799 FFFF7E3C180000      <1>
 52775 000127A0 0000000000000030- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 030h, 078h, 078h, 030h, 030h, 000h, 030h,
000h
 52776 000127A9 78783030003000      <1>
 52777 000127B0 6C6C6C000000006C- <1>     db  06ch, 06ch, 06ch, 000h, 000h, 000h, 000h, 000h, 06ch, 06ch, 0feh, 06ch, 0feh, 06ch, 06ch,
000h
 52778 000127B9 6CFE6CFE6C6C00      <1>
 52779 000127C0 307CC0780CF8300000- <1>     db  030h, 07ch, 0c0h, 078h, 00ch, 0f8h, 030h, 000h, 000h, 0c6h, 0cch, 018h, 030h, 066h, 0c6h,
000h
 52780 000127C9 C6CC183066C600      <1>
 52781 000127D0 386C3876DCCC760060- <1>     db  038h, 06ch, 038h, 076h, 0dch, 0cch, 076h, 000h, 060h, 060h, 0c0h, 000h, 000h, 000h, 000h,
000h
 52782 000127D9 60C00000000000      <1>
 52783 000127E0 183060606030180060- <1>     db  018h, 030h, 060h, 060h, 060h, 030h, 018h, 000h, 060h, 030h, 018h, 018h, 018h, 030h, 060h,
000h
 52784 000127E9 30181818306000      <1>
 52785 000127F0 00663CFF3C66000000- <1>     db  000h, 066h, 03ch, 0ffh, 03ch, 066h, 000h, 000h, 000h, 030h, 030h, 0fch, 030h, 030h, 000h,
000h
 52786 000127F9 3030FC30300000      <1>
 52787 00012800 0000000000306000- <1>     db  000h, 000h, 000h, 000h, 000h, 030h, 030h, 060h, 000h, 000h, 000h, 0fch, 000h, 000h, 000h,
000h
 52788 00012809 0000FC00000000      <1>
 52789 00012810 0000000000306000- <1>     db  000h, 000h, 000h, 000h, 000h, 030h, 030h, 000h, 006h, 00ch, 018h, 030h, 060h, 0c0h, 080h,
000h
 52790 00012819 0C183060C08000      <1>
 52791 00012820 7CC6CEDEF6E67C0030- <1>     db  07ch, 0c6h, 0ceh, 0deh, 0f6h, 0e6h, 07ch, 000h, 030h, 070h, 030h, 030h, 030h, 030h, 0fch,
000h
 52792 00012829 7030303030FC00      <1>
 52793 00012830 78CC0C3860CCFC0078- <1>     db  078h, 0cch, 00ch, 038h, 060h, 0cch, 0fch, 000h, 078h, 0cch, 00ch, 038h, 00ch, 0cch, 078h,
000h
 52794 00012839 CC0C380CCC7800      <1>
 52795 00012840 1C3C6CCCFE0C1E00FC- <1>     db  01ch, 03ch, 06ch, 0cch, 0feh, 00ch, 01eh, 000h, 0fch, 0c0h, 0f8h, 00ch, 00ch, 0cch, 078h,
000h
 52796 00012849 C0F80C0CCC7800      <1>
 52797 00012850 3860C0F8CCCC7800FC- <1>     db  038h, 060h, 0c0h, 0f8h, 0cch, 0cch, 078h, 000h, 0fch, 0cch, 00ch, 018h, 030h, 030h, 030h,
000h
 52798 00012859 CC0C1830303000      <1>
 52799 00012860 78CCCC78CCCC780078- <1>     db  078h, 0cch, 0cch, 078h, 0cch, 0cch, 078h, 000h, 078h, 0cch, 0cch, 07ch, 00ch, 018h, 070h,
000h
 52800 00012869 CCCC7C0C187000      <1>
 52801 00012870 0030300000030300000- <1>     db  000h, 030h, 030h, 000h, 000h, 030h, 030h, 000h, 000h, 030h, 030h, 000h, 000h, 030h, 030h,
060h
 52802 00012879 30300000303060      <1>
 52803 00012880 183060C06030180000- <1>     db  018h, 030h, 060h, 0c0h, 060h, 030h, 018h, 000h, 000h, 000h, 0fch, 000h, 000h, 0fch, 000h,
000h
 52804 00012889 00FC0000FC0000      <1>
 52805 00012890 6030180C1830600078- <1>     db  060h, 030h, 018h, 00ch, 018h, 030h, 060h, 000h, 078h, 0cch, 00ch, 018h, 030h, 000h, 030h,
000h
 52806 00012899 CC0C1830003000      <1>
 52807 000128A0 7CC6DEDEDEC0780030- <1>     db  07ch, 0c6h, 0deh, 0deh, 0deh, 0c0h, 078h, 000h, 030h, 078h, 0cch, 0cch, 0fch, 0cch, 0cch,
000h
 52808 000128A9 78CCCCFCCCCC00      <1>
 52809 000128B0 FC66667C6666FC003C- <1>     db  0fch, 066h, 066h, 07ch, 066h, 066h, 0fch, 000h, 03ch, 066h, 0c0h, 0c0h, 0c0h, 066h, 03ch,
000h
 52810 000128B9 66C0C0C0663C00      <1>
```

```
 52811 000128C0 F86C6666666CF800FE- <1>     db  0f8h, 06ch, 066h, 066h, 066h, 06ch, 0f8h, 000h, 0feh, 062h, 068h, 078h, 068h, 062h, 0feh,
000h
 52812 000128C9 6268786862FE00      <1>
 52813 000128D0 FE6268786860F0003C- <1>     db  0feh, 062h, 068h, 078h, 068h, 060h, 0f0h, 000h, 03ch, 066h, 0c0h, 0c0h, 0ceh, 066h, 03eh,
000h
 52814 000128D9 66C0C0CE663E00      <1>
 52815 000128E0 CCCCCCFCCCCCCC0078- <1>     db  0cch, 0cch, 0cch, 0fch, 0cch, 0cch, 0cch, 000h, 078h, 030h, 030h, 030h, 030h, 030h, 078h,
000h
 52816 000128E9 30303030307800      <1>
 52817 000128F0 1E0C0C0CCCCC7800E6- <1>     db  01eh, 00ch, 00ch, 00ch, 0cch, 0cch, 078h, 000h, 0e6h, 066h, 06ch, 078h, 06ch, 066h, 0e6h,
000h
 52818 000128F9 666C786C66E600      <1>
 52819 00012900 F06060606266FE00C6- <1>     db  0f0h, 060h, 060h, 060h, 062h, 066h, 0feh, 000h, 0c6h, 0eeh, 0feh, 0feh, 0d6h, 0c6h, 0c6h,
000h
 52820 00012909 EEFEFED6C6C600      <1>
 52821 00012910 C6E6F6DECEC6C60038- <1>     db  0c6h, 0e6h, 0f6h, 0deh, 0ceh, 0c6h, 0c6h, 000h, 038h, 06ch, 0c6h, 0c6h, 0c6h, 06ch, 038h,
000h
 52822 00012919 6CC6C6C66C3800      <1>
 52823 00012920 FC66667C6060F00078- <1>     db  0fch, 066h, 066h, 07ch, 060h, 060h, 0f0h, 000h, 078h, 0cch, 0cch, 0cch, 0dch, 078h, 01ch,
000h
 52824 00012929 CCCCCCDC781C00      <1>
 52825 00012930 FC66667C6C66E60078- <1>     db  0fch, 066h, 066h, 07ch, 06ch, 066h, 0e6h, 000h, 078h, 0cch, 0e0h, 070h, 01ch, 0cch, 078h,
000h
 52826 00012939 CCE0701CCC7800      <1>
 52827 00012940 FCB4303030307800CC- <1>     db  0fch, 0b4h, 030h, 030h, 030h, 030h, 078h, 000h, 0cch, 0cch, 0cch, 0cch, 0cch, 0cch, 0fch,
000h
 52828 00012949 CCCCCCCCCCFC00      <1>
 52829 00012950 CCCCCCCCCC783000C6- <1>     db  0cch, 0cch, 0cch, 0cch, 0cch, 078h, 030h, 000h, 0c6h, 0c6h, 0c6h, 0d6h, 0feh, 0eeh, 0c6h,
000h
 52830 00012959 C6C6D6FEEEC600      <1>
 52831 00012960 C6C66C38386CC600CC- <1>     db  0c6h, 0c6h, 06ch, 038h, 038h, 06ch, 0c6h, 000h, 0cch, 0cch, 0cch, 078h, 030h, 030h, 078h,
000h
 52832 00012969 CCCC7830307800      <1>
 52833 00012970 FEC68C183266FE0078- <1>     db  0feh, 0c6h, 08ch, 018h, 032h, 066h, 0feh, 000h, 078h, 060h, 060h, 060h, 060h, 060h, 078h,
000h
 52834 00012979 60606060607800      <1>
 52835 00012980 C06030180C06020078- <1>     db  0c0h, 060h, 030h, 018h, 00ch, 006h, 002h, 000h, 078h, 018h, 018h, 018h, 018h, 018h, 078h,
000h
 52836 00012989 18181818187800      <1>
 52837 00012990 10386CC60000000000- <1>     db  010h, 038h, 06ch, 0c6h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
0ffh
 52838 00012999 000000000000FF      <1>
 52839 000129A0 303018000000000000- <1>     db  030h, 030h, 018h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 078h, 00ch, 07ch, 0cch, 076h,
000h
 52840 000129A9 00780C7CCC7600      <1>
 52841 000129B0 E060607C6666DC0000- <1>     db  0e0h, 060h, 060h, 07ch, 066h, 066h, 0dch, 000h, 000h, 000h, 078h, 0cch, 0c0h, 0cch, 078h,
000h
 52842 000129B9 0078CCC0CC7800      <1>
 52843 000129C0 1C0C0C7CCCCC760000- <1>     db  01ch, 00ch, 00ch, 07ch, 0cch, 0cch, 076h, 000h, 000h, 000h, 078h, 0cch, 0fch, 0c0h, 078h,
000h
 52844 000129C9 0078CCFCC07800      <1>
 52845 000129D0 386C60F06060F00000- <1>     db  038h, 06ch, 060h, 0f0h, 060h, 060h, 0f0h, 000h, 000h, 000h, 076h, 0cch, 0cch, 07ch, 00ch,
0f8h
 52846 000129D9 0076CCCC7C0CF8      <1>
 52847 000129E0 E0606C766666E60030- <1>     db  0e0h, 060h, 06ch, 076h, 066h, 066h, 0e6h, 000h, 030h, 000h, 070h, 030h, 030h, 030h, 078h,
000h
 52848 000129E9 00703030307800      <1>
 52849 000129F0 0C000C0C0CCCCC78E0- <1>     db  00ch, 000h, 00ch, 00ch, 00ch, 0cch, 0cch, 078h, 0e0h, 060h, 066h, 06ch, 078h, 06ch, 0e6h,
000h
 52850 000129F9 60666C786CE600      <1>
 52851 00012A00 703030303030780000- <1>     db  070h, 030h, 030h, 030h, 030h, 030h, 078h, 000h, 000h, 000h, 0cch, 0feh, 0feh, 0d6h, 0c6h,
000h
 52852 00012A09 00CCFEFED6C600      <1>
 52853 00012A10 0000F8CCCCCCCC0000- <1>     db  000h, 000h, 0f8h, 0cch, 0cch, 0cch, 0cch, 000h, 000h, 000h, 078h, 0cch, 0cch, 0cch, 078h,
000h
 52854 00012A19 0078CCCCCC7800      <1>
 52855 00012A20 0000DC66667C60F000- <1>     db  000h, 000h, 0dch, 066h, 066h, 07ch, 060h, 0f0h, 000h, 000h, 076h, 0cch, 0cch, 07ch, 00ch,
01eh
 52856 00012A29 0076CCCC7C0C1E      <1>
 52857 00012A30 0000DC766660F00000- <1>     db  000h, 000h, 0dch, 076h, 066h, 060h, 0f0h, 000h, 000h, 000h, 07ch, 0c0h, 078h, 00ch, 0f8h,
000h
 52858 00012A39 007CC0780CF800      <1>
 52859 00012A40 10307C303034180000- <1>     db  010h, 030h, 07ch, 030h, 030h, 034h, 018h, 000h, 000h, 000h, 0cch, 0cch, 0cch, 0cch, 076h,
000h
 52860 00012A49 00CCCCCCCC7600      <1>
 52861 00012A50 0000CCCCCC78300000- <1>     db  000h, 000h, 0cch, 0cch, 0cch, 078h, 030h, 000h, 000h, 000h, 0c6h, 0d6h, 0feh, 0feh, 06ch,
000h
 52862 00012A59 00C6D6FEFE6C00      <1>
 52863 00012A60 0000C66C386CC60000- <1>     db  000h, 000h, 0c6h, 06ch, 038h, 06ch, 0c6h, 000h, 000h, 000h, 0cch, 0cch, 0cch, 07ch, 00ch,
0f8h
 52864 00012A69 00CCCCCC7C0CF8      <1>
 52865 00012A70 0000FC983064FC001C- <1>     db  000h, 000h, 0fch, 098h, 030h, 064h, 0fch, 000h, 01ch, 030h, 030h, 0e0h, 030h, 030h, 01ch,
000h
 52866 00012A79 3030E030301C00      <1>
 52867 00012A80 18181800181818 00E0- <1>    db  018h, 018h, 018h, 000h, 018h, 018h, 018h, 000h, 0e0h, 030h, 030h, 01ch, 030h, 030h, 0e0h,
000h
 52868 00012A89 30301C3030E000      <1>
 52869 00012A90 76DC00000000000000- <1>     db  076h, 0dch, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 010h, 038h, 06ch, 0c6h, 0c6h, 0feh,
000h
 52870 00012A99 10386CC6C6FE00      <1>
 52871 00012AA0 78CCC0CC78180C7800- <1>     db  078h, 0cch, 0c0h, 0cch, 078h, 018h, 00ch, 078h, 000h, 0cch, 000h, 0cch, 0cch, 0cch, 07eh,
000h
 52872 00012AA9 CC00CCCCCC7E00      <1>
 52873 00012AB0 1C0078CCFCC078007E- <1>     db  01ch, 000h, 078h, 0cch, 0fch, 0c0h, 078h, 000h, 07eh, 0c3h, 03ch, 006h, 03eh, 066h, 03fh,
000h
 52874 00012AB9 C33C063E663F00      <1>
 52875 00012AC0 CC00780C7CCC7E00E0- <1>     db  0cch, 000h, 078h, 00ch, 07ch, 0cch, 07eh, 000h, 0e0h, 000h, 078h, 00ch, 07ch, 0cch, 07eh,
000h
 52876 00012AC9 00780C7CCC7E00      <1>
 52877 00012AD0 3030780C7CCC7E0000- <1>     db  030h, 030h, 078h, 00ch, 07ch, 0cch, 07eh, 000h, 000h, 000h, 078h, 0c0h, 0c0h, 078h, 00ch,
038h
 52878 00012AD9 0078C0C0780C38      <1>
```

```
 52879 00012AE0 7EC33C667E603C00CC- <1>     db  07eh, 0c3h, 03ch, 066h, 07eh, 060h, 03ch, 000h, 0cch, 000h, 078h, 0cch, 0fch, 0c0h, 078h,
000h
 52880 00012AE9 0078CCFCC07800      <1>
 52881 00012AF0 E00078CCFCC07800CC- <1>     db  0e0h, 000h, 078h, 0cch, 0fch, 0c0h, 078h, 000h, 0cch, 000h, 070h, 030h, 030h, 030h, 078h,
000h
 52882 00012AF9 00703030307800      <1>
 52883 00012B00 7CC6381818183C00E0- <1>     db  07ch, 0c6h, 038h, 018h, 018h, 018h, 03ch, 000h, 0e0h, 000h, 070h, 030h, 030h, 030h, 078h,
000h
 52884 00012B09 00703030307800      <1>
 52885 00012B10 C6386CC6FEC6C60030- <1>     db  0c6h, 038h, 06ch, 0c6h, 0feh, 0c6h, 0c6h, 000h, 030h, 030h, 000h, 078h, 0cch, 0fch, 0cch,
000h
 52886 00012B19 300078CCFCCC00      <1>
 52887 00012B20 1C00FC607860FC0000- <1>     db  01ch, 000h, 0fch, 060h, 078h, 060h, 0fch, 000h, 000h, 000h, 07fh, 00ch, 07fh, 0cch, 07fh,
000h
 52888 00012B29 007F0C7FCC7F00      <1>
 52889 00012B30 3E6CCCFECCCCCE0078- <1>     db  03eh, 06ch, 0cch, 0feh, 0cch, 0cch, 0ceh, 000h, 078h, 0cch, 000h, 078h, 0cch, 0cch, 078h,
000h
 52890 00012B39 CC0078CCCC7800      <1>
 52891 00012B40 00CC0078CCCC780000- <1>     db  000h, 0cch, 000h, 078h, 0cch, 0cch, 078h, 000h, 000h, 0e0h, 000h, 078h, 0cch, 0cch, 078h,
000h
 52892 00012B49 E00078CCCC7800      <1>
 52893 00012B50 78CC00CCCCCC7E0000- <1>     db  078h, 0cch, 000h, 0cch, 0cch, 0cch, 07eh, 000h, 000h, 0e0h, 000h, 0cch, 0cch, 0cch, 07eh,
000h
 52894 00012B59 E000CCCCCC7E00      <1>
 52895 00012B60 00CC00CCCC7C0CF8C3- <1>     db  000h, 0cch, 000h, 0cch, 0cch, 07ch, 00ch, 0f8h, 0c3h, 018h, 03ch, 066h, 066h, 03ch, 018h,
000h
 52896 00012B69 183C66663C1800      <1>
 52897 00012B70 CC00CCCCCCCC780018- <1>     db  0cch, 000h, 0cch, 0cch, 0cch, 0cch, 078h, 000h, 018h, 018h, 07eh, 0c0h, 0c0h, 07eh, 018h,
018h
 52898 00012B79 187EC0C07E1818      <1>
 52899 00012B80 386C64F060E6FC00CC- <1>     db  038h, 06ch, 064h, 0f0h, 060h, 0e6h, 0fch, 000h, 0cch, 0cch, 078h, 0fch, 030h, 0fch, 030h,
030h
 52900 00012B89 CC78FC30FC3030      <1>
 52901 00012B90 F8CCCCFAC6CFC6C70E- <1>     db  0f8h, 0cch, 0cch, 0fah, 0c6h, 0cfh, 0c6h, 0c7h, 00eh, 01bh, 018h, 03ch, 018h, 018h, 0d8h,
070h
 52902 00012B99 1B183C1818D870      <1>
 52903 00012BA0 1C00780C7CCC7E0038- <1>     db  01ch, 000h, 078h, 00ch, 07ch, 0cch, 07eh, 000h, 038h, 000h, 070h, 030h, 030h, 030h, 078h,
000h
 52904 00012BA9 00703030307800      <1>
 52905 00012BB0 001C0078CCCC780000- <1>     db  000h, 01ch, 000h, 078h, 0cch, 0cch, 078h, 000h, 000h, 01ch, 000h, 0cch, 0cch, 0cch, 07eh,
000h
 52906 00012BB9 1C00CCCCCC7E00      <1>
 52907 00012BC0 00F800F8CCCCCC00FC- <1>     db  000h, 0f8h, 000h, 0f8h, 0cch, 0cch, 0cch, 000h, 0fch, 000h, 0cch, 0ech, 0fch, 0dch, 0cch,
000h
 52908 00012BC9 00CCECFCDCCC00      <1>
 52909 00012BD0 3C6C6C3E007E000038- <1>     db  03ch, 06ch, 06ch, 03eh, 000h, 07eh, 000h, 000h, 038h, 06ch, 06ch, 038h, 000h, 07ch, 000h,
000h
 52910 00012BD9 6C6C38007C0000      <1>
 52911 00012BE0 30003060C0CC780000- <1>     db  030h, 000h, 030h, 060h, 0c0h, 0cch, 078h, 000h, 000h, 000h, 000h, 0fch, 0c0h, 0c0h, 000h,
000h
 52912 00012BE9 0000FCC0C00000      <1>
 52913 00012BF0 000000FC0C0C0000C3- <1>     db  000h, 000h, 000h, 0fch, 00ch, 00ch, 000h, 000h, 0c3h, 0c6h, 0cch, 0deh, 033h, 066h, 0cch,
00fh
 52914 00012BF9 C6CCDE3366CC0F      <1>
 52915 00012C00 C3C6CCDB376FCF0318- <1>     db  0c3h, 0c6h, 0cch, 0dbh, 037h, 06fh, 0cfh, 003h, 018h, 018h, 000h, 018h, 018h, 018h, 018h,
000h
 52916 00012C09 18001818181800      <1>
 52917 00012C10 003366CC6633000000- <1>     db  000h, 033h, 066h, 0cch, 066h, 033h, 000h, 000h, 000h, 0cch, 066h, 033h, 066h, 0cch, 000h,
000h
 52918 00012C19 CC663366CC0000      <1>
 52919 00012C20 228822882288228855- <1>     db  022h, 088h, 022h, 088h, 022h, 088h, 022h, 088h, 055h, 0aah, 055h, 0aah, 055h, 0aah, 055h,
0aah
 52920 00012C29 AA55AA55AA55AA      <1>
 52921 00012C30 DB77DBEEDB77DBEE18- <1>     db  0dbh, 077h, 0dbh, 0eeh, 0dbh, 077h, 0dbh, 0eeh, 018h, 018h, 018h, 018h, 018h, 018h, 018h,
018h
 52922 00012C39 18181818181818      <1>
 52923 00012C40 18181818F818181818- <1>     db  018h, 018h, 018h, 018h, 0f8h, 018h, 018h, 018h, 018h, 018h, 0f8h, 018h, 0f8h, 018h, 018h,
018h
 52924 00012C49 18F818F8181818      <1>
 52925 00012C50 36363636F636363600- <1>     db  036h, 036h, 036h, 036h, 0f6h, 036h, 036h, 036h, 000h, 000h, 000h, 000h, 0feh, 036h, 036h,
036h
 52926 00012C59 000000FE363636      <1>
 52927 00012C60 0000F818F818181836- <1>     db  000h, 000h, 0f8h, 018h, 0f8h, 018h, 018h, 018h, 036h, 036h, 0f6h, 006h, 0f6h, 036h, 036h,
036h
 52928 00012C69 36F606F6363636      <1>
 52929 00012C70 363636363636363600- <1>     db  036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 000h, 000h, 0feh, 006h, 0f6h, 036h, 036h,
036h
 52930 00012C79 00FE06F6363636      <1>
 52931 00012C80 3636F606FE00000036- <1>     db  036h, 036h, 0f6h, 006h, 0feh, 000h, 000h, 000h, 036h, 036h, 036h, 036h, 0feh, 000h, 000h,
000h
 52932 00012C89 363636FE000000      <1>
 52933 00012C90 1818F818F800000000- <1>     db  018h, 018h, 0f8h, 018h, 0f8h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0f8h, 018h, 018h,
018h
 52934 00012C99 000000F8181818      <1>
 52935 00012CA0 181818181F00000018- <1>     db  018h, 018h, 018h, 018h, 01fh, 000h, 000h, 000h, 018h, 018h, 018h, 018h, 0ffh, 000h, 000h,
000h
 52936 00012CA9 181818FF000000      <1>
 52937 00012CB0 00000000FF18181818- <1>     db  000h, 000h, 000h, 000h, 0ffh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 01fh, 018h, 018h,
018h
 52938 00012CB9 1818181F181818      <1>
 52939 00012CC0 00000000FF00000018- <1>     db  000h, 000h, 000h, 000h, 0ffh, 000h, 000h, 000h, 018h, 018h, 018h, 018h, 0ffh, 018h, 018h,
018h
 52940 00012CC9 181818FF181818      <1>
 52941 00012CD0 18181F181F18181836- <1>     db  018h, 018h, 01fh, 018h, 01fh, 018h, 018h, 018h, 036h, 036h, 036h, 036h, 037h, 036h, 036h,
036h
 52942 00012CD9 36363637363636      <1>
 52943 00012CE0 363637303F00000000- <1>     db  036h, 036h, 037h, 030h, 03fh, 000h, 000h, 000h, 000h, 000h, 03fh, 030h, 037h, 036h, 036h,
036h
 52944 00012CE9 003F3037363636      <1>
 52945 00012CF0 3636F700FF00000000- <1>     db  036h, 036h, 0f7h, 000h, 0ffh, 000h, 000h, 000h, 000h, 000h, 0ffh, 000h, 0f7h, 036h, 036h,
036h
 52946 00012CF9 00FF00F7363636      <1>
```

```
 52947 00012D00 363637303736363600- <1>     db  036h, 036h, 037h, 030h, 037h, 036h, 036h, 036h, 000h, 000h, 0ffh, 000h, 0ffh, 000h, 000h,
000h
 52948 00012D09 00FF00FF000000      <1>
 52949 00012D10 3636F700F736363618- <1>     db  036h, 036h, 0f7h, 000h, 0f7h, 036h, 036h, 036h, 018h, 018h, 0ffh, 000h, 0ffh, 000h, 000h,
000h
 52950 00012D19 18FF00FF000000      <1>
 52951 00012D20 36363636FF00000000- <1>     db  036h, 036h, 036h, 036h, 0ffh, 000h, 000h, 000h, 000h, 000h, 0ffh, 000h, 0ffh, 018h, 018h,
018h
 52952 00012D29 00FF00FF181818      <1>
 52953 00012D30 00000000FF36363636- <1>     db  000h, 000h, 000h, 000h, 0ffh, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 03fh, 000h, 000h,
000h
 52954 00012D39 3636363F000000      <1>
 52955 00012D40 18181F181F00000000- <1>     db  018h, 018h, 01fh, 018h, 01fh, 000h, 000h, 000h, 000h, 000h, 01fh, 018h, 01fh, 018h, 018h,
018h
 52956 00012D49 001F181F181818      <1>
 52957 00012D50 000000003F36363636- <1>     db  000h, 000h, 000h, 000h, 03fh, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 0ffh, 036h, 036h,
036h
 52958 00012D59 363636FF363636      <1>
 52959 00012D60 1818FF18FF18181818- <1>     db  018h, 018h, 0ffh, 018h, 0ffh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 0f8h, 000h, 000h,
000h
 52960 00012D69 181818F8000000      <1>
 52961 00012D70 000000001F181818FF- <1>     db  000h, 000h, 000h, 000h, 01fh, 018h, 018h, 018h, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh,
0ffh
 52962 00012D79 FFFFFFFFFFFFFF      <1>
 52963 00012D80 00000000FFFFFFFFF0- <1>     db  000h, 000h, 000h, 000h, 0ffh, 0ffh, 0ffh, 0ffh, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h,
0f0h
 52964 00012D89 F0F0F0F0F0F0F0      <1>
 52965 00012D90 0F0F0F0F0F0F0F0FFF- <1>     db  00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 0ffh, 0ffh, 0ffh, 0ffh, 000h, 000h, 000h,
000h
 52966 00012D99 FFFFFF00000000      <1>
 52967 00012DA0 000076DCC8DC760000- <1>     db  000h, 000h, 076h, 0dch, 0c8h, 0dch, 076h, 000h, 000h, 078h, 0cch, 0f8h, 0cch, 0f8h, 0c0h,
0c0h
 52968 00012DA9 78CCF8CCF8C0C0      <1>
 52969 00012DB0 00FCCCC0C0C0C00000- <1>     db  000h, 0fch, 0cch, 0c0h, 0c0h, 0c0h, 0c0h, 000h, 000h, 0feh, 06ch, 06ch, 06ch, 06ch, 06ch,
000h
 52970 00012DB9 FE6C6C6C6C6C00      <1>
 52971 00012DC0 FCCC603060CCFC0000- <1>     db  0fch, 0cch, 060h, 030h, 060h, 0cch, 0fch, 000h, 000h, 000h, 07eh, 0d8h, 0d8h, 0d8h, 070h,
000h
 52972 00012DC9 007ED8D8D87000      <1>
 52973 00012DD0 00666666667C60C000- <1>     db  000h, 066h, 066h, 066h, 066h, 07ch, 060h, 0c0h, 000h, 076h, 0dch, 018h, 018h, 018h, 018h,
000h
 52974 00012DD9 76DC1818181800      <1>
 52975 00012DE0 FC3078CCCC7830FC38- <1>     db  0fch, 030h, 078h, 0cch, 0cch, 078h, 030h, 0fch, 038h, 06ch, 0c6h, 0feh, 0c6h, 06ch, 038h,
000h
 52976 00012DE9 6CC6FEC66C3800      <1>
 52977 00012DF0 386CC6C66C6CEE001C- <1>     db  038h, 06ch, 0c6h, 0c6h, 06ch, 06ch, 0eeh, 000h, 01ch, 030h, 018h, 07ch, 0cch, 0cch, 078h,
000h
 52978 00012DF9 30187CCCCC7800      <1>
 52979 00012E00 00007EDBDB7E000006- <1>     db  000h, 000h, 07eh, 0dbh, 0dbh, 07eh, 000h, 000h, 006h, 00ch, 07eh, 0dbh, 0dbh, 07eh, 060h,
0c0h
 52980 00012E09 0C7EDBDB7E60C0      <1>
 52981 00012E10 3860C0F8C060380078- <1>     db  038h, 060h, 0c0h, 0f8h, 0c0h, 060h, 038h, 000h, 078h, 0cch, 0cch, 0cch, 0cch, 0cch, 0cch,
000h
 52982 00012E19 CCCCCCCCCCCC00      <1>
 52983 00012E20 00FC00FC00FC000030- <1>     db  000h, 0fch, 000h, 0fch, 000h, 0fch, 000h, 000h, 030h, 030h, 0fch, 030h, 030h, 000h, 0fch,
000h
 52984 00012E29 30FC303000FC00      <1>
 52985 00012E30 603018306000FC0018- <1>     db  060h, 030h, 018h, 030h, 060h, 000h, 0fch, 000h, 018h, 030h, 060h, 030h, 018h, 000h, 0fch,
000h
 52986 00012E39 3060301800FC00      <1>
 52987 00012E40 0E1B1B181818181818- <1>     db  00eh, 01bh, 01bh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 0d8h, 0d8h,
070h
 52988 00012E49 18181818D8D870      <1>
 52989 00012E50 303000FC0030300000- <1>     db  030h, 030h, 000h, 0fch, 000h, 030h, 030h, 000h, 000h, 076h, 0dch, 000h, 076h, 0dch, 000h,
000h
 52990 00012E59 76DC0076DC0000      <1>
 52991 00012E60 386C6C380000000000- <1>     db  038h, 06ch, 06ch, 038h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 018h, 018h, 000h, 000h,
000h
 52992 00012E69 00001818000000      <1>
 52993 00012E70 000000001800000000F- <1>    db  000h, 000h, 000h, 000h, 018h, 000h, 000h, 000h, 00fh, 00ch, 00ch, 00ch, 0ech, 06ch, 03ch,
01ch
 52994 00012E79 0C0C0CEC6C3C1C      <1>
 52995 00012E80 786C6C6C6C00000070- <1>     db  078h, 06ch, 06ch, 06ch, 06ch, 000h, 000h, 000h, 070h, 018h, 030h, 060h, 078h, 000h, 000h,
000h
 52996 00012E89 18306078000000      <1>
 52997 00012E90 00003C3C3C3C000000- <1>     db  000h, 000h, 03ch, 03ch, 03ch, 03ch, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 52998 00012E99 00000000000000      <1>
 52999                              <1> vgafont14:
 53000 00012EA0 000000000000000000- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 53001 00012EA9 00000000000000      <1>
 53002 00012EB0 7E81A58181BD99817E- <1>     db  07eh, 081h, 0a5h, 081h, 081h, 0bdh, 099h, 081h, 07eh, 000h, 000h, 000h, 000h, 000h, 07eh,
0ffh
 53003 00012EB9 00000000007EFF      <1>
 53004 00012EC0 DBFFFFC3E7FF7E0000- <1>     db  0dbh, 0ffh, 0ffh, 0c3h, 0e7h, 0ffh, 07eh, 000h, 000h, 000h, 000h, 000h, 000h, 06ch, 0feh,
0feh
 53005 00012EC9 000000006CFEFE      <1>
 53006 00012ED0 FEFE7C3810000000000- <1>    db  0feh, 0feh, 07ch, 038h, 010h, 000h, 000h, 000h, 000h, 000h, 000h, 010h, 038h, 07ch, 0feh,
07ch
 53007 00012ED9 000010387CFE7C      <1>
 53008 00012EE0 3810000000000000018- <1>    db  038h, 010h, 000h, 000h, 000h, 000h, 000h, 000h, 018h, 03ch, 03ch, 0e7h, 0e7h, 0e7h, 018h,
018h
 53009 00012EE9 3C3CE7E7E71818      <1>
 53010 00012EF0 3C0000000000183C7E- <1>     db  03ch, 000h, 000h, 000h, 000h, 000h, 018h, 03ch, 07eh, 0ffh, 0ffh, 07eh, 018h, 018h, 03ch,
000h
 53011 00012EF9 FFFF7E18183C00      <1>
 53012 00012F00 000000000000000183C- <1>    db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 018h, 03ch, 03ch, 018h, 000h, 000h, 000h, 000h,
000h
 53013 00012F09 3C180000000000      <1>
 53014 00012F10 FFFFFFFFFFE7C3C3E7- <1>     db  0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0e7h, 0c3h, 0c3h, 0e7h, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 000h,
000h
 53015 00012F19 FFFFFFFFFF0000      <1>
```

```
 53016 00012F20 00003C664242663C00- <1>     db  000h, 000h, 03ch, 066h, 042h, 042h, 066h, 03ch, 000h, 000h, 000h, 000h, 0ffh, 0ffh, 0ffh,
0ffh
 53017 00012F29 000000FFFFFFFF      <1>
 53018 00012F30 C399BDBD99C3FFFFFF- <1>     db  0c3h, 099h, 0bdh, 0bdh, 099h, 0c3h, 0ffh, 0ffh, 0ffh, 0ffh, 000h, 000h, 01eh, 00eh, 01ah,
032h
 53019 00012F39 FF00001E0E1A32      <1>
 53020 00012F40 78CCCCCC7800000000- <1>     db  078h, 0cch, 0cch, 0cch, 078h, 000h, 000h, 000h, 000h, 000h, 03ch, 066h, 066h, 066h, 03ch,
018h
 53021 00012F49 003C6666663C18      <1>
 53022 00012F50 7E1818000000000003F- <1>     db  07eh, 018h, 018h, 000h, 000h, 000h, 000h, 000h, 03fh, 033h, 03fh, 030h, 030h, 030h, 070h,
0f0h
 53023 00012F59 333F30303070F0      <1>
 53024 00012F60 E0000000000007F637F- <1>     db  0e0h, 000h, 000h, 000h, 000h, 000h, 07fh, 063h, 07fh, 063h, 063h, 063h, 067h, 0e7h, 0e6h,
0c0h
 53025 00012F69 63636367E7E6C0      <1>
 53026 00012F70 000000001818DB3CE7- <1>     db  000h, 000h, 000h, 000h, 018h, 018h, 0dbh, 03ch, 0e7h, 03ch, 0dbh, 018h, 018h, 000h, 000h,
000h
 53027 00012F79 3CDB1818000000      <1>
 53028 00012F80 000080C0E0F8FEF8E0- <1>     db  000h, 000h, 080h, 0c0h, 0e0h, 0f8h, 0feh, 0f8h, 0e0h, 0c0h, 080h, 000h, 000h, 000h, 000h,
000h
 53029 00012F89 C0800000000000      <1>
 53030 00012F90 02060E3EFE3E0E0602- <1>     db  002h, 006h, 00eh, 03eh, 0feh, 03eh, 00eh, 006h, 002h, 000h, 000h, 000h, 000h, 000h, 018h,
03ch
 53031 00012F99 0000000000183C      <1>
 53032 00012FA0 7E1818187E3C180000- <1>     db  07eh, 018h, 018h, 018h, 07eh, 03ch, 018h, 000h, 000h, 000h, 000h, 000h, 066h, 066h, 066h,
066h
 53033 00012FA9 00000066666666      <1>
 53034 00012FB0 666600666600000000- <1>     db  066h, 066h, 000h, 066h, 066h, 000h, 000h, 000h, 000h, 000h, 07fh, 0dbh, 0dbh, 0dbh, 07bh,
01bh
 53035 00012FB9 007FDBDBDB7B1B      <1>
 53036 00012FC0 1B1B1B0000000007CC6- <1>     db  01bh, 01bh, 01bh, 000h, 000h, 000h, 000h, 07ch, 0c6h, 060h, 038h, 06ch, 0c6h, 0c6h, 06ch,
038h
 53037 00012FC9 60386CC6C66C38      <1>
 53038 00012FD0 0CC67C000000000000- <1>     db  00ch, 0c6h, 07ch, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0feh, 0feh, 0feh,
000h
 53039 00012FD9 000000FEFEFE00      <1>
 53040 00012FE0 00000000183C7E1818- <1>     db  000h, 000h, 000h, 000h, 018h, 03ch, 07eh, 018h, 018h, 018h, 07eh, 03ch, 018h, 07eh, 000h,
000h
 53041 00012FE9 187E3C187E0000      <1>
 53042 00012FF0 0000183C7E18181818- <1>     db  000h, 000h, 018h, 03ch, 07eh, 018h, 018h, 018h, 018h, 018h, 018h, 000h, 000h, 000h, 000h,
000h
 53043 00012FF9 18180000000000      <1>
 53044 00013000 1818181818187E3C18- <1>     db  018h, 018h, 018h, 018h, 018h, 018h, 07eh, 03ch, 018h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 53045 00013009 00000000000000      <1>
 53046 00013010 180CFE0C1800000000- <1>     db  018h, 00ch, 0feh, 00ch, 018h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 030h,
060h
 53047 00013019 00000000003060      <1>
 53048 00013020 FE60300000000000000- <1>     db  0feh, 060h, 030h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0c0h, 0c0h,
0c0h
 53049 00013029 00000000C0C0C0      <1>
 53050 00013030 FE00000000000000000- <1>     db  0feh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 028h, 06ch, 0feh, 06ch, 028h,
000h
 53051 00013039 00286CFE6C2800      <1>
 53052 00013040 000000000000001038- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 010h, 038h, 038h, 07ch, 07ch, 0feh, 0feh, 000h,
000h
 53053 00013049 387C7CFEFE0000      <1>
 53054 00013050 0000000000FEFE7C7C- <1>     db  000h, 000h, 000h, 000h, 000h, 0feh, 0feh, 07ch, 07ch, 038h, 038h, 010h, 000h, 000h, 000h,
000h
 53055 00013059 38381000000000      <1>
 53056 00013060 000000000000000000- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 53057 00013069 00000000000000      <1>
 53058 00013070 183C3C3C1818001818- <1>     db  018h, 03ch, 03ch, 03ch, 018h, 018h, 000h, 018h, 018h, 000h, 000h, 000h, 000h, 066h, 066h,
066h
 53059 00013079 00000000666666      <1>
 53060 00013080 240000000000000000- <1>     db  024h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 06ch, 06ch, 0feh,
06ch
 53061 00013089 0000006C6CFE6C      <1>
 53062 00013090 6C6CFE6C6C00000018- <1>     db  06ch, 06ch, 0feh, 06ch, 06ch, 000h, 000h, 000h, 018h, 018h, 07ch, 0c6h, 0c2h, 0c0h, 07ch,
006h
 53063 00013099 187CC6C2C07C06      <1>
 53064 000130A0 86C67C1818000000000- <1>     db  086h, 0c6h, 07ch, 018h, 018h, 000h, 000h, 000h, 000h, 000h, 0c2h, 0c6h, 00ch, 018h, 030h,
066h
 53065 000130A9 00C2C60C183066      <1>
 53066 000130B0 C60000000000386C6C- <1>     db  0c6h, 000h, 000h, 000h, 000h, 000h, 038h, 06ch, 06ch, 038h, 076h, 0dch, 0cch, 0cch, 076h,
000h
 53067 000130B9 3876DCCCCC7600      <1>
 53068 000130C0 000000303030600000- <1>     db  000h, 000h, 000h, 030h, 030h, 030h, 060h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 53069 000130C9 00000000000000      <1>
 53070 000130D0 00000C183030303030- <1>     db  000h, 000h, 00ch, 018h, 030h, 030h, 030h, 030h, 030h, 018h, 00ch, 000h, 000h, 000h, 000h,
000h
 53071 000130D9 180C0000000000      <1>
 53072 000130E0 30180C0C0C0C0C1830- <1>     db  030h, 018h, 00ch, 00ch, 00ch, 00ch, 00ch, 018h, 030h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 53073 000130E9 00000000000000      <1>
 53074 000130F0 663CFF3C6600000000- <1>     db  066h, 03ch, 0ffh, 03ch, 066h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 018h,
018h
 53075 000130F9 00000000001818      <1>
 53076 00013100 7E1818000000000000- <1>     db  07eh, 018h, 018h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 53077 00013109 00000000000000      <1>
 53078 00013110 1818183000000000000- <1>     db  018h, 018h, 018h, 030h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0feh, 000h, 000h,
000h
 53079 00013119 000000FE000000      <1>
 53080 00013120 000000000000000000- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 018h, 018h,
000h
 53081 00013129 00000000181800      <1>
 53082 00013130 0000000002060C1830- <1>     db  000h, 000h, 000h, 000h, 002h, 006h, 00ch, 018h, 030h, 060h, 0c0h, 080h, 000h, 000h, 000h,
000h
 53083 00013139 60C08000000000      <1>
```

```
 53084 00013140 00007CC6CEDEF6E6C6- <1>     db  000h, 000h, 07ch, 0c6h, 0ceh, 0deh, 0f6h, 0e6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h,
000h
 53085 00013149 C67C0000000000      <1>
 53086 00013150 18387818181818187E- <1>     db  018h, 038h, 078h, 018h, 018h, 018h, 018h, 018h, 07eh, 000h, 000h, 000h, 000h, 000h, 07ch,
0c6h
 53087 00013159 00000000007CC6      <1>
 53088 00013160 060C183060C6FE0000- <1>     db  006h, 00ch, 018h, 030h, 060h, 0c6h, 0feh, 000h, 000h, 000h, 000h, 000h, 07ch, 0c6h, 006h,
006h
 53089 00013169 0000007CC60606      <1>
 53090 00013170 3C0606C67C00000000- <1>     db  03ch, 006h, 006h, 0c6h, 07ch, 000h, 000h, 000h, 000h, 000h, 00ch, 01ch, 03ch, 06ch, 0cch,
0feh
 53091 00013179 000C1C3C6CCCFE      <1>
 53092 00013180 0C0C1E0000000000FE- <1>     db  00ch, 00ch, 01eh, 000h, 000h, 000h, 000h, 000h, 0feh, 0c0h, 0c0h, 0c0h, 0fch, 006h, 006h,
0c6h
 53093 00013189 C0C0C0FC0606C6      <1>
 53094 00013190 7C000000000003860C0- <1>     db  07ch, 000h, 000h, 000h, 000h, 000h, 038h, 060h, 0c0h, 0c0h, 0fch, 0c6h, 0c6h, 0c6h, 07ch,
000h
 53095 00013199 C0FCC6C6C67C00      <1>
 53096 000131A0 00000000FEC6060C18- <1>     db  000h, 000h, 000h, 000h, 0feh, 0c6h, 006h, 00ch, 018h, 030h, 030h, 030h, 030h, 000h, 000h,
000h
 53097 000131A9 30303030000000      <1>
 53098 000131B0 00007CC6C6C67CC6C6- <1>     db  000h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 07ch, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h,
000h
 53099 000131B9 C67C0000000000      <1>
 53100 000131C0 7CC6C6C67E06060C78- <1>     db  07ch, 0c6h, 0c6h, 0c6h, 07eh, 006h, 006h, 00ch, 078h, 000h, 000h, 000h, 000h, 000h, 000h,
018h
 53101 000131C9 00000000000018      <1>
 53102 000131D0 1800000018180000000- <1>     db  018h, 000h, 000h, 000h, 018h, 018h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 018h, 018h,
000h
 53103 000131D9 00000000181800      <1>
 53104 000131E0 0000181830000000000- <1>     db  000h, 000h, 018h, 018h, 030h, 000h, 000h, 000h, 000h, 000h, 006h, 00ch, 018h, 030h, 060h,
030h
 53105 000131E9 00060C18306030      <1>
 53106 000131F0 180C060000000000000- <1>     db  018h, 00ch, 006h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 07eh, 000h, 000h, 07eh,
000h
 53107 000131F9 00007E00007E00      <1>
 53108 00013200 0000000000000603018- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 060h, 030h, 018h, 00ch, 006h, 00ch, 018h, 030h, 060h,
000h
 53109 00013209 0C060C18306000      <1>
 53110 00013210 000000007CC6C60C18- <1>     db  000h, 000h, 000h, 000h, 07ch, 0c6h, 0c6h, 00ch, 018h, 018h, 000h, 018h, 018h, 000h, 000h,
000h
 53111 00013219 18001818000000      <1>
 53112 00013220 00007CC6C6DEDEDEDC- <1>     db  000h, 000h, 07ch, 0c6h, 0c6h, 0deh, 0deh, 0deh, 0dch, 0c0h, 07ch, 000h, 000h, 000h, 000h,
000h
 53113 00013229 C07C0000000000      <1>
 53114 00013230 10386CC6C6FEC6C6C6- <1>     db  010h, 038h, 06ch, 0c6h, 0c6h, 0feh, 0c6h, 0c6h, 0c6h, 000h, 000h, 000h, 000h, 000h, 0fch,
066h
 53115 00013239 0000000000FC66      <1>
 53116 00013240 66667C666666FC0000- <1>     db  066h, 066h, 07ch, 066h, 066h, 066h, 0fch, 000h, 000h, 000h, 000h, 000h, 03ch, 066h, 0c2h,
0c0h
 53117 00013249 0000003C66C2C0      <1>
 53118 00013250 C0C0C2663C00000000- <1>     db  0c0h, 0c0h, 0c2h, 066h, 03ch, 000h, 000h, 000h, 000h, 000h, 0f8h, 06ch, 066h, 066h, 066h,
066h
 53119 00013259 00F86C66666666      <1>
 53120 00013260 666CF800000000000FE- <1>     db  066h, 06ch, 0f8h, 000h, 000h, 000h, 000h, 000h, 0feh, 066h, 062h, 068h, 078h, 068h, 062h,
066h
 53121 00013269 66626878686266      <1>
 53122 00013270 FE0000000000FE6662- <1>     db  0feh, 000h, 000h, 000h, 000h, 000h, 0feh, 066h, 062h, 068h, 078h, 068h, 060h, 060h, 0f0h,
000h
 53123 00013279 6878686060F000      <1>
 53124 00013280 000000003C66C2C0C0- <1>     db  000h, 000h, 000h, 000h, 03ch, 066h, 0c2h, 0c0h, 0c0h, 0deh, 0c6h, 066h, 03ah, 000h, 000h,
000h
 53125 00013289 DEC6663A000000      <1>
 53126 00013290 0000C6C6C6C6FEC6C6- <1>     db  000h, 000h, 0c6h, 0c6h, 0c6h, 0c6h, 0feh, 0c6h, 0c6h, 0c6h, 0c6h, 000h, 000h, 000h, 000h,
000h
 53127 00013299 C6C60000000000      <1>
 53128 000132A0 3C181818181818183C- <1>     db  03ch, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h, 000h, 000h, 01eh,
00ch
 53129 000132A9 00000000001E0C      <1>
 53130 000132B0 0C0C0C0CCCCC780000- <1>     db  00ch, 00ch, 00ch, 00ch, 0cch, 0cch, 078h, 000h, 000h, 000h, 000h, 000h, 0e6h, 066h, 06ch,
06ch
 53131 000132B9 000000E6666C6C      <1>
 53132 000132C0 786C6C66E600000000- <1>     db  078h, 06ch, 06ch, 066h, 0e6h, 000h, 000h, 000h, 000h, 000h, 0f0h, 060h, 060h, 060h, 060h,
060h
 53133 000132C9 00F06060606060      <1>
 53134 000132D0 6266FE0000000000C6- <1>     db  062h, 066h, 0feh, 000h, 000h, 000h, 000h, 000h, 0c6h, 0eeh, 0feh, 0feh, 0d6h, 0c6h, 0c6h,
0c6h
 53135 000132D9 EEFEFED6C6C6C6      <1>
 53136 000132E0 C600000000000C6E6F6- <1>     db  0c6h, 000h, 000h, 000h, 000h, 000h, 0c6h, 0e6h, 0f6h, 0feh, 0deh, 0ceh, 0c6h, 0c6h, 0c6h,
000h
 53137 000132E9 FEDECEC6C6C600      <1>
 53138 000132F0 00000000386CC6C6C6- <1>     db  000h, 000h, 000h, 000h, 038h, 06ch, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 06ch, 038h, 000h, 000h,
000h
 53139 000132F9 C6C66C38000000      <1>
 53140 00013300 0000FC6666667C6060- <1>     db  000h, 000h, 0fch, 066h, 066h, 066h, 07ch, 060h, 060h, 060h, 0f0h, 000h, 000h, 000h, 000h,
000h
 53141 00013309 60F000000000000     <1>
 53142 00013310 7CC6C6C6C6D6DE7C0C- <1>     db  07ch, 0c6h, 0c6h, 0c6h, 0c6h, 0d6h, 0deh, 07ch, 00ch, 00eh, 000h, 000h, 000h, 000h, 0fch,
066h
 53143 00013319 0E00000000FC66      <1>
 53144 00013320 66667C6C6666E60000- <1>     db  066h, 066h, 07ch, 06ch, 066h, 066h, 0e6h, 000h, 000h, 000h, 000h, 000h, 07ch, 0c6h, 0c6h,
060h
 53145 00013329 0000007CC6C660      <1>
 53146 00013330 380CC6C67C00000000- <1>     db  038h, 00ch, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h, 000h, 07eh, 07eh, 05ah, 018h, 018h,
018h
 53147 00013339 007E7E5A181818      <1>
 53148 00013340 18183C00000000000C6- <1>     db  018h, 018h, 03ch, 000h, 000h, 000h, 000h, 000h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h,
0c6h
 53149 00013349 C6C6C6C6C6C6C6      <1>
 53150 00013350 7C0000000000C6C6C6- <1>     db  07ch, 000h, 000h, 000h, 000h, 000h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 06ch, 038h, 010h,
000h
 53151 00013359 C6C6C66C381000      <1>
```

```
 53152 00013360 00000000C6C6C6C6D6- <1>     db  000h, 000h, 000h, 000h, 0c6h, 0c6h, 0c6h, 0c6h, 0d6h, 0d6h, 0feh, 07ch, 06ch, 000h, 000h,
000h
 53153 00013369 D6FE7C6C000000      <1>
 53154 00013370 0000C6C66C3838386C- <1>     db  000h, 000h, 0c6h, 0c6h, 06ch, 038h, 038h, 038h, 06ch, 0c6h, 0c6h, 000h, 000h, 000h, 000h,
000h
 53155 00013379 C6C60000000000      <1>
 53156 00013380 666666663C1818183C- <1>     db  066h, 066h, 066h, 066h, 03ch, 018h, 018h, 018h, 03ch, 000h, 000h, 000h, 000h, 000h, 0feh,
0c6h
 53157 00013389 0000000000FEC6      <1>
 53158 00013390 8C183060C2C6FE0000- <1>     db  08ch, 018h, 030h, 060h, 0c2h, 0c6h, 0feh, 000h, 000h, 000h, 000h, 000h, 03ch, 030h, 030h,
030h
 53159 00013399 0000003C303030      <1>
 53160 000133A0 303030303C00000000- <1>     db  030h, 030h, 030h, 030h, 03ch, 000h, 000h, 000h, 000h, 000h, 080h, 0c0h, 0e0h, 070h, 038h,
01ch
 53161 000133A9 0080C0E070381C      <1>
 53162 000133B0 0E0602000000000003C- <1>    db  00eh, 006h, 002h, 000h, 000h, 000h, 000h, 000h, 03ch, 00ch, 00ch, 00ch, 00ch, 00ch, 00ch,
00ch
 53163 000133B9 0C0C0C0C0C0C0C      <1>
 53164 000133C0 3C000000010386CC600- <1>    db  03ch, 000h, 000h, 000h, 010h, 038h, 06ch, 0c6h, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 53165 000133C9 00000000000000      <1>
 53166 000133D0 0000000000000000000- <1>    db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0ffh,
000h
 53167 000133D9 0000000000FF00      <1>
 53168 000133E0 3030180000000000000- <1>    db  030h, 030h, 018h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 53169 000133E9 00000000000000      <1>
 53170 000133F0 000000780C7CCCCC76- <1>     db  000h, 000h, 000h, 078h, 00ch, 07ch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h, 000h, 0e0h,
060h
 53171 000133F9 0000000000E060      <1>
 53172 00013400 60786C6666667C0000- <1>     db  060h, 078h, 06ch, 066h, 066h, 066h, 07ch, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
07ch
 53173 00013409 0000000000007C      <1>
 53174 00013410 C6C0C0C67C00000000- <1>     db  0c6h, 0c0h, 0c0h, 0c6h, 07ch, 000h, 000h, 000h, 000h, 000h, 01ch, 00ch, 00ch, 03ch, 06ch,
0cch
 53175 00013419 001C0C0C3C6CCC      <1>
 53176 00013420 CCCC760000000000000- <1>    db  0cch, 0cch, 076h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 07ch, 0c6h, 0feh, 0c0h,
0c6h
 53177 00013429 00007CC6FEC0C6      <1>
 53178 00013430 7C0000000000386C64- <1>     db  07ch, 000h, 000h, 000h, 000h, 000h, 038h, 06ch, 064h, 060h, 0f0h, 060h, 060h, 060h, 0f0h,
000h
 53179 00013439 60F0606060F000      <1>
 53180 00013440 0000000000000076CC- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 076h, 0cch, 0cch, 0cch, 07ch, 00ch, 0cch, 078h,
000h
 53181 00013449 CCCC7C0CCC7800      <1>
 53182 00013450 0000E060606C766666- <1>     db  000h, 000h, 0e0h, 060h, 060h, 06ch, 076h, 066h, 066h, 066h, 0e6h, 000h, 000h, 000h, 000h,
000h
 53183 00013459 66E60000000000      <1>
 53184 00013460 18180038181818183C- <1>     db  018h, 018h, 000h, 038h, 018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h, 000h, 000h, 006h,
006h
 53185 00013469 00000000000606      <1>
 53186 00013470 000E06060606666 3C- <1>     db  000h, 00eh, 006h, 006h, 006h, 006h, 066h, 066h, 03ch, 000h, 000h, 000h, 0e0h, 060h, 060h,
066h
 53187 00013479 000000E0606066      <1>
 53188 00013480 6C786C66E600000000- <1>     db  06ch, 078h, 06ch, 066h, 0e6h, 000h, 000h, 000h, 000h, 000h, 038h, 018h, 018h, 018h, 018h,
018h
 53189 00013489 00381818181818      <1>
 53190 00013490 18183C0000000000000- <1>    db  018h, 018h, 03ch, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0ech, 0feh, 0d6h, 0d6h,
0d6h
 53191 00013499 0000ECFED6D6D6      <1>
 53192 000134A0 C600000000000000000- <1>    db  0c6h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0dch, 066h, 066h, 066h, 066h, 066h,
000h
 53193 000134A9 DC666666666600      <1>
 53194 000134B0 0000000000000007CC6- <1>    db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h,
000h
 53195 000134B9 C6C6C67C000000      <1>
 53196 000134C0 0000000000DC666666- <1>     db  000h, 000h, 000h, 000h, 000h, 0dch, 066h, 066h, 066h, 07ch, 060h, 060h, 0f0h, 000h, 000h,
000h
 53197 000134C9 7C6060F0000000      <1>
 53198 000134D0 00000076CCCCCC7C0C- <1>     db  000h, 000h, 000h, 076h, 0cch, 0cch, 0cch, 07ch, 00ch, 00ch, 01eh, 000h, 000h, 000h, 000h,
000h
 53199 000134D9 0C1E0000000000      <1>
 53200 000134E0 00DC76666060F00000- <1>     db  000h, 0dch, 076h, 066h, 060h, 060h, 0f0h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
07ch
 53201 000134E9 0000000000007C      <1>
 53202 000134F0 C6701CC67C00000000- <1>     db  0c6h, 070h, 01ch, 0c6h, 07ch, 000h, 000h, 000h, 000h, 000h, 010h, 030h, 030h, 0fch, 030h,
030h
 53203 000134F9 00103030FC3030      <1>
 53204 00013500 30361C0000000000000- <1>    db  030h, 036h, 01ch, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0cch, 0cch, 0cch, 0cch,
0cch
 53205 00013509 0000CCCCCCCCCC      <1>
 53206 00013510 7600000000000000000- <1>    db  076h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 066h, 066h, 066h, 066h, 03ch, 018h,
000h
 53207 00013519 666666663C1800      <1>
 53208 00013520 000000000000000C6C6- <1>    db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 0c6h, 0c6h, 0d6h, 0d6h, 0feh, 06ch, 000h, 000h,
000h
 53209 00013529 D6D6FE6C000000      <1>
 53210 00013530 0000000000C66C3838- <1>     db  000h, 000h, 000h, 000h, 000h, 0c6h, 06ch, 038h, 038h, 06ch, 0c6h, 000h, 000h, 000h, 000h,
000h
 53211 00013539 6CC60000000000      <1>
 53212 00013540 000000C6C6C6C67E06- <1>     db  000h, 000h, 000h, 0c6h, 0c6h, 0c6h, 0c6h, 07eh, 006h, 00ch, 0f8h, 000h, 000h, 000h, 000h,
000h
 53213 00013549 0CF80000000000      <1>
 53214 00013550 00FECC183066FE0000- <1>     db  000h, 0feh, 0cch, 018h, 030h, 066h, 0feh, 000h, 000h, 000h, 000h, 000h, 00eh, 018h, 018h,
018h
 53215 00013559 0000000E181818      <1>
 53216 00013560 701818180E00000000- <1>     db  070h, 018h, 018h, 018h, 00eh, 000h, 000h, 000h, 000h, 000h, 018h, 018h, 018h, 018h, 000h,
018h
 53217 00013569 00181818180018      <1>
 53218 00013570 1818180000000000070- <1>    db  018h, 018h, 018h, 000h, 000h, 000h, 000h, 000h, 070h, 018h, 018h, 018h, 00eh, 018h, 018h,
018h
 53219 00013579 1818180E181818      <1>
```

```
 53220 00013580 70000000000076DC00- <1>     db  070h, 000h, 000h, 000h, 000h, 000h, 076h, 0dch, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 53221 00013589 00000000000000      <1>
 53222 00013590 00000000000010386C- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 010h, 038h, 06ch, 0c6h, 0c6h, 0feh, 000h, 000h, 000h,
000h
 53223 00013599 C6C6FE00000000      <1>
 53224 000135A0 00003C66C2C0C0C266- <1>     db  000h, 000h, 03ch, 066h, 0c2h, 0c0h, 0c0h, 0c2h, 066h, 03ch, 00ch, 006h, 07ch, 000h, 000h,
000h
 53225 000135A9 3C0C067C000000      <1>
 53226 000135B0 CCCC00CCCCCCCCCC76- <1>     db  0cch, 0cch, 000h, 0cch, 0cch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h, 00ch, 018h,
030h
 53227 000135B9 000000000C1830      <1>
 53228 000135C0 007CC6FEC0C67C0000- <1>     db  000h, 07ch, 0c6h, 0feh, 0c0h, 0c6h, 07ch, 000h, 000h, 000h, 000h, 010h, 038h, 06ch, 000h,
078h
 53229 000135C9 000010386C0078      <1>
 53230 000135D0 0C7CCCCC7600000000- <1>     db  00ch, 07ch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h, 000h, 0cch, 0cch, 000h, 078h, 00ch,
07ch
 53231 000135D9 00CCCC00780C7C      <1>
 53232 000135E0 CCCC76000000006030- <1>     db  0cch, 0cch, 076h, 000h, 000h, 000h, 000h, 060h, 030h, 018h, 000h, 078h, 00ch, 07ch, 0cch,
0cch
 53233 000135E9 1800780C7CCCCC      <1>
 53234 000135F0 7600000000386C3800- <1>     db  076h, 000h, 000h, 000h, 000h, 038h, 06ch, 038h, 000h, 078h, 00ch, 07ch, 0cch, 0cch, 076h,
000h
 53235 000135F9 780C7CCCCC7600      <1>
 53236 00013600 0000000000003C6660- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 03ch, 066h, 060h, 066h, 03ch, 00ch, 006h, 03ch, 000h,
000h
 53237 00013609 663C0C063C0000      <1>
 53238 00013610 0010386C007CC6FEC0- <1>     db  000h, 010h, 038h, 06ch, 000h, 07ch, 0c6h, 0feh, 0c0h, 0c6h, 07ch, 000h, 000h, 000h, 000h,
000h
 53239 00013619 C67C0000000000      <1>
 53240 00013620 CCCC007CC6FEC0C67C- <1>     db  0cch, 0cch, 000h, 07ch, 0c6h, 0feh, 0c0h, 0c6h, 07ch, 000h, 000h, 000h, 000h, 060h, 030h,
018h
 53241 00013629 00000000603018      <1>
 53242 00013630 007CC6FEC0C67C0000- <1>     db  000h, 07ch, 0c6h, 0feh, 0c0h, 0c6h, 07ch, 000h, 000h, 000h, 000h, 000h, 066h, 066h, 000h,
038h
 53243 00013639 00000066660038      <1>
 53244 00013640 181818183C00000000- <1>     db  018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h, 000h, 018h, 03ch, 066h, 000h, 038h, 018h,
018h
 53245 00013649 183C6600381818      <1>
 53246 00013650 18183C000000006030- <1>     db  018h, 018h, 03ch, 000h, 000h, 000h, 000h, 060h, 030h, 018h, 000h, 038h, 018h, 018h, 018h,
018h
 53247 00013659 18003818181818      <1>
 53248 00013660 3C00000000C6C61038- <1>     db  03ch, 000h, 000h, 000h, 000h, 0c6h, 0c6h, 010h, 038h, 06ch, 0c6h, 0c6h, 0feh, 0c6h, 0c6h,
000h
 53249 00013669 6CC6C6FEC6C600      <1>
 53250 00013670 0000386C3800386CC6- <1>     db  000h, 000h, 038h, 06ch, 038h, 000h, 038h, 06ch, 0c6h, 0c6h, 0feh, 0c6h, 0c6h, 000h, 000h,
000h
 53251 00013679 C6FEC6C6000000      <1>
 53252 00013680 18306000FE66607C60- <1>     db  018h, 030h, 060h, 000h, 0feh, 066h, 060h, 07ch, 060h, 066h, 0feh, 000h, 000h, 000h, 000h,
000h
 53253 00013689 66FE0000000000      <1>
 53254 00013690 0000CC76367ED8D86E- <1>     db  000h, 000h, 0cch, 076h, 036h, 07eh, 0d8h, 0d8h, 06eh, 000h, 000h, 000h, 000h, 000h, 03eh,
06ch
 53255 00013699 00000000003E6C      <1>
 53256 000136A0 CCCCFECCCCCCCE0000- <1>     db  0cch, 0cch, 0feh, 0cch, 0cch, 0cch, 0ceh, 000h, 000h, 000h, 000h, 010h, 038h, 06ch, 000h,
07ch
 53257 000136A9 000010386C007C      <1>
 53258 000136B0 C6C6C6C67C00000000- <1>     db  0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h, 000h, 0c6h, 0c6h, 000h, 07ch, 0c6h,
0c6h
 53259 000136B9 00C6C6007CC6C6      <1>
 53260 000136C0 C6C67C000000006030- <1>     db  0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h, 060h, 030h, 018h, 000h, 07ch, 0c6h, 0c6h, 0c6h,
0c6h
 53261 000136C9 18007CC6C6C6C6      <1>
 53262 000136D0 7C000000003078CC00- <1>     db  07ch, 000h, 000h, 000h, 000h, 030h, 078h, 0cch, 000h, 0cch, 0cch, 0cch, 0cch, 0cch, 076h,
000h
 53263 000136D9 CCCCCCCCCC7600      <1>
 53264 000136E0 00000060301800CCCC- <1>     db  000h, 000h, 000h, 060h, 030h, 018h, 000h, 0cch, 0cch, 0cch, 0cch, 0cch, 076h, 000h, 000h,
000h
 53265 000136E9 CCCCCC76000000      <1>
 53266 000136F0 0000C6C600C6C6C6C6- <1>     db  000h, 000h, 0c6h, 0c6h, 000h, 0c6h, 0c6h, 0c6h, 0c6h, 07eh, 006h, 00ch, 078h, 000h, 000h,
0c6h
 53267 000136F9 7E060C780000C6      <1>
 53268 00013700 C6386CC6C6C6C66C38- <1>     db  0c6h, 038h, 06ch, 0c6h, 0c6h, 0c6h, 0c6h, 06ch, 038h, 000h, 000h, 000h, 000h, 0c6h, 0c6h,
000h
 53269 00013709 00000000C6C600      <1>
 53270 00013710 C6C6C6C6C6C67C0000- <1>     db  0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h, 018h, 018h, 03ch, 066h,
060h
 53271 00013719 000018183C6660      <1>
 53272 00013720 60663C1818000000000- <1>    db  060h, 066h, 03ch, 018h, 018h, 000h, 000h, 000h, 000h, 038h, 06ch, 064h, 060h, 0f0h, 060h,
060h
 53273 00013729 386C6460F06060      <1>
 53274 00013730 60E6FC000000000066- <1>     db  060h, 0e6h, 0fch, 000h, 000h, 000h, 000h, 000h, 066h, 066h, 03ch, 018h, 07eh, 018h, 07eh,
018h
 53275 00013739 663C187E187E18      <1>
 53276 00013740 1800000000F8CCCCF8- <1>     db  018h, 000h, 000h, 000h, 000h, 0f8h, 0cch, 0cch, 0f8h, 0c4h, 0cch, 0deh, 0cch, 0cch, 0c6h,
000h
 53277 00013749 C4CCDECCCCC600      <1>
 53278 00013750 0000000E1B1818187E- <1>     db  000h, 000h, 000h, 00eh, 01bh, 018h, 018h, 018h, 07eh, 018h, 018h, 018h, 018h, 0d8h, 070h,
000h
 53279 00013759 18181818D87000      <1>
 53280 00013760 0018306000780C7CCC- <1>     db  000h, 018h, 030h, 060h, 000h, 078h, 00ch, 07ch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h,
00ch
 53281 00013769 CC76000000000C      <1>
 53282 00013770 183000381818183C- <1>     db  018h, 030h, 000h, 038h, 018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h, 000h, 018h, 030h,
060h
 53283 00013779 00000000183060      <1>
 53284 00013780 007CC6C6C6C67C0000- <1>     db  000h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h, 018h, 030h, 060h, 000h,
0cch
 53285 00013789 000018306000CC      <1>
 53286 00013790 CCCCCCCC7600000000- <1>     db  0cch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h, 000h, 076h, 0dch, 000h, 0dch, 066h,
066h
 53287 00013799 0076DC00DC6666      <1>
```

```
 53288 000137A0 66666600000076DC00- <1>     db  066h, 066h, 066h, 000h, 000h, 000h, 076h, 0dch, 000h, 0c6h, 0e6h, 0f6h, 0feh, 0deh, 0ceh,
0c6h
 53289 000137A9 C6E6F6FEDECEC6      <1>
 53290 000137B0 C6000000003C6C6C3E- <1>     db  0c6h, 000h, 000h, 000h, 000h, 03ch, 06ch, 06ch, 03eh, 000h, 07eh, 000h, 000h, 000h, 000h,
000h
 53291 000137B9 007E0000000000      <1>
 53292 000137C0 000000386C6C38007C- <1>     db  000h, 000h, 000h, 038h, 06ch, 06ch, 038h, 000h, 07ch, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 53293 000137C9 00000000000000      <1>
 53294 000137D0 0000303000303060C6- <1>     db  000h, 000h, 030h, 030h, 000h, 030h, 030h, 060h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h,
000h
 53295 000137D9 C67C0000000000      <1>
 53296 000137E0 00000000FEC0C0C000- <1>     db  000h, 000h, 000h, 000h, 0feh, 0c0h, 0c0h, 0c0h, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 53297 000137E9 00000000000000      <1>
 53298 000137F0 0000FE0606060000000- <1>    db  000h, 000h, 0feh, 006h, 006h, 006h, 000h, 000h, 000h, 000h, 000h, 0c0h, 0c0h, 0c6h, 0cch,
0d8h
 53299 000137F9 0000C0C0C6CCD8      <1>
 53300 00013800 3060DC860C183E0000- <1>     db  030h, 060h, 0dch, 086h, 00ch, 018h, 03eh, 000h, 000h, 0c0h, 0c0h, 0c6h, 0cch, 0d8h, 030h,
066h
 53301 00013809 C0C0C6CCD83066      <1>
 53302 00013810 CE9E3E0606000000018- <1>    db  0ceh, 09eh, 03eh, 006h, 006h, 000h, 000h, 000h, 018h, 018h, 000h, 018h, 018h, 03ch, 03ch,
03ch
 53303 00013819 180018183C3C3C      <1>
 53304 00013820 1800000000000000036- <1>    db  018h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 036h, 06ch, 0d8h, 06ch, 036h, 000h, 000h,
000h
 53305 00013829 6CD86C36000000      <1>
 53306 00013830 000000000000D86C36- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 0d8h, 06ch, 036h, 06ch, 0d8h, 000h, 000h, 000h, 000h,
000h
 53307 00013839 6CD80000000000      <1>
 53308 00013840 114414411441144411- <1>     db  011h, 044h, 011h, 044h, 011h, 044h, 011h, 044h, 011h, 044h, 011h, 044h, 011h, 044h, 055h,
0aah
 53309 00013849 441144114455AA      <1>
 53310 00013850 55AA55AA55AA55AA55- <1>     db  055h, 0aah, 055h, 0aah, 055h, 0aah, 055h, 0aah, 055h, 0aah, 055h, 0aah, 0ddh, 077h, 0ddh,
077h
 53311 00013859 AA55AADD77DD77      <1>
 53312 00013860 DD77DD77DD77DD77DD- <1>     db  0ddh, 077h, 0ddh, 077h, 0ddh, 077h, 0ddh, 077h, 0ddh, 077h, 018h, 018h, 018h, 018h, 018h,
018h
 53313 00013869 77181818181818      <1>
 53314 00013870 1818181818181818- <1>     db  018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h,
0f8h
 53315 00013879 181818181818F8      <1>
 53316 00013880 1818181818181818- <1>     db  018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 0f8h, 018h, 0f8h, 018h,
018h
 53317 00013889 1818F818F81818      <1>
 53318 00013890 1818181836363636- <1>     db  018h, 018h, 018h, 018h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 0f6h, 036h, 036h, 036h,
036h
 53319 00013899 3636F636363636      <1>
 53320 000138A0 3636000000000000- <1>     db  036h, 036h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0feh, 036h, 036h, 036h, 036h, 036h,
036h
 53321 000138A9 FE363636363636      <1>
 53322 000138B0 0000000000F818F818- <1>     db  000h, 000h, 000h, 000h, 000h, 0f8h, 018h, 0f8h, 018h, 018h, 018h, 018h, 018h, 018h, 036h,
036h
 53323 000138B9 18181818183636      <1>
 53324 000138C0 363636F606F6363636- <1>     db  036h, 036h, 036h, 0f6h, 006h, 0f6h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h,
036h
 53325 000138C9 36363636363636      <1>
 53326 000138D0 3636363636363636- <1>     db  036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 000h, 000h, 000h, 000h, 000h,
0feh
 53327 000138D9 360000000000FE      <1>
 53328 000138E0 06F636363636363636- <1>     db  006h, 0f6h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 0f6h, 006h,
0feh
 53329 000138E9 36363636F606FE      <1>
 53330 000138F0 0000000000000363636- <1>    db  000h, 000h, 000h, 000h, 000h, 000h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 0feh, 000h,
000h
 53331 000138F9 36363636FE0000      <1>
 53332 00013900 0000000001818181818- <1>    db  000h, 000h, 000h, 000h, 018h, 018h, 018h, 018h, 018h, 0f8h, 018h, 0f8h, 000h, 000h, 000h,
000h
 53333 00013909 F818F800000000      <1>
 53334 00013910 0000000000000000000- <1>    db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0f8h, 018h, 018h, 018h, 018h, 018h,
018h
 53335 00013919 F8181818181818      <1>
 53336 00013920 1818181818181F00- <1>     db  018h, 018h, 018h, 018h, 018h, 018h, 018h, 01fh, 000h, 000h, 000h, 000h, 000h, 000h, 018h,
018h
 53337 00013929 00000000001818      <1>
 53338 00013930 1818181818FF000000- <1>     db  018h, 018h, 018h, 018h, 018h, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 53339 00013939 00000000000000      <1>
 53340 00013940 000000FF1818181818- <1>     db  000h, 000h, 000h, 0ffh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h,
018h
 53341 00013949 18181818181818      <1>
 53342 00013950 181F18181818181800- <1>     db  018h, 01fh, 018h, 018h, 018h, 018h, 018h, 018h, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
0ffh
 53343 00013959 000000000000FF      <1>
 53344 00013960 0000000000000181818- <1>    db  000h, 000h, 000h, 000h, 000h, 000h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 0ffh, 018h,
018h
 53345 00013969 18181818FF1818      <1>
 53346 00013970 1818181818181818- <1>     db  018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 01fh, 018h, 01fh, 018h, 018h, 018h,
018h
 53347 00013979 1F181F18181818      <1>
 53348 00013980 1818363636363636- <1>     db  018h, 018h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 037h, 036h, 036h, 036h, 036h, 036h,
036h
 53349 00013989 37363636363636      <1>
 53350 00013990 3636363637303F00- <1>     db  036h, 036h, 036h, 036h, 036h, 037h, 030h, 03fh, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 53351 00013999 00000000000000      <1>
 53352 000139A0 0000003F3037363636- <1>     db  000h, 000h, 000h, 03fh, 030h, 037h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h,
036h
 53353 000139A9 36363636363636      <1>
 53354 000139B0 36F700FF0000000000- <1>     db  036h, 0f7h, 000h, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
0ffh
 53355 000139B9 000000000000FF      <1>
```

```
 53356 000139C0 00F736363636363636- <1>     db  000h, 0f7h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 037h, 030h,
037h
 53357 000139C9 36363636373037      <1>
 53358 000139D0 3636363636000000- <1>     db  036h, 036h, 036h, 036h, 036h, 036h, 000h, 000h, 000h, 000h, 000h, 0ffh, 000h, 0ffh, 000h,
000h
 53359 000139D9 0000FF00FF0000      <1>
 53360 000139E0 000000003636363636- <1>     db  000h, 000h, 000h, 000h, 036h, 036h, 036h, 036h, 036h, 0f7h, 000h, 0f7h, 036h, 036h, 036h,
036h
 53361 000139E9 F700F736363636      <1>
 53362 000139F0 36361818181818FF00- <1>     db  036h, 036h, 018h, 018h, 018h, 018h, 018h, 0ffh, 000h, 0ffh, 000h, 000h, 000h, 000h, 000h,
000h
 53363 000139F9 FF000000000000      <1>
 53364 00013A00 36363636363636FF00- <1>     db  036h, 036h, 036h, 036h, 036h, 036h, 036h, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 53365 00013A09 00000000000000      <1>
 53366 00013A10 000000FF00FF181818- <1>     db  000h, 000h, 000h, 0ffh, 000h, 0ffh, 018h, 018h, 018h, 018h, 018h, 018h, 000h, 000h, 000h,
000h
 53367 00013A19 18181800000000      <1>
 53368 00013A20 000000FF3636363636- <1>     db  000h, 000h, 000h, 0ffh, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h,
036h
 53369 00013A29 36363636363636      <1>
 53370 00013A30 363F00000000000018- <1>     db  036h, 03fh, 000h, 000h, 000h, 000h, 000h, 000h, 018h, 018h, 018h, 018h, 018h, 01fh, 018h,
01fh
 53371 00013A39 181818181F181F      <1>
 53372 00013A40 000000000000000000- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 01fh, 018h, 01fh, 018h,
018h
 53373 00013A49 00001F181F1818      <1>
 53374 00013A50 1818181800000000- <1>     db  018h, 018h, 018h, 018h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 03fh, 036h, 036h, 036h,
036h
 53375 00013A59 00003F36363636      <1>
 53376 00013A60 3636363636363636- <1>     db  036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 0ffh, 036h, 036h, 036h, 036h, 036h,
036h
 53377 00013A69 FF363636363636      <1>
 53378 00013A70 1818181818FF18FF18- <1>     db  018h, 018h, 018h, 018h, 018h, 0ffh, 018h, 0ffh, 018h, 018h, 018h, 018h, 018h, 018h, 018h,
018h
 53379 00013A79 18181818181818      <1>
 53380 00013A80 1818181818F8000000- <1>     db  018h, 018h, 018h, 018h, 018h, 0f8h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 53381 00013A89 00000000000000      <1>
 53382 00013A90 0000001F1818181818- <1>     db  000h, 000h, 000h, 01fh, 018h, 018h, 018h, 018h, 018h, 018h, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh,
0ffh
 53383 00013A99 18FFFFFFFFFFFF      <1>
 53384 00013AA0 FFFFFFFFFFFFFFFF00- <1>     db  0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
0ffh
 53385 00013AA9 000000000000FF      <1>
 53386 00013AB0 FFFFFFFFFFFF0F0F0- <1>     db  0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h,
0f0h
 53387 00013AB9 F0F0F0F0F0F0F0      <1>
 53388 00013AC0 F0F0F0F00F0F0F0F0F- <1>     db  0f0h, 0f0h, 0f0h, 0f0h, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh,
00fh
 53389 00013AC9 0F0F0F0F0F0F0F      <1>
 53390 00013AD0 0F0FFFFFFFFFFFFFFF- <1>     db  00fh, 00fh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 53391 00013AD9 00000000000000      <1>
 53392 00013AE0 000000000076DCD8D8- <1>     db  000h, 000h, 000h, 000h, 000h, 076h, 0dch, 0d8h, 0d8h, 0dch, 076h, 000h, 000h, 000h, 000h,
000h
 53393 00013AE9 DC760000000000      <1>
 53394 00013AF0 00007CC6FCC6C6FCC0- <1>     db  000h, 000h, 07ch, 0c6h, 0fch, 0c6h, 0c6h, 0fch, 0c0h, 0c0h, 040h, 000h, 000h, 000h, 0feh,
0c6h
 53395 00013AF9 C040000000FEC6      <1>
 53396 00013B00 C6C0C0C0C0C0C00000- <1>     db  0c6h, 0c0h, 0c0h, 0c0h, 0c0h, 0c0h, 0c0h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0feh,
06ch
 53397 00013B09 0000000000FE6C      <1>
 53398 00013B10 6C6C6C6C6C00000000- <1>     db  06ch, 06ch, 06ch, 06ch, 06ch, 000h, 000h, 000h, 000h, 000h, 0feh, 0c6h, 060h, 030h, 018h,
030h
 53399 00013B19 00FEC660301830      <1>
 53400 00013B20 60C6FE000000000000- <1>     db  060h, 0c6h, 0feh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 07eh, 0d8h, 0d8h, 0d8h,
0d8h
 53401 00013B29 00007ED8D8D8D8      <1>
 53402 00013B30 7000000000000000066- <1>     db  070h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 066h, 066h, 066h, 066h, 07ch, 060h, 060h,
0c0h
 53403 00013B39 6666667C6060C0      <1>
 53404 00013B40 00000000000076DC18- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 076h, 0dch, 018h, 018h, 018h, 018h, 018h, 000h, 000h,
000h
 53405 00013B49 18181818000000      <1>
 53406 00013B50 00007E183C6666663C- <1>     db  000h, 000h, 07eh, 018h, 03ch, 066h, 066h, 066h, 03ch, 018h, 07eh, 000h, 000h, 000h, 000h,
000h
 53407 00013B59 187E0000000000      <1>
 53408 00013B60 386CC6C6FEC6C66C38- <1>     db  038h, 06ch, 0c6h, 0c6h, 0feh, 0c6h, 0c6h, 06ch, 038h, 000h, 000h, 000h, 000h, 000h, 038h,
06ch
 53409 00013B69 00000000000386C     <1>
 53410 00013B70 C6C6C66C6C6CEE0000- <1>     db  0c6h, 0c6h, 0c6h, 06ch, 06ch, 06ch, 0eeh, 000h, 000h, 000h, 000h, 000h, 01eh, 030h, 018h,
00ch
 53411 00013B79 0000001E30180C      <1>
 53412 00013B80 3E6666663C00000000- <1>     db  03eh, 066h, 066h, 066h, 03ch, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 07eh, 0dbh,
0dbh
 53413 00013B89 000000007EDBDB      <1>
 53414 00013B90 7E0000000000000003- <1>     db  07eh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 003h, 006h, 07eh, 0dbh, 0dbh, 0f3h, 07eh,
060h
 53415 00013B99 067EDBDBF37E60      <1>
 53416 00013BA0 C000000000001C3060- <1>     db  0c0h, 000h, 000h, 000h, 000h, 000h, 01ch, 030h, 060h, 060h, 07ch, 060h, 060h, 030h, 01ch,
000h
 53417 00013BA9 607C6060301C00      <1>
 53418 00013BB0 00000000007CC6C6C6- <1>     db  000h, 000h, 000h, 000h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 000h, 000h,
000h
 53419 00013BB9 C6C6C6C6000000      <1>
 53420 00013BC0 000000FE0000FE0000- <1>     db  000h, 000h, 000h, 0feh, 000h, 000h, 0feh, 000h, 000h, 0feh, 000h, 000h, 000h, 000h, 000h,
000h
 53421 00013BC9 FE000000000000      <1>
 53422 00013BD0 0018187E18180000FF- <1>     db  000h, 018h, 018h, 07eh, 018h, 018h, 000h, 000h, 0ffh, 000h, 000h, 000h, 000h, 000h, 030h,
018h
 53423 00013BD9 00000000003018      <1>
```

```
 53424 00013BE0 0C060C1830007E0000- <1>     db  00ch, 006h, 00ch, 018h, 030h, 000h, 07eh, 000h, 000h, 000h, 000h, 000h, 00ch, 018h, 030h,
060h
 53425 00013BE9 0000000C183060      <1>
 53426 00013BF0 30180C007E00000000- <1>     db  030h, 018h, 00ch, 000h, 07eh, 000h, 000h, 000h, 000h, 000h, 00eh, 01bh, 01bh, 018h, 018h,
018h
 53427 00013BF9 000E1B1B181818      <1>
 53428 00013C00 181818181818181818- <1>     db  018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 0d8h,
0d8h
 53429 00013C09 1818181818D8D8      <1>
 53430 00013C10 700000000000001818- <1>     db  070h, 000h, 000h, 000h, 000h, 000h, 000h, 018h, 018h, 000h, 07eh, 000h, 018h, 018h, 000h,
000h
 53431 00013C19 007E0018180000      <1>
 53432 00013C20 00000000000076DC00- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 076h, 0dch, 000h, 076h, 0dch, 000h, 000h, 000h, 000h,
000h
 53433 00013C29 76DC0000000000      <1>
 53434 00013C30 00386C6C3800000000- <1>     db  000h, 038h, 06ch, 06ch, 038h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 53435 00013C39 00000000000000      <1>
 53436 00013C40 0000000018180000- <1>     db  000h, 000h, 000h, 000h, 018h, 018h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 53437 00013C49 00000000000000      <1>
 53438 00013C50 0000001800000000- <1>     db  000h, 000h, 000h, 018h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 00fh, 00ch, 00ch, 00ch,
00ch
 53439 00013C59 00000F0C0C0C0C      <1>
 53440 00013C60 0CEC6C3C1C00000000- <1>     db  00ch, 0ech, 06ch, 03ch, 01ch, 000h, 000h, 000h, 000h, 0d8h, 06ch, 06ch, 06ch, 06ch, 06ch,
000h
 53441 00013C69 D86C6C6C6C6C00      <1>
 53442 00013C70 0000000000000070D8- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 070h, 0d8h, 030h, 060h, 0c8h, 0f8h, 000h, 000h,
000h
 53443 00013C79 3060C8F8000000      <1>
 53444 00013C80 000000000000000007C- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 07ch, 07ch, 07ch, 07ch, 07ch, 07ch, 000h,
000h
 53445 00013C89 7C7C7C7C7C0000      <1>
 53446 00013C90 0000000000000000000- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 53447 00013C99 00000000000000      <1>
 53448                              <1> vgafont16:
 53449 00013CA0 0000000000000000000- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 53450 00013CA9 00000000000000      <1>
 53451 00013CB0 00007E81A58181BD99- <1>     db  000h, 000h, 07eh, 081h, 0a5h, 081h, 081h, 0bdh, 099h, 081h, 081h, 07eh, 000h, 000h, 000h,
000h
 53452 00013CB9 81817E00000000      <1>
 53453 00013CC0 00007EFFDBFFFFC3E7- <1>     db  000h, 000h, 07eh, 0ffh, 0dbh, 0ffh, 0ffh, 0c3h, 0e7h, 0ffh, 0ffh, 07eh, 000h, 000h, 000h,
000h
 53454 00013CC9 FFFF7E00000000      <1>
 53455 00013CD0 000000006CFEFEFEFE- <1>     db  000h, 000h, 000h, 000h, 06ch, 0feh, 0feh, 0feh, 0feh, 07ch, 038h, 010h, 000h, 000h, 000h,
000h
 53456 00013CD9 7C381000000000      <1>
 53457 00013CE0 0000000010387CFE7C- <1>     db  000h, 000h, 000h, 000h, 010h, 038h, 07ch, 0feh, 07ch, 038h, 010h, 000h, 000h, 000h, 000h,
000h
 53458 00013CE9 38100000000000      <1>
 53459 00013CF0 000000183C3CE7E7E7- <1>     db  000h, 000h, 000h, 018h, 03ch, 03ch, 0e7h, 0e7h, 0e7h, 018h, 018h, 03ch, 000h, 000h, 000h,
000h
 53460 00013CF9 18183C00000000      <1>
 53461 00013D00 000000183C7EFFFF7E- <1>     db  000h, 000h, 000h, 018h, 03ch, 07eh, 0ffh, 0ffh, 07eh, 018h, 018h, 03ch, 000h, 000h, 000h,
000h
 53462 00013D09 18183C00000000      <1>
 53463 00013D10 000000000000183C3C- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 018h, 03ch, 03ch, 018h, 000h, 000h, 000h, 000h, 000h,
000h
 53464 00013D19 18000000000000      <1>
 53465 00013D20 FFFFFFFFFFFFE7C3C3- <1>     db  0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0e7h, 0c3h, 0c3h, 0e7h, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh,
0ffh
 53466 00013D29 E7FFFFFFFFFFFF      <1>
 53467 00013D30 00000000003C664242- <1>     db  000h, 000h, 000h, 000h, 000h, 03ch, 066h, 042h, 042h, 066h, 03ch, 000h, 000h, 000h, 000h,
000h
 53468 00013D39 663C0000000000      <1>
 53469 00013D40 FFFFFFFFFFC399BDBD- <1>     db  0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0c3h, 099h, 0bdh, 0bdh, 099h, 0c3h, 0ffh, 0ffh, 0ffh, 0ffh,
0ffh
 53470 00013D49 99C3FFFFFFFFFF      <1>
 53471 00013D50 00001E0E1A3278CCCC- <1>     db  000h, 000h, 01eh, 00eh, 01ah, 032h, 078h, 0cch, 0cch, 0cch, 0cch, 078h, 000h, 000h, 000h,
000h
 53472 00013D59 CCCC7800000000      <1>
 53473 00013D60 00003C666666663C18- <1>     db  000h, 000h, 03ch, 066h, 066h, 066h, 066h, 03ch, 018h, 07eh, 018h, 018h, 000h, 000h, 000h,
000h
 53474 00013D69 7E181800000000      <1>
 53475 00013D70 00003F333F30303030- <1>     db  000h, 000h, 03fh, 033h, 03fh, 030h, 030h, 030h, 030h, 070h, 0f0h, 0e0h, 000h, 000h, 000h,
000h
 53476 00013D79 70F0E000000000      <1>
 53477 00013D80 00007F637F63636363- <1>     db  000h, 000h, 07fh, 063h, 07fh, 063h, 063h, 063h, 063h, 067h, 0e7h, 0e6h, 0c0h, 000h, 000h,
000h
 53478 00013D89 67E7E6C0000000      <1>
 53479 00013D90 0000001818DB3CE73C- <1>     db  000h, 000h, 000h, 018h, 018h, 0dbh, 03ch, 0e7h, 03ch, 0dbh, 018h, 018h, 000h, 000h, 000h,
000h
 53480 00013D99 DB181800000000      <1>
 53481 00013DA0 0080C0E0F0F8FEF8F0- <1>     db  000h, 080h, 0c0h, 0e0h, 0f0h, 0f8h, 0feh, 0f8h, 0f0h, 0e0h, 0c0h, 080h, 000h, 000h, 000h,
000h
 53482 00013DA9 E0C08000000000      <1>
 53483 00013DB0 0002060E1E3EFE3E1E- <1>     db  000h, 002h, 006h, 00eh, 01eh, 03eh, 0feh, 03eh, 01eh, 00eh, 006h, 002h, 000h, 000h, 000h,
000h
 53484 00013DB9 0E060200000000      <1>
 53485 00013DC0 0000183C7E1818187E- <1>     db  000h, 000h, 018h, 03ch, 07eh, 018h, 018h, 018h, 07eh, 03ch, 018h, 000h, 000h, 000h, 000h,
000h
 53486 00013DC9 3C180000000000      <1>
 53487 00013DD0 000066666666666666- <1>     db  000h, 000h, 066h, 066h, 066h, 066h, 066h, 066h, 066h, 000h, 066h, 066h, 000h, 000h, 000h,
000h
 53488 00013DD9 00666600000000      <1>
 53489 00013DE0 00007FDBDBDB7B1B1B- <1>     db  000h, 000h, 07fh, 0dbh, 0dbh, 0dbh, 07bh, 01bh, 01bh, 01bh, 01bh, 01bh, 000h, 000h, 000h,
000h
 53490 00013DE9 1B1B1B00000000      <1>
 53491 00013DF0 007CC660386CC6C66C- <1>     db  000h, 07ch, 0c6h, 060h, 038h, 06ch, 0c6h, 0c6h, 06ch, 038h, 00ch, 0c6h, 07ch, 000h, 000h,
000h
 53492 00013DF9 380CC67C000000      <1>
```

```
 53493 00013E00 0000000000000000FE- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0feh, 0feh, 0feh, 0feh, 000h, 000h, 000h,
000h
 53494 00013E09 FEFEFE00000000      <1>
 53495 00013E10 0000183C7E1818187E- <1>     db  000h, 000h, 018h, 03ch, 07eh, 018h, 018h, 018h, 07eh, 03ch, 018h, 07eh, 000h, 000h, 000h,
000h
 53496 00013E19 3C187E00000000      <1>
 53497 00013E20 0000183C7E18181818- <1>     db  000h, 000h, 018h, 03ch, 07eh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 000h, 000h, 000h,
000h
 53498 00013E29 18181800000000      <1>
 53499 00013E30 000018181818181818- <1>     db  000h, 000h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 07eh, 03ch, 018h, 000h, 000h, 000h,
000h
 53500 00013E39 7E3C1800000000      <1>
 53501 00013E40 0000000000180CFE0C- <1>     db  000h, 000h, 000h, 000h, 000h, 018h, 00ch, 0feh, 00ch, 018h, 000h, 000h, 000h, 000h, 000h,
000h
 53502 00013E49 18000000000000      <1>
 53503 00013E50 000000000003060FE60- <1>     db  000h, 000h, 000h, 000h, 000h, 030h, 060h, 0feh, 060h, 030h, 000h, 000h, 000h, 000h, 000h,
000h
 53504 00013E59 30000000000000      <1>
 53505 00013E60 0000000000000C0C0C0- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 0c0h, 0c0h, 0c0h, 0feh, 000h, 000h, 000h, 000h, 000h,
000h
 53506 00013E69 FE000000000000      <1>
 53507 00013E70 00000000002466FF66- <1>     db  000h, 000h, 000h, 000h, 000h, 024h, 066h, 0ffh, 066h, 024h, 000h, 000h, 000h, 000h, 000h,
000h
 53508 00013E79 24000000000000      <1>
 53509 00013E80 000000001038387C7C- <1>     db  000h, 000h, 000h, 000h, 010h, 038h, 038h, 07ch, 07ch, 0feh, 0feh, 000h, 000h, 000h, 000h,
000h
 53510 00013E89 FEFE0000000000      <1>
 53511 00013E90 00000000FEFE7C7C38- <1>     db  000h, 000h, 000h, 000h, 0feh, 0feh, 07ch, 07ch, 038h, 038h, 010h, 000h, 000h, 000h, 000h,
000h
 53512 00013E99 38100000000000      <1>
 53513 00013EA0 0000000000000000000- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 53514 00013EA9 00000000000000      <1>
 53515 00013EB0 0000183C3C3C181818- <1>     db  000h, 000h, 018h, 03ch, 03ch, 03ch, 018h, 018h, 018h, 000h, 018h, 018h, 000h, 000h, 000h,
000h
 53516 00013EB9 00181800000000      <1>
 53517 00013EC0 006666662400000000- <1>     db  000h, 066h, 066h, 066h, 024h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 53518 00013EC9 00000000000000      <1>
 53519 00013ED0 0000006C6CFE6C6C6C- <1>     db  000h, 000h, 000h, 06ch, 06ch, 0feh, 06ch, 06ch, 06ch, 0feh, 06ch, 06ch, 000h, 000h, 000h,
000h
 53520 00013ED9 FE6C6C00000000      <1>
 53521 00013EE0 18187CC6C2C07C0606- <1>     db  018h, 018h, 07ch, 0c6h, 0c2h, 0c0h, 07ch, 006h, 006h, 086h, 0c6h, 07ch, 018h, 018h, 000h,
000h
 53522 00013EE9 86C67C18180000      <1>
 53523 00013EF0 00000000C2C60C1830- <1>     db  000h, 000h, 000h, 000h, 0c2h, 0c6h, 00ch, 018h, 030h, 060h, 0c6h, 086h, 000h, 000h, 000h,
000h
 53524 00013EF9 60C68600000000      <1>
 53525 00013F00 0000386C6C3876DCCC- <1>     db  000h, 000h, 038h, 06ch, 06ch, 038h, 076h, 0dch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h,
000h
 53526 00013F09 CCCC7600000000      <1>
 53527 00013F10 0030303060000000000- <1>     db  000h, 030h, 030h, 030h, 060h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 53528 00013F19 00000000000000      <1>
 53529 00013F20 00000C183030303030- <1>     db  000h, 000h, 00ch, 018h, 030h, 030h, 030h, 030h, 030h, 030h, 018h, 00ch, 000h, 000h, 000h,
000h
 53530 00013F29 30180C00000000      <1>
 53531 00013F30 000030180C0C0C0C0C- <1>     db  000h, 000h, 030h, 018h, 00ch, 00ch, 00ch, 00ch, 00ch, 00ch, 018h, 030h, 000h, 000h, 000h,
000h
 53532 00013F39 0C183000000000      <1>
 53533 00013F40 0000000000663CFF3C- <1>     db  000h, 000h, 000h, 000h, 000h, 066h, 03ch, 0ffh, 03ch, 066h, 000h, 000h, 000h, 000h, 000h,
000h
 53534 00013F49 66000000000000      <1>
 53535 00013F50 000000000018187E18- <1>     db  000h, 000h, 000h, 000h, 000h, 018h, 018h, 07eh, 018h, 018h, 000h, 000h, 000h, 000h, 000h,
000h
 53536 00013F59 18000000000000      <1>
 53537 00013F60 0000000000000000000- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 018h, 018h, 018h, 030h, 000h, 000h,
000h
 53538 00013F69 18181830000000      <1>
 53539 00013F70 00000000000000FE00- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 0feh, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 53540 00013F79 00000000000000      <1>
 53541 00013F80 0000000000000000000- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 018h, 018h, 000h, 000h, 000h,
000h
 53542 00013F89 00181800000000      <1>
 53543 00013F90 0000000002060C1830- <1>     db  000h, 000h, 000h, 000h, 002h, 006h, 00ch, 018h, 030h, 060h, 0c0h, 080h, 000h, 000h, 000h,
000h
 53544 00013F99 60C08000000000      <1>
 53545 00013FA0 00003C66C3C3DBDBC3- <1>     db  000h, 000h, 03ch, 066h, 0c3h, 0c3h, 0dbh, 0dbh, 0c3h, 0c3h, 066h, 03ch, 000h, 000h, 000h,
000h
 53546 00013FA9 C3663C00000000      <1>
 53547 00013FB0 000018387818181818- <1>     db  000h, 000h, 018h, 038h, 078h, 018h, 018h, 018h, 018h, 018h, 018h, 07eh, 000h, 000h, 000h,
000h
 53548 00013FB9 18187E00000000      <1>
 53549 00013FC0 00007CC6060C183060- <1>     db  000h, 000h, 07ch, 0c6h, 006h, 00ch, 018h, 030h, 060h, 0c0h, 0c6h, 0feh, 000h, 000h, 000h,
000h
 53550 00013FC9 C0C6FE00000000      <1>
 53551 00013FD0 00007CC606063C0606- <1>     db  000h, 000h, 07ch, 0c6h, 006h, 006h, 03ch, 006h, 006h, 006h, 0c6h, 07ch, 000h, 000h, 000h,
000h
 53552 00013FD9 06C67C00000000      <1>
 53553 00013FE0 00000C1C3C6CCCFE0C- <1>     db  000h, 000h, 00ch, 01ch, 03ch, 06ch, 0cch, 0feh, 00ch, 00ch, 00ch, 01eh, 000h, 000h, 000h,
000h
 53554 00013FE9 0C0C1E00000000      <1>
 53555 00013FF0 0000FEC0C0C0FC0606- <1>     db  000h, 000h, 0feh, 0c0h, 0c0h, 0c0h, 0fch, 006h, 006h, 006h, 0c6h, 07ch, 000h, 000h, 000h,
000h
 53556 00013FF9 06C67C00000000      <1>
 53557 00014000 00003860C0C0FCC6C6- <1>     db  000h, 000h, 038h, 060h, 0c0h, 0c0h, 0fch, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h,
000h
 53558 00014009 C6C67C00000000      <1>
 53559 00014010 0000FEC606060C1830- <1>     db  000h, 000h, 0feh, 0c6h, 006h, 006h, 00ch, 018h, 030h, 030h, 030h, 030h, 000h, 000h, 000h,
000h
 53560 00014019 30303000000000      <1>
```

```
 53561 00014020 00007CC6C6C67CC6C6- <1>     db  000h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h,
000h
 53562 00014029 C6C67C00000000      <1>
 53563 00014030 00007CC6C6C67E0606- <1>     db  000h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 07eh, 006h, 006h, 006h, 00ch, 078h, 000h, 000h, 000h,
000h
 53564 00014039 060C7800000000      <1>
 53565 00014040 000000001818000000- <1>     db  000h, 000h, 000h, 000h, 018h, 018h, 000h, 000h, 000h, 018h, 018h, 000h, 000h, 000h, 000h,
000h
 53566 00014049 18180000000000      <1>
 53567 00014050 000000001818000000- <1>     db  000h, 000h, 000h, 000h, 018h, 018h, 000h, 000h, 000h, 018h, 018h, 030h, 000h, 000h, 000h,
000h
 53568 00014059 18183000000000      <1>
 53569 00014060 000000060C18306030- <1>     db  000h, 000h, 000h, 006h, 00ch, 018h, 030h, 060h, 030h, 018h, 00ch, 006h, 000h, 000h, 000h,
000h
 53570 00014069 180C0600000000      <1>
 53571 00014070 00000000007E00007E- <1>     db  000h, 000h, 000h, 000h, 000h, 07eh, 000h, 000h, 07eh, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 53572 00014079 00000000000000      <1>
 53573 00014080 0000006030180C060C- <1>     db  000h, 000h, 000h, 060h, 030h, 018h, 00ch, 006h, 00ch, 018h, 030h, 060h, 000h, 000h, 000h,
000h
 53574 00014089 18306000000000      <1>
 53575 00014090 00007CC6C60C181818- <1>     db  000h, 000h, 07ch, 0c6h, 0c6h, 00ch, 018h, 018h, 018h, 000h, 018h, 018h, 000h, 000h, 000h,
000h
 53576 00014099 00181800000000      <1>
 53577 000140A0 0000007CC6C6DEDEDE- <1>     db  000h, 000h, 000h, 07ch, 0c6h, 0c6h, 0deh, 0deh, 0deh, 0dch, 0c0h, 07ch, 000h, 000h, 000h,
000h
 53578 000140A9 DCC07C00000000      <1>
 53579 000140B0 000010386CC6C6FEC6- <1>     db  000h, 000h, 010h, 038h, 06ch, 0c6h, 0c6h, 0feh, 0c6h, 0c6h, 0c6h, 0c6h, 000h, 000h, 000h,
000h
 53580 000140B9 C6C6C600000000      <1>
 53581 000140C0 0000FC6666667C6666- <1>     db  000h, 000h, 0fch, 066h, 066h, 066h, 07ch, 066h, 066h, 066h, 066h, 0fch, 000h, 000h, 000h,
000h
 53582 000140C9 6666FC00000000      <1>
 53583 000140D0 00003C66C2C0C0C0C0- <1>     db  000h, 000h, 03ch, 066h, 0c2h, 0c0h, 0c0h, 0c0h, 0c0h, 0c2h, 066h, 03ch, 000h, 000h, 000h,
000h
 53584 000140D9 C2663C00000000      <1>
 53585 000140E0 0000F86C6666666666- <1>     db  000h, 000h, 0f8h, 06ch, 066h, 066h, 066h, 066h, 066h, 066h, 06ch, 0f8h, 000h, 000h, 000h,
000h
 53586 000140E9 666CF800000000      <1>
 53587 000140F0 0000FE666268786860- <1>     db  000h, 000h, 0feh, 066h, 062h, 068h, 078h, 068h, 060h, 062h, 066h, 0feh, 000h, 000h, 000h,
000h
 53588 000140F9 6266FE00000000      <1>
 53589 00014100 0000FE666268786860- <1>     db  000h, 000h, 0feh, 066h, 062h, 068h, 078h, 068h, 060h, 060h, 060h, 0f0h, 000h, 000h, 000h,
000h
 53590 00014109 6060F000000000      <1>
 53591 00014110 00003C66C2C0C0DEC6- <1>     db  000h, 000h, 03ch, 066h, 0c2h, 0c0h, 0c0h, 0deh, 0c6h, 0c6h, 066h, 03ah, 000h, 000h, 000h,
000h
 53592 00014119 C6663A00000000      <1>
 53593 00014120 0000C6C6C6C6FEC6C6- <1>     db  000h, 000h, 0c6h, 0c6h, 0c6h, 0c6h, 0feh, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 000h, 000h, 000h,
000h
 53594 00014129 C6C6C600000000      <1>
 53595 00014130 00003C181818181818- <1>     db  000h, 000h, 03ch, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h,
000h
 53596 00014139 18183C00000000      <1>
 53597 00014140 00001E0C0C0C0C0CCC- <1>     db  000h, 000h, 01eh, 00ch, 00ch, 00ch, 00ch, 00ch, 0cch, 0cch, 0cch, 078h, 000h, 000h, 000h,
000h
 53598 00014149 CCCC7800000000      <1>
 53599 00014150 0000E666666C78786C- <1>     db  000h, 000h, 0e6h, 066h, 066h, 06ch, 078h, 078h, 06ch, 066h, 066h, 0e6h, 000h, 000h, 000h,
000h
 53600 00014159 6666E600000000      <1>
 53601 00014160 0000F06060606060- <1>     db  000h, 000h, 0f0h, 060h, 060h, 060h, 060h, 060h, 060h, 062h, 066h, 0feh, 000h, 000h, 000h,
000h
 53602 00014169 6266FE00000000      <1>
 53603 00014170 0000C3E7FFFFDBC3C3- <1>     db  000h, 000h, 0c3h, 0e7h, 0ffh, 0ffh, 0dbh, 0c3h, 0c3h, 0c3h, 0c3h, 0c3h, 000h, 000h, 000h,
000h
 53604 00014179 C3C3C300000000      <1>
 53605 00014180 0000C6E6F6FEDECEC6- <1>     db  000h, 000h, 0c6h, 0e6h, 0f6h, 0feh, 0deh, 0ceh, 0c6h, 0c6h, 0c6h, 0c6h, 000h, 000h, 000h,
000h
 53606 00014189 C6C6C600000000      <1>
 53607 00014190 00007CC6C6C6C6C6C6- <1>     db  000h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h,
000h
 53608 00014199 C6C67C00000000      <1>
 53609 000141A0 0000FC6666667C6060- <1>     db  000h, 000h, 0fch, 066h, 066h, 066h, 07ch, 060h, 060h, 060h, 060h, 0f0h, 000h, 000h, 000h,
000h
 53610 000141A9 6060F000000000      <1>
 53611 000141B0 00007CC6C6C6C6C6C6- <1>     db  000h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0d6h, 0deh, 07ch, 00ch, 00eh, 000h,
000h
 53612 000141B9 D6DE7C0C0E0000      <1>
 53613 000141C0 0000FC6666667C6C66- <1>     db  000h, 000h, 0fch, 066h, 066h, 066h, 07ch, 06ch, 066h, 066h, 066h, 0e6h, 000h, 000h, 000h,
000h
 53614 000141C9 6666E600000000      <1>
 53615 000141D0 00007CC6C660380C06- <1>     db  000h, 000h, 07ch, 0c6h, 0c6h, 060h, 038h, 00ch, 006h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h,
000h
 53616 000141D9 C6C67C00000000      <1>
 53617 000141E0 0000FFDB9918181818- <1>     db  000h, 000h, 0ffh, 0dbh, 099h, 018h, 018h, 018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h,
000h
 53618 000141E9 18183C00000000      <1>
 53619 000141F0 0000C6C6C6C6C6C6C6- <1>     db  000h, 000h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h,
000h
 53620 000141F9 C6C67C00000000      <1>
 53621 00014200 0000C3C3C3C3C3C3C3- <1>     db  000h, 000h, 0c3h, 0c3h, 0c3h, 0c3h, 0c3h, 0c3h, 0c3h, 066h, 03ch, 018h, 000h, 000h, 000h,
000h
 53622 00014209 663C1800000000      <1>
 53623 00014210 0000C3C3C3C3C3DBDB- <1>     db  000h, 000h, 0c3h, 0c3h, 0c3h, 0c3h, 0c3h, 0dbh, 0dbh, 0ffh, 066h, 066h, 000h, 000h, 000h,
000h
 53624 00014219 FF666600000000      <1>
 53625 00014220 0000C3C3663C18183C- <1>     db  000h, 000h, 0c3h, 0c3h, 066h, 03ch, 018h, 018h, 03ch, 066h, 0c3h, 0c3h, 000h, 000h, 000h,
000h
 53626 00014229 66C3C300000000      <1>
 53627 00014230 0000C3C3C3663C1818- <1>     db  000h, 000h, 0c3h, 0c3h, 0c3h, 066h, 03ch, 018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h,
000h
 53628 00014239 18183C00000000      <1>
```

```
 53629 00014240 0000FFC3860C183060- <1>     db  000h, 000h, 0ffh, 0c3h, 086h, 00ch, 018h, 030h, 060h, 0c1h, 0c3h, 0ffh, 000h, 000h, 000h,
000h
 53630 00014249 C1C3FF00000000      <1>
 53631 00014250 00003C303030303030- <1>     db  000h, 000h, 03ch, 030h, 030h, 030h, 030h, 030h, 030h, 030h, 030h, 03ch, 000h, 000h, 000h,
000h
 53632 00014259 30303C00000000      <1>
 53633 00014260 00000080C0E070381C- <1>     db  000h, 000h, 000h, 080h, 0c0h, 0e0h, 070h, 038h, 01ch, 00eh, 006h, 002h, 000h, 000h, 000h,
000h
 53634 00014269 0E060200000000      <1>
 53635 00014270 00003C0C0C0C0C0C0C- <1>     db  000h, 000h, 03ch, 00ch, 00ch, 00ch, 00ch, 00ch, 00ch, 00ch, 00ch, 03ch, 000h, 000h, 000h,
000h
 53636 00014279 0C0C3C00000000      <1>
 53637 00014280 10386CC600000000000- <1>     db  010h, 038h, 06ch, 0c6h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 53638 00014289 00000000000000      <1>
 53639 00014290 0000000000000000000- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0ffh, 000h,
000h
 53640 00014299 00000000FF0000      <1>
 53641 000142A0 3030180000000000000- <1>     db  030h, 030h, 018h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 53642 000142A9 00000000000000      <1>
 53643 000142B0 0000000000780C7CCC- <1>     db  000h, 000h, 000h, 000h, 000h, 078h, 00ch, 07ch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h,
000h
 53644 000142B9 CCCC7600000000      <1>
 53645 000142C0 0000E06060786C6666- <1>     db  000h, 000h, 0e0h, 060h, 060h, 078h, 06ch, 066h, 066h, 066h, 066h, 07ch, 000h, 000h, 000h,
000h
 53646 000142C9 66667C00000000      <1>
 53647 000142D0 00000000007CC6C0C0- <1>     db  000h, 000h, 000h, 000h, 000h, 07ch, 0c6h, 0c0h, 0c0h, 0c0h, 0c6h, 07ch, 000h, 000h, 000h,
000h
 53648 000142D9 C0C67C00000000      <1>
 53649 000142E0 00001C0C0C3C6CCCCC- <1>     db  000h, 000h, 01ch, 00ch, 00ch, 03ch, 06ch, 0cch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h,
000h
 53650 000142E9 CCCC7600000000      <1>
 53651 000142F0 00000000007CC6FEC0- <1>     db  000h, 000h, 000h, 000h, 000h, 07ch, 0c6h, 0feh, 0c0h, 0c0h, 0c6h, 07ch, 000h, 000h, 000h,
000h
 53652 000142F9 C0C67C00000000      <1>
 53653 00014300 0000386C6460F06060- <1>     db  000h, 000h, 038h, 06ch, 064h, 060h, 0f0h, 060h, 060h, 060h, 060h, 0f0h, 000h, 000h, 000h,
000h
 53654 00014309 6060F000000000      <1>
 53655 00014310 000000000076CCCCCC- <1>     db  000h, 000h, 000h, 000h, 000h, 076h, 0cch, 0cch, 0cch, 0cch, 0cch, 07ch, 00ch, 0cch, 078h,
000h
 53656 00014319 CCCC7C0CCC7800      <1>
 53657 00014320 0000E060606C766666- <1>     db  000h, 000h, 0e0h, 060h, 060h, 06ch, 076h, 066h, 066h, 066h, 066h, 0e6h, 000h, 000h, 000h,
000h
 53658 00014329 6666E600000000      <1>
 53659 00014330 000018180038181818- <1>     db  000h, 000h, 018h, 018h, 000h, 038h, 018h, 018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h,
000h
 53660 00014339 18183C00000000      <1>
 53661 00014340 00000606000E060606- <1>     db  000h, 000h, 006h, 006h, 000h, 00eh, 006h, 006h, 006h, 006h, 006h, 006h, 066h, 066h, 03ch,
000h
 53662 00014349 06060666663C00      <1>
 53663 00014350 0000E06060666C7878- <1>     db  000h, 000h, 0e0h, 060h, 060h, 066h, 06ch, 078h, 078h, 06ch, 066h, 0e6h, 000h, 000h, 000h,
000h
 53664 00014359 6C66E600000000      <1>
 53665 00014360 000038181818181818- <1>     db  000h, 000h, 038h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h,
000h
 53666 00014369 18183C00000000      <1>
 53667 00014370 0000000000E6FFDBDB- <1>     db  000h, 000h, 000h, 000h, 000h, 0e6h, 0ffh, 0dbh, 0dbh, 0dbh, 0dbh, 0dbh, 000h, 000h, 000h,
000h
 53668 00014379 DBDBDB00000000      <1>
 53669 00014380 0000000000DC666666- <1>     db  000h, 000h, 000h, 000h, 000h, 0dch, 066h, 066h, 066h, 066h, 066h, 066h, 000h, 000h, 000h,
000h
 53670 00014389 66666600000000      <1>
 53671 00014390 00000000007CC6C6C6- <1>     db  000h, 000h, 000h, 000h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h,
000h
 53672 00014399 C6C67C00000000      <1>
 53673 000143A0 0000000000DC666666- <1>     db  000h, 000h, 000h, 000h, 000h, 0dch, 066h, 066h, 066h, 066h, 066h, 07ch, 060h, 060h, 0f0h,
000h
 53674 000143A9 66667C6060F000      <1>
 53675 000143B0 000000000076CCCCCC- <1>     db  000h, 000h, 000h, 000h, 000h, 076h, 0cch, 0cch, 0cch, 0cch, 0cch, 07ch, 00ch, 00ch, 01eh,
000h
 53676 000143B9 CCCC7C0C0C1E00      <1>
 53677 000143C0 0000000000DC766660- <1>     db  000h, 000h, 000h, 000h, 000h, 0dch, 076h, 066h, 060h, 060h, 060h, 0f0h, 000h, 000h, 000h,
000h
 53678 000143C9 6060F000000000      <1>
 53679 000143D0 00000000007CC66038- <1>     db  000h, 000h, 000h, 000h, 000h, 07ch, 0c6h, 060h, 038h, 00ch, 0c6h, 07ch, 000h, 000h, 000h,
000h
 53680 000143D9 0CC67C00000000      <1>
 53681 000143E0 0000103030FC303030- <1>     db  000h, 000h, 010h, 030h, 030h, 0fch, 030h, 030h, 030h, 030h, 036h, 01ch, 000h, 000h, 000h,
000h
 53682 000143E9 30361C00000000      <1>
 53683 000143F0 0000000000CCCCCCCC- <1>     db  000h, 000h, 000h, 000h, 000h, 0cch, 0cch, 0cch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h,
000h
 53684 000143F9 CCCC7600000000      <1>
 53685 00014400 0000000000C3C3C3C3- <1>     db  000h, 000h, 000h, 000h, 000h, 0c3h, 0c3h, 0c3h, 0c3h, 066h, 03ch, 018h, 000h, 000h, 000h,
000h
 53686 00014409 663C1800000000      <1>
 53687 00014410 0000000000C3C3C3DB- <1>     db  000h, 000h, 000h, 000h, 000h, 0c3h, 0c3h, 0c3h, 0dbh, 0dbh, 0ffh, 066h, 000h, 000h, 000h,
000h
 53688 00014419 DBFF6600000000      <1>
 53689 00014420 0000000000C3663C18- <1>     db  000h, 000h, 000h, 000h, 000h, 0c3h, 066h, 03ch, 018h, 03ch, 066h, 0c3h, 000h, 000h, 000h,
000h
 53690 00014429 3C66C300000000      <1>
 53691 00014430 0000000000C6C6C6C6- <1>     db  000h, 000h, 000h, 000h, 000h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 07eh, 006h, 00ch, 0f8h,
000h
 53692 00014439 C6C67E060CF800      <1>
 53693 00014440 0000000000FECC1830- <1>     db  000h, 000h, 000h, 000h, 000h, 0feh, 0cch, 018h, 030h, 060h, 0c6h, 0feh, 000h, 000h, 000h,
000h
 53694 00014449 60C6FE00000000      <1>
 53695 00014450 00000E181818701818- <1>     db  000h, 000h, 00eh, 018h, 018h, 018h, 070h, 018h, 018h, 018h, 018h, 00eh, 000h, 000h, 000h,
000h
 53696 00014459 18180E00000000      <1>
```

```
 53697 00014460 000018181818001818- <1>     db  000h, 000h, 018h, 018h, 018h, 018h, 000h, 018h, 018h, 018h, 018h, 018h, 000h, 000h, 000h,
000h
 53698 00014469 18181800000000      <1>
 53699 00014470 0000701818180E1818- <1>     db  000h, 000h, 070h, 018h, 018h, 018h, 00eh, 018h, 018h, 018h, 018h, 070h, 000h, 000h, 000h,
000h
 53700 00014479 18187000000000      <1>
 53701 00014480 000076DC0000000000- <1>     db  000h, 000h, 076h, 0dch, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 53702 00014489 00000000000000      <1>
 53703 00014490 0000000010386CC6C6- <1>     db  000h, 000h, 000h, 000h, 010h, 038h, 06ch, 0c6h, 0c6h, 0c6h, 0feh, 000h, 000h, 000h, 000h,
000h
 53704 00014499 C6FE0000000000      <1>
 53705 000144A0 00003C66C2C0C0C0C2- <1>     db  000h, 000h, 03ch, 066h, 0c2h, 0c0h, 0c0h, 0c0h, 0c2h, 066h, 03ch, 00ch, 006h, 07ch, 000h,
000h
 53706 000144A9 663C0C067C0000      <1>
 53707 000144B0 0000CC0000CCCCCCCC- <1>     db  000h, 000h, 0cch, 000h, 000h, 0cch, 0cch, 0cch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h,
000h
 53708 000144B9 CCCC7600000000      <1>
 53709 000144C0 000C1830007CC6FEC0- <1>     db  000h, 00ch, 018h, 030h, 000h, 07ch, 0c6h, 0feh, 0c0h, 0c0h, 0c6h, 07ch, 000h, 000h, 000h,
000h
 53710 000144C9 C0C67C00000000      <1>
 53711 000144D0 0010386C00780C7CCC- <1>     db  000h, 010h, 038h, 06ch, 000h, 078h, 00ch, 07ch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h,
000h
 53712 000144D9 CCCC7600000000      <1>
 53713 000144E0 0000CC0000780C7CCC- <1>     db  000h, 000h, 0cch, 000h, 000h, 078h, 00ch, 07ch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h,
000h
 53714 000144E9 CCCC7600000000      <1>
 53715 000144F0 0060301800780C7CCC- <1>     db  000h, 060h, 030h, 018h, 000h, 078h, 00ch, 07ch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h,
000h
 53716 000144F9 CCCC7600000000      <1>
 53717 00014500 00386C3800780C7CCC- <1>     db  000h, 038h, 06ch, 038h, 000h, 078h, 00ch, 07ch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h,
000h
 53718 00014509 CCCC7600000000      <1>
 53719 00014510 000000003C66606066- <1>     db  000h, 000h, 000h, 000h, 03ch, 066h, 060h, 060h, 066h, 03ch, 00ch, 006h, 03ch, 000h, 000h,
000h
 53720 00014519 3C0C063C000000      <1>
 53721 00014520 0010386C007CC6FEC0- <1>     db  000h, 010h, 038h, 06ch, 000h, 07ch, 0c6h, 0feh, 0c0h, 0c0h, 0c6h, 07ch, 000h, 000h, 000h,
000h
 53722 00014529 C0C67C00000000      <1>
 53723 00014530 0000C600007CC6FEC0- <1>     db  000h, 000h, 0c6h, 000h, 000h, 07ch, 0c6h, 0feh, 0c0h, 0c0h, 0c6h, 07ch, 000h, 000h, 000h,
000h
 53724 00014539 C0C67C00000000      <1>
 53725 00014540 00603018007CC6FEC0- <1>     db  000h, 060h, 030h, 018h, 000h, 07ch, 0c6h, 0feh, 0c0h, 0c0h, 0c6h, 07ch, 000h, 000h, 000h,
000h
 53726 00014549 C0C67C00000000      <1>
 53727 00014550 000066000038181818- <1>     db  000h, 000h, 066h, 000h, 000h, 038h, 018h, 018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h,
000h
 53728 00014559 18183C00000000      <1>
 53729 00014560 00183C660038181818- <1>     db  000h, 018h, 03ch, 066h, 000h, 038h, 018h, 018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h,
000h
 53730 00014569 18183C00000000      <1>
 53731 00014570 006030180038181818- <1>     db  000h, 060h, 030h, 018h, 000h, 038h, 018h, 018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h,
000h
 53732 00014579 18183C00000000      <1>
 53733 00014580 00C60010386CC6C6FE- <1>     db  000h, 0c6h, 000h, 010h, 038h, 06ch, 0c6h, 0c6h, 0feh, 0c6h, 0c6h, 0c6h, 000h, 000h, 000h,
000h
 53734 00014589 C6C6C600000000      <1>
 53735 00014590 386C3800386CC6C6FE- <1>     db  038h, 06ch, 038h, 000h, 038h, 06ch, 0c6h, 0c6h, 0feh, 0c6h, 0c6h, 0c6h, 000h, 000h, 000h,
000h
 53736 00014599 C6C6C600000000      <1>
 53737 000145A0 18306000FE66607C60- <1>     db  018h, 030h, 060h, 000h, 0feh, 066h, 060h, 07ch, 060h, 060h, 066h, 0feh, 000h, 000h, 000h,
000h
 53738 000145A9 6066FE00000000      <1>
 53739 000145B0 00000000006E3B1B7E- <1>     db  000h, 000h, 000h, 000h, 000h, 06eh, 03bh, 01bh, 07eh, 0d8h, 0dch, 077h, 000h, 000h, 000h,
000h
 53740 000145B9 D8DC7700000000      <1>
 53741 000145C0 00003E6CCCCCFECCCC- <1>     db  000h, 000h, 03eh, 06ch, 0cch, 0cch, 0feh, 0cch, 0cch, 0cch, 0cch, 0ceh, 000h, 000h, 000h,
000h
 53742 000145C9 CCCCCE00000000      <1>
 53743 000145D0 0010386C007CC6C6C6- <1>     db  000h, 010h, 038h, 06ch, 000h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h,
000h
 53744 000145D9 C6C67C00000000      <1>
 53745 000145E0 0000C600007CC6C6C6- <1>     db  000h, 000h, 0c6h, 000h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h,
000h
 53746 000145E9 C6C67C00000000      <1>
 53747 000145F0 00603018007CC6C6C6- <1>     db  000h, 060h, 030h, 018h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h,
000h
 53748 000145F9 C6C67C00000000      <1>
 53749 00014600 003078CC00CCCCCCCC- <1>     db  000h, 030h, 078h, 0cch, 000h, 0cch, 0cch, 0cch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h,
000h
 53750 00014609 CCCC7600000000      <1>
 53751 00014610 0060301800CCCCCCCC- <1>     db  000h, 060h, 030h, 018h, 000h, 0cch, 0cch, 0cch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h,
000h
 53752 00014619 CCCC7600000000      <1>
 53753 00014620 0000C60000C6C6C6C6- <1>     db  000h, 000h, 0c6h, 000h, 000h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 07eh, 006h, 00ch, 078h,
000h
 53754 00014629 C6C67E060C7800      <1>
 53755 00014630 00C6007CC6C6C6C6C6- <1>     db  000h, 0c6h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h,
000h
 53756 00014639 C6C67C00000000      <1>
 53757 00014640 00C600C6C6C6C6C6C6- <1>     db  000h, 0c6h, 000h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h,
000h
 53758 00014649 C6C67C00000000      <1>
 53759 00014650 0018187EC3C0C0C0C3- <1>     db  000h, 018h, 018h, 07eh, 0c3h, 0c0h, 0c0h, 0c0h, 0c3h, 07eh, 018h, 018h, 000h, 000h, 000h,
000h
 53760 00014659 7E1818000000000     <1>
 53761 00014660 00386C6460F0606060- <1>     db  000h, 038h, 06ch, 064h, 060h, 0f0h, 060h, 060h, 060h, 060h, 0e6h, 0fch, 000h, 000h, 000h,
000h
 53762 00014669 60E6FC00000000      <1>
 53763 00014670 0000C3663C18FF18FF- <1>     db  000h, 000h, 0c3h, 066h, 03ch, 018h, 0ffh, 018h, 0ffh, 018h, 018h, 018h, 000h, 000h, 000h,
000h
 53764 00014679 18181800000000      <1>
```

```
 53765 00014680 00FC66667C62666F66- <1>     db  000h, 0fch, 066h, 066h, 07ch, 062h, 066h, 06fh, 066h, 066h, 066h, 0f3h, 000h, 000h, 000h,
000h
 53766 00014689 6666F300000000      <1>
 53767 00014690 000E1B1818187E1818- <1>     db  000h, 00eh, 01bh, 018h, 018h, 018h, 07eh, 018h, 018h, 018h, 018h, 018h, 0d8h, 070h, 000h,
000h
 53768 00014699 181818D8700000      <1>
 53769 000146A0 0018306000780C7CCC- <1>     db  000h, 018h, 030h, 060h, 000h, 078h, 00ch, 07ch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h,
000h
 53770 000146A9 CCCC7600000000      <1>
 53771 000146B0 000C18300038181818- <1>     db  000h, 00ch, 018h, 030h, 000h, 038h, 018h, 018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h,
000h
 53772 000146B9 18183C00000000      <1>
 53773 000146C0 00183060007CC6C6C6- <1>     db  000h, 018h, 030h, 060h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h,
000h
 53774 000146C9 C6C67C00000000      <1>
 53775 000146D0 0018306000CCCCCCCC- <1>     db  000h, 018h, 030h, 060h, 000h, 0cch, 0cch, 0cch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h,
000h
 53776 000146D9 CCCC7600000000      <1>
 53777 000146E0 000076DC00DC666666- <1>     db  000h, 000h, 076h, 0dch, 000h, 0dch, 066h, 066h, 066h, 066h, 066h, 066h, 000h, 000h, 000h,
000h
 53778 000146E9 66666600000000      <1>
 53779 000146F0 76DC00C6E6F6FEDECE- <1>     db  076h, 0dch, 000h, 0c6h, 0e6h, 0f6h, 0feh, 0deh, 0ceh, 0c6h, 0c6h, 0c6h, 000h, 000h, 000h,
000h
 53780 000146F9 C6C6C600000000      <1>
 53781 00014700 003C6C6C3E007E0000- <1>     db  000h, 03ch, 06ch, 06ch, 03eh, 000h, 07eh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 53782 00014709 00000000000000      <1>
 53783 00014710 00386C6C38007C0000- <1>     db  000h, 038h, 06ch, 06ch, 038h, 000h, 07ch, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 53784 00014719 00000000000000      <1>
 53785 00014720 0000303000303060C0- <1>     db  000h, 000h, 030h, 030h, 000h, 030h, 030h, 060h, 0c0h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h,
000h
 53786 00014729 C6C67C00000000      <1>
 53787 00014730 000000000000FEC0C0- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 0feh, 0c0h, 0c0h, 0c0h, 0c0h, 000h, 000h, 000h, 000h,
000h
 53788 00014739 C0C00000000000      <1>
 53789 00014740 000000000000FE0606- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 0feh, 006h, 006h, 006h, 006h, 000h, 000h, 000h, 000h,
000h
 53790 00014749 06060000000000      <1>
 53791 00014750 00C0C0C2C6CC183060- <1>     db  000h, 0c0h, 0c0h, 0c2h, 0c6h, 0cch, 018h, 030h, 060h, 0ceh, 09bh, 006h, 00ch, 01fh, 000h,
000h
 53792 00014759 CE9B060C1F0000      <1>
 53793 00014760 00C0C0C2C6CC183066- <1>     db  000h, 0c0h, 0c0h, 0c2h, 0c6h, 0cch, 018h, 030h, 066h, 0ceh, 096h, 03eh, 006h, 006h, 000h,
000h
 53794 00014769 CE963E06060000      <1>
 53795 00014770 00001818001818183C- <1>     db  000h, 000h, 018h, 018h, 000h, 018h, 018h, 018h, 03ch, 03ch, 03ch, 018h, 000h, 000h, 000h,
000h
 53796 00014779 3C3C1800000000      <1>
 53797 00014780 0000000000366CD86C- <1>     db  000h, 000h, 000h, 000h, 000h, 036h, 06ch, 0d8h, 06ch, 036h, 000h, 000h, 000h, 000h, 000h,
000h
 53798 00014789 36000000000000      <1>
 53799 00014790 0000000000D86C366C- <1>     db  000h, 000h, 000h, 000h, 000h, 0d8h, 06ch, 036h, 06ch, 0d8h, 000h, 000h, 000h, 000h, 000h,
000h
 53800 00014799 D8000000000000      <1>
 53801 000147A0 114411441144114411- <1>     db  011h, 044h, 011h, 044h, 011h, 044h, 011h, 044h, 011h, 044h, 011h, 044h, 011h, 044h, 011h,
044h
 53802 000147A9 44114411441144      <1>
 53803 000147B0 55AA55AA55AA55AA55- <1>     db  055h, 0aah, 055h, 0aah, 055h, 0aah, 055h, 0aah, 055h, 0aah, 055h, 0aah, 055h, 0aah, 055h,
0aah
 53804 000147B9 AA55AA55AA55AA      <1>
 53805 000147C0 DD77DD77DD77DD77DD- <1>     db  0ddh, 077h, 0ddh, 077h, 0ddh, 077h, 0ddh, 077h, 0ddh, 077h, 0ddh, 077h, 0ddh, 077h, 0ddh,
077h
 53806 000147C9 77DD77DD77DD77      <1>
 53807 000147D0 1818181818181818- <1>     db  018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h,
018h
 53808 000147D9 18181818181818      <1>
 53809 000147E0 18181818181818F818- <1>     db  018h, 018h, 018h, 018h, 018h, 018h, 018h, 0f8h, 018h, 018h, 018h, 018h, 018h, 018h, 018h,
018h
 53810 000147E9 18181818181818      <1>
 53811 000147F0 1818181818F818F818- <1>     db  018h, 018h, 018h, 018h, 018h, 0f8h, 018h, 0f8h, 018h, 018h, 018h, 018h, 018h, 018h, 018h,
018h
 53812 000147F9 18181818181818      <1>
 53813 00014800 363636363636F636- <1>     db  036h, 036h, 036h, 036h, 036h, 036h, 036h, 0f6h, 036h, 036h, 036h, 036h, 036h, 036h, 036h,
036h
 53814 00014809 36363636363636      <1>
 53815 00014810 00000000000000FE36- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 0feh, 036h, 036h, 036h, 036h, 036h, 036h, 036h,
036h
 53816 00014819 36363636363636      <1>
 53817 00014820 0000000000F818F818- <1>     db  000h, 000h, 000h, 000h, 000h, 0f8h, 018h, 0f8h, 018h, 018h, 018h, 018h, 018h, 018h, 018h,
018h
 53818 00014829 18181818181818      <1>
 53819 00014830 3636363636F606F636- <1>     db  036h, 036h, 036h, 036h, 036h, 0f6h, 006h, 0f6h, 036h, 036h, 036h, 036h, 036h, 036h, 036h,
036h
 53820 00014839 36363636363636      <1>
 53821 00014840 3636363636363636- <1>     db  036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h,
036h
 53822 00014849 36363636363636      <1>
 53823 00014850 0000000000FE06F636- <1>     db  000h, 000h, 000h, 000h, 000h, 0feh, 006h, 0f6h, 036h, 036h, 036h, 036h, 036h, 036h, 036h,
036h
 53824 00014859 36363636363636      <1>
 53825 00014860 3636363636F606FE00- <1>     db  036h, 036h, 036h, 036h, 036h, 0f6h, 006h, 0feh, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 53826 00014869 00000000000000      <1>
 53827 00014870 36363636363636FE00- <1>     db  036h, 036h, 036h, 036h, 036h, 036h, 036h, 0feh, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 53828 00014879 00000000000000      <1>
 53829 00014880 1818181818F818F800- <1>     db  018h, 018h, 018h, 018h, 018h, 0f8h, 018h, 0f8h, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 53830 00014889 00000000000000      <1>
 53831 00014890 000000000000000F818- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 0f8h, 018h, 018h, 018h, 018h, 018h, 018h, 018h,
018h
 53832 00014899 18181818181818      <1>
```

```
 53833 000148A0 181818181818181F00- <1>     db  018h, 018h, 018h, 018h, 018h, 018h, 018h, 01fh, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 53834 000148A9 00000000000000      <1>
 53835 000148B0 18181818181818FF00- <1>     db  018h, 018h, 018h, 018h, 018h, 018h, 018h, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 53836 000148B9 00000000000000      <1>
 53837 000148C0 00000000000000FF18- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 0ffh, 018h, 018h, 018h, 018h, 018h, 018h, 018h,
018h
 53838 000148C9 18181818181818      <1>
 53839 000148D0 181818181818181F18- <1>     db  018h, 018h, 018h, 018h, 018h, 018h, 018h, 01fh, 018h, 018h, 018h, 018h, 018h, 018h, 018h,
018h
 53840 000148D9 18181818181818      <1>
 53841 000148E0 00000000000000FF00- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 53842 000148E9 00000000000000      <1>
 53843 000148F0 18181818181818FF18- <1>     db  018h, 018h, 018h, 018h, 018h, 018h, 018h, 0ffh, 018h, 018h, 018h, 018h, 018h, 018h, 018h,
018h
 53844 000148F9 18181818181818      <1>
 53845 00014900 18181818181F181F18- <1>     db  018h, 018h, 018h, 018h, 018h, 01fh, 018h, 01fh, 018h, 018h, 018h, 018h, 018h, 018h, 018h,
018h
 53846 00014909 18181818181818      <1>
 53847 00014910 363636363636363736- <1>     db  036h, 036h, 036h, 036h, 036h, 036h, 036h, 037h, 036h, 036h, 036h, 036h, 036h, 036h, 036h,
036h
 53848 00014919 36363636363636      <1>
 53849 00014920 363636363637303F00- <1>     db  036h, 036h, 036h, 036h, 036h, 037h, 030h, 03fh, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 53850 00014929 00000000000000      <1>
 53851 00014930 00000000003F303736- <1>     db  000h, 000h, 000h, 000h, 000h, 03fh, 030h, 037h, 036h, 036h, 036h, 036h, 036h, 036h, 036h,
036h
 53852 00014939 36363636363636      <1>
 53853 00014940 3636363636F700FF00- <1>     db  036h, 036h, 036h, 036h, 036h, 0f7h, 000h, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 53854 00014949 00000000000000      <1>
 53855 00014950 0000000000FF00F736- <1>     db  000h, 000h, 000h, 000h, 000h, 0ffh, 000h, 0f7h, 036h, 036h, 036h, 036h, 036h, 036h, 036h,
036h
 53856 00014959 36363636363636      <1>
 53857 00014960 363636363637303736- <1>     db  036h, 036h, 036h, 036h, 036h, 037h, 030h, 037h, 036h, 036h, 036h, 036h, 036h, 036h, 036h,
036h
 53858 00014969 36363636363636      <1>
 53859 00014970 0000000000FF00FF00- <1>     db  000h, 000h, 000h, 000h, 000h, 0ffh, 000h, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 53860 00014979 00000000000000      <1>
 53861 00014980 3636363636F700F736- <1>     db  036h, 036h, 036h, 036h, 036h, 0f7h, 000h, 0f7h, 036h, 036h, 036h, 036h, 036h, 036h, 036h,
036h
 53862 00014989 36363636363636      <1>
 53863 00014990 1818181818FF00FF00- <1>     db  018h, 018h, 018h, 018h, 018h, 0ffh, 000h, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 53864 00014999 00000000000000      <1>
 53865 000149A0 36363636363636FF00- <1>     db  036h, 036h, 036h, 036h, 036h, 036h, 036h, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 53866 000149A9 00000000000000      <1>
 53867 000149B0 0000000000FF00FF18- <1>     db  000h, 000h, 000h, 000h, 000h, 0ffh, 000h, 0ffh, 018h, 018h, 018h, 018h, 018h, 018h, 018h,
018h
 53868 000149B9 18181818181818      <1>
 53869 000149C0 00000000000000FF36- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 0ffh, 036h, 036h, 036h, 036h, 036h, 036h, 036h,
036h
 53870 000149C9 36363636363636      <1>
 53871 000149D0 363636363636363F00- <1>     db  036h, 036h, 036h, 036h, 036h, 036h, 036h, 03fh, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 53872 000149D9 00000000000000      <1>
 53873 000149E0 18181818181F181F00- <1>     db  018h, 018h, 018h, 018h, 018h, 01fh, 018h, 01fh, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 53874 000149E9 00000000000000      <1>
 53875 000149F0 00000000001F181F18- <1>     db  000h, 000h, 000h, 000h, 000h, 01fh, 018h, 01fh, 018h, 018h, 018h, 018h, 018h, 018h, 018h,
018h
 53876 000149F9 18181818181818      <1>
 53877 00014A00 000000000000003F36- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 03fh, 036h, 036h, 036h, 036h, 036h, 036h, 036h,
036h
 53878 00014A09 36363636363636      <1>
 53879 00014A10 36363636363636FF36- <1>     db  036h, 036h, 036h, 036h, 036h, 036h, 036h, 0ffh, 036h, 036h, 036h, 036h, 036h, 036h, 036h,
036h
 53880 00014A19 36363636363636      <1>
 53881 00014A20 1818181818FF18FF18- <1>     db  018h, 018h, 018h, 018h, 018h, 0ffh, 018h, 0ffh, 018h, 018h, 018h, 018h, 018h, 018h, 018h,
018h
 53882 00014A29 18181818181818      <1>
 53883 00014A30 18181818181818F800- <1>     db  018h, 018h, 018h, 018h, 018h, 018h, 018h, 0f8h, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 53884 00014A39 00000000000000      <1>
 53885 00014A40 000000000000001F18- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 01fh, 018h, 018h, 018h, 018h, 018h, 018h, 018h,
018h
 53886 00014A49 18181818181818      <1>
 53887 00014A50 FFFFFFFFFFFFFFFFFF- <1>     db  0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh,
0ffh
 53888 00014A59 FFFFFFFFFFFFFF      <1>
 53889 00014A60 00000000000000FFFF- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh,
0ffh
 53890 00014A69 FFFFFFFFFFFFFF      <1>
 53891 00014A70 F0F0F0F0F0F0F0F0F0- <1>     db  0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h,
0f0h
 53892 00014A79 F0F0F0F0F0F0F0      <1>
 53893 00014A80 0F0F0F0F0F0F0F0F0F- <1>     db  00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh,
00fh
 53894 00014A89 0F0F0F0F0F0F0F      <1>
 53895 00014A90 FFFFFFFFFFFFFF0000- <1>     db  0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 53896 00014A99 00000000000000      <1>
 53897 00014AA0 000000000076DCD8D8- <1>     db  000h, 000h, 000h, 000h, 000h, 076h, 0dch, 0d8h, 0d8h, 0d8h, 0dch, 076h, 000h, 000h, 000h,
000h
 53898 00014AA9 D8DC7600000000      <1>
 53899 00014AB0 000078CCCCCCD8CCC6- <1>     db  000h, 000h, 078h, 0cch, 0cch, 0cch, 0d8h, 0cch, 0c6h, 0c6h, 0c6h, 0cch, 000h, 000h, 000h,
000h
 53900 00014AB9 C6C6CC00000000      <1>
```

```
 53901 00014AC0 0000FEC6C6C0C0C0C0- <1>     db  000h, 000h, 0feh, 0c6h, 0c6h, 0c0h, 0c0h, 0c0h, 0c0h, 0c0h, 0c0h, 0c0h, 000h, 000h, 000h,
000h
 53902 00014AC9 C0C0C000000000      <1>
 53903 00014AD0 00000000FE6C6C6C6C- <1>     db  000h, 000h, 000h, 000h, 0feh, 06ch, 06ch, 06ch, 06ch, 06ch, 06ch, 06ch, 000h, 000h, 000h,
000h
 53904 00014AD9 6C6C6C00000000      <1>
 53905 00014AE0 000000FEC660301830- <1>     db  000h, 000h, 000h, 0feh, 0c6h, 060h, 030h, 018h, 030h, 060h, 0c6h, 0feh, 000h, 000h, 000h,
000h
 53906 00014AE9 60C6FE00000000      <1>
 53907 00014AF0 00000000007ED8D8D8- <1>     db  000h, 000h, 000h, 000h, 000h, 07eh, 0d8h, 0d8h, 0d8h, 0d8h, 0d8h, 070h, 000h, 000h, 000h,
000h
 53908 00014AF9 D8D87000000000      <1>
 53909 00014B00 000000006666666666- <1>     db  000h, 000h, 000h, 000h, 066h, 066h, 066h, 066h, 066h, 07ch, 060h, 060h, 0c0h, 000h, 000h,
000h
 53910 00014B09 7C6060C0000000      <1>
 53911 00014B10 0000000076DC181818- <1>     db  000h, 000h, 000h, 000h, 076h, 0dch, 018h, 018h, 018h, 018h, 018h, 018h, 000h, 000h, 000h,
000h
 53912 00014B19 18181800000000      <1>
 53913 00014B20 0000007E183C666666- <1>     db  000h, 000h, 000h, 07eh, 018h, 03ch, 066h, 066h, 066h, 03ch, 018h, 07eh, 000h, 000h, 000h,
000h
 53914 00014B29 3C187E00000000      <1>
 53915 00014B30 000000386CC6C6FEC6- <1>     db  000h, 000h, 000h, 038h, 06ch, 0c6h, 0c6h, 0feh, 0c6h, 0c6h, 06ch, 038h, 000h, 000h, 000h,
000h
 53916 00014B39 C66C3800000000      <1>
 53917 00014B40 0000386CC6C6C66C6C- <1>     db  000h, 000h, 038h, 06ch, 0c6h, 0c6h, 0c6h, 06ch, 06ch, 06ch, 06ch, 0eeh, 000h, 000h, 000h,
000h
 53918 00014B49 6C6CEE00000000      <1>
 53919 00014B50 00001E30180C3E6666- <1>     db  000h, 000h, 01eh, 030h, 018h, 00ch, 03eh, 066h, 066h, 066h, 066h, 03ch, 000h, 000h, 000h,
000h
 53920 00014B59 66663C00000000      <1>
 53921 00014B60 000000000007EDBDBDB- <1>    db  000h, 000h, 000h, 000h, 000h, 07eh, 0dbh, 0dbh, 0dbh, 07eh, 000h, 000h, 000h, 000h, 000h,
000h
 53922 00014B69 7E000000000000      <1>
 53923 00014B70 00000003067EDBDBF3- <1>     db  000h, 000h, 000h, 003h, 006h, 07eh, 0dbh, 0dbh, 0f3h, 07eh, 060h, 0c0h, 000h, 000h, 000h,
000h
 53924 00014B79 7E60C000000000      <1>
 53925 00014B80 00001C3060607C6060- <1>     db  000h, 000h, 01ch, 030h, 060h, 060h, 07ch, 060h, 060h, 060h, 030h, 01ch, 000h, 000h, 000h,
000h
 53926 00014B89 60301C00000000      <1>
 53927 00014B90 0000007CC6C6C6C6C6- <1>     db  000h, 000h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 000h, 000h, 000h,
000h
 53928 00014B99 C6C6C600000000      <1>
 53929 00014BA0 00000000FE0000FE00- <1>     db  000h, 000h, 000h, 000h, 0feh, 000h, 000h, 0feh, 000h, 000h, 0feh, 000h, 000h, 000h, 000h,
000h
 53930 00014BA9 00FE0000000000      <1>
 53931 00014BB0 0000000018187E1818- <1>     db  000h, 000h, 000h, 000h, 018h, 018h, 07eh, 018h, 018h, 000h, 000h, 0ffh, 000h, 000h, 000h,
000h
 53932 00014BB9 0000FF00000000      <1>
 53933 00014BC0 00000030180C060C18- <1>     db  000h, 000h, 000h, 030h, 018h, 00ch, 006h, 00ch, 018h, 030h, 000h, 07eh, 000h, 000h, 000h,
000h
 53934 00014BC9 30007E00000000      <1>
 53935 00014BD0 0000000C1830603018- <1>     db  000h, 000h, 000h, 00ch, 018h, 030h, 060h, 030h, 018h, 00ch, 000h, 07eh, 000h, 000h, 000h,
000h
 53936 00014BD9 0C007E00000000      <1>
 53937 00014BE0 00000E1B1B18181818- <1>     db  000h, 000h, 00eh, 01bh, 01bh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h,
018h
 53938 00014BE9 18181818181818      <1>
 53939 00014BF0 1818181818181818D8- <1>     db  018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 0d8h, 0d8h, 0d8h, 070h, 000h, 000h, 000h,
000h
 53940 00014BF9 D8D87000000000      <1>
 53941 00014C00 000000001818007E00- <1>     db  000h, 000h, 000h, 000h, 018h, 018h, 000h, 07eh, 000h, 018h, 018h, 000h, 000h, 000h, 000h,
000h
 53942 00014C09 18180000000000      <1>
 53943 00014C10 000000000076DC0076- <1>     db  000h, 000h, 000h, 000h, 000h, 076h, 0dch, 000h, 076h, 0dch, 000h, 000h, 000h, 000h, 000h,
000h
 53944 00014C19 DC000000000000      <1>
 53945 00014C20 00386C6C3800000000- <1>     db  000h, 038h, 06ch, 06ch, 038h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 53946 00014C29 00000000000000      <1>
 53947 00014C30 000000000000001818- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 018h, 018h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 53948 00014C39 00000000000000      <1>
 53949 00014C40 0000000000000000018- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 018h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 53950 00014C49 00000000000000      <1>
 53951 00014C50 000F0C0C0C0C0CEC6C- <1>     db  000h, 00fh, 00ch, 00ch, 00ch, 00ch, 00ch, 0ech, 06ch, 06ch, 03ch, 01ch, 000h, 000h, 000h,
000h
 53952 00014C59 6C3C1C00000000      <1>
 53953 00014C60 00D86C6C6C6C6C0000- <1>     db  000h, 0d8h, 06ch, 06ch, 06ch, 06ch, 06ch, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 53954 00014C69 00000000000000      <1>
 53955 00014C70 0070D83060C8F80000- <1>     db  000h, 070h, 0d8h, 030h, 060h, 0c8h, 0f8h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 53956 00014C79 00000000000000      <1>
 53957 00014C80 000000007C7C7C7C7C- <1>     db  000h, 000h, 000h, 000h, 07ch, 07ch, 07ch, 07ch, 07ch, 07ch, 07ch, 000h, 000h, 000h, 000h,
000h
 53958 00014C89 7C7C0000000000      <1>
 53959 00014C90 000000000000000000- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h,
000h
 53960 00014C99 00000000000000      <1>
 53961                              <1> vgafont14alt:
 53962 00014CA0 1D000000002466FF66- <1>     db  01dh, 000h, 000h, 000h, 000h, 024h, 066h, 0ffh, 066h, 024h, 000h, 000h, 000h, 000h, 000h,
022h
 53963 00014CA9 24000000000022      <1>
 53964 00014CB0 006363632200000000- <1>     db  000h, 063h, 063h, 063h, 022h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 02bh,
000h
 53965 00014CB9 00000000002B00      <1>
 53966 00014CC0 0000181818FF181818- <1>     db  000h, 000h, 018h, 018h, 018h, 0ffh, 018h, 018h, 018h, 000h, 000h, 000h, 000h, 02dh, 000h,
000h
 53967 00014CC9 000000002D0000      <1>
 53968 00014CD0 00000000FF00000000- <1>     db  000h, 000h, 000h, 000h, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 04dh, 000h, 000h,
0c3h
 53969 00014CD9 0000004D0000C3      <1>
```

```
 53970 00014CE0 E7FFDBC3C3C3C3C300- <1>     db  0e7h, 0ffh, 0dbh, 0c3h, 0c3h, 0c3h, 0c3h, 0c3h, 000h, 000h, 000h, 054h, 000h, 000h, 0ffh,
0dbh
 53971 00014CE9 0000540000FFDB      <1>
 53972 00014CF0 9918181818183C0000- <1>     db  099h, 018h, 018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h, 056h, 000h, 000h, 0c3h, 0c3h,
0c3h
 53973 00014CF9 00560000C3C3C3      <1>
 53974 00014D00 C3C3C3663C18000000- <1>     db  0c3h, 0c3h, 0c3h, 066h, 03ch, 018h, 000h, 000h, 000h, 057h, 000h, 000h, 0c3h, 0c3h, 0c3h,
0c3h
 53975 00014D09 570000C3C3C3C3      <1>
 53976 00014D10 DBDBFF666600000058- <1>     db  0dbh, 0dbh, 0ffh, 066h, 066h, 000h, 000h, 000h, 058h, 000h, 000h, 0c3h, 0c3h, 066h, 03ch,
018h
 53977 00014D19 0000C3C3663C18      <1>
 53978 00014D20 3C66C3C30000005900- <1>     db  03ch, 066h, 0c3h, 0c3h, 000h, 000h, 000h, 059h, 000h, 000h, 0c3h, 0c3h, 0c3h, 066h, 03ch,
018h
 53979 00014D29 00C3C3C3663C18      <1>
 53980 00014D30 18183C0000005A0000- <1>     db  018h, 018h, 03ch, 000h, 000h, 000h, 05ah, 000h, 000h, 0ffh, 0c3h, 086h, 00ch, 018h, 030h,
061h
 53981 00014D39 FFC3860C183061      <1>
 53982 00014D40 C3FF0000006D000000- <1>     db  0c3h, 0ffh, 000h, 000h, 000h, 06dh, 000h, 000h, 000h, 000h, 000h, 0e6h, 0ffh, 0dbh, 0dbh,
0dbh
 53983 00014D49 0000E6FFDBDBDB      <1>
 53984 00014D50 DB0000007600000000- <1>     db  0dbh, 000h, 000h, 000h, 076h, 000h, 000h, 000h, 000h, 000h, 0c3h, 0c3h, 0c3h, 066h, 03ch,
018h
 53985 00014D59 00C3C3C3663C18      <1>
 53986 00014D60 000000770000000000- <1>     db  000h, 000h, 000h, 077h, 000h, 000h, 000h, 000h, 000h, 0c3h, 0c3h, 0dbh, 0dbh, 0ffh, 066h,
000h
 53987 00014D69 C3C3DBDBFF6600      <1>
 53988 00014D70 000091000000006E3B- <1>     db  000h, 000h, 091h, 000h, 000h, 000h, 000h, 06eh, 03bh, 01bh, 07eh, 0d8h, 0dch, 077h, 000h,
000h
 53989 00014D79 1B7ED8DC770000      <1>
 53990 00014D80 009B0018187EC3C0C0- <1>     db  000h, 09bh, 000h, 018h, 018h, 07eh, 0c3h, 0c0h, 0c0h, 0c3h, 07eh, 018h, 018h, 000h, 000h,
000h
 53991 00014D89 C37E1818000000      <1>
 53992 00014D90 9D0000C3663C18FF18- <1>     db  09dh, 000h, 000h, 0c3h, 066h, 03ch, 018h, 0ffh, 018h, 0ffh, 018h, 018h, 000h, 000h, 000h,
09eh
 53993 00014D99 FF18180000009E      <1>
 53994 00014DA0 00FC66667C62666F66- <1>     db  000h, 0fch, 066h, 066h, 07ch, 062h, 066h, 06fh, 066h, 066h, 0f3h, 000h, 000h, 000h, 0f1h,
000h
 53995 00014DA9 66F3000000F100      <1>
 53996 00014DB0 00181818FF18181800- <1>     db  000h, 018h, 018h, 018h, 0ffh, 018h, 018h, 018h, 000h, 0ffh, 000h, 000h, 000h, 0f6h, 000h,
000h
 53997 00014DB9 FF000000F60000      <1>
 53998 00014DC0 18180000FF00001818- <1>     db  018h, 018h, 000h, 000h, 0ffh, 000h, 000h, 018h, 018h, 000h, 000h, 000h, 000h
 53999 00014DC9 00000000            <1>
 54000                              <1> vgafont16alt:
 54001 00014DCD 1D00000000002466FF- <1>     db  01dh, 000h, 000h, 000h, 000h, 000h, 024h, 066h, 0ffh, 066h, 024h, 000h, 000h, 000h, 000h,
000h
 54002 00014DD6 66240000000000      <1>
 54003 00014DDD 003000003C66C3C3DB- <1>     db  000h, 030h, 000h, 000h, 03ch, 066h, 0c3h, 0c3h, 0dbh, 0dbh, 0c3h, 0c3h, 066h, 03ch, 000h,
000h
 54004 00014DE6 DBC3C3663C0000      <1>
 54005 00014DED 00004D0000C3E7FFFF- <1>     db  000h, 000h, 04dh, 000h, 000h, 0c3h, 0e7h, 0ffh, 0ffh, 0dbh, 0c3h, 0c3h, 0c3h, 0c3h, 0c3h,
000h
 54006 00014DF6 DBC3C3C3C3C300      <1>
 54007 00014DFD 000000540000FFDB99- <1>     db  000h, 000h, 000h, 054h, 000h, 000h, 0ffh, 0dbh, 099h, 018h, 018h, 018h, 018h, 018h, 018h,
03ch
 54008 00014E06 1818181818183C      <1>
 54009 00014E0D 00000000560000C3C3- <1>     db  000h, 000h, 000h, 000h, 056h, 000h, 000h, 0c3h, 0c3h, 0c3h, 0c3h, 0c3h, 0c3h, 0c3h, 066h,
03ch
 54010 00014E16 C3C3C3C3C3663C      <1>
 54011 00014E1D 18000000000570000C3- <1>     db  018h, 000h, 000h, 000h, 000h, 057h, 000h, 000h, 0c3h, 0c3h, 0c3h, 0c3h, 0c3h, 0dbh, 0dbh,
0ffh
 54012 00014E26 C3C3C3C3DBDBFF      <1>
 54013 00014E2D 666600000000580000- <1>     db  066h, 066h, 000h, 000h, 000h, 000h, 058h, 000h, 000h, 0c3h, 0c3h, 066h, 03ch, 018h, 018h,
03ch
 54014 00014E36 C3C3663C18183C      <1>
 54015 00014E3D 66C3C3000000005900- <1>     db  066h, 0c3h, 0c3h, 000h, 000h, 000h, 000h, 059h, 000h, 000h, 0c3h, 0c3h, 0c3h, 066h, 03ch,
018h
 54016 00014E46 00C3C3C3663C18      <1>
 54017 00014E4D 1818183C000000005A- <1>     db  018h, 018h, 018h, 03ch, 000h, 000h, 000h, 000h, 05ah, 000h, 000h, 0ffh, 0c3h, 086h, 00ch,
018h
 54018 00014E56 0000FFC3860C18      <1>
 54019 00014E5D 3060C1C3FF00000000- <1>     db  030h, 060h, 0c1h, 0c3h, 0ffh, 000h, 000h, 000h, 000h, 06dh, 000h, 000h, 000h, 000h, 000h,
0e6h
 54020 00014E66 6D0000000000E6      <1>
 54021 00014E6D FFDBDBDBDBDB000000- <1>     db  0ffh, 0dbh, 0dbh, 0dbh, 0dbh, 0dbh, 000h, 000h, 000h, 000h, 076h, 000h, 000h, 000h, 000h,
000h
 54022 00014E76 00760000000000      <1>
 54023 00014E7D C3C3C3C3663C180000- <1>     db  0c3h, 0c3h, 0c3h, 0c3h, 066h, 03ch, 018h, 000h, 000h, 000h, 000h, 077h, 000h, 000h, 000h,
000h
 54024 00014E86 00007700000000      <1>
 54025 00014E8D 00C3C3C3DBDBFF6600- <1>     db  000h, 0c3h, 0c3h, 0c3h, 0dbh, 0dbh, 0ffh, 066h, 000h, 000h, 000h, 000h, 078h, 000h, 000h,
000h
 54026 00014E96 00000078000000      <1>
 54027 00014E9D 0000C3663C183C66C3- <1>     db  000h, 000h, 0c3h, 066h, 03ch, 018h, 03ch, 066h, 0c3h, 000h, 000h, 000h, 000h, 091h, 000h,
000h
 54028 00014EA6 00000000910000      <1>
 54029 00014EAD 0000006E3B1B7ED8DC- <1>     db  000h, 000h, 000h, 06eh, 03bh, 01bh, 07eh, 0d8h, 0dch, 077h, 000h, 000h, 000h, 000h, 09bh,
000h
 54030 00014EB6 77000000009B00      <1>
 54031 00014EBD 18187EC3C0C0C0C37E- <1>     db  018h, 018h, 07eh, 0c3h, 0c0h, 0c0h, 0c0h, 0c3h, 07eh, 018h, 018h, 000h, 000h, 000h, 000h,
09dh
 54032 00014EC6 1818000000009D      <1>
 54033 00014ECD 0000C3663C18FF18FF- <1>     db  000h, 000h, 0c3h, 066h, 03ch, 018h, 0ffh, 018h, 0ffh, 018h, 018h, 018h, 000h, 000h, 000h,
000h
 54034 00014ED6 18181800000000      <1>
 54035 00014EDD 9E00FC66667C62666F- <1>     db  09eh, 000h, 0fch, 066h, 066h, 07ch, 062h, 066h, 06fh, 066h, 066h, 066h, 0f3h, 000h, 000h,
000h
 54036 00014EE6 666666F3000000      <1>
 54037 00014EED 00AB00C0C0C2C6CC18- <1>     db  000h, 0abh, 000h, 0c0h, 0c0h, 0c2h, 0c6h, 0cch, 018h, 030h, 060h, 0ceh, 09bh, 006h, 00ch,
01fh
 54038 00014EF6 3060CE9B060C1F      <1>
```

```
 54039 00014EFD 0000AC00C0C0C2C6CC- <1>     db  000h, 000h, 0ach, 000h, 0c0h, 0c0h, 0c2h, 0c6h, 0cch, 018h, 030h, 066h, 0ceh, 096h, 03eh,
006h
 54040 00014F06 183066CE963E06      <1>
 54041 00014F0D 06000000            <1>     db  006h, 000h, 000h, 000h
 54042
 54043 00014F11 90                      align 2
 54044
 54045                                  ; EPOCH Variables
 54046                                  ; 13/04/2015 - Retro UNIX 386 v1 Beginning
 54047                                  ; 09/04/2013 epoch variables
 54048                                  ; Retro UNIX 8086 v1 Prototype: UNIXCOPY.ASM, 10/03/2013
 54049                                  ;
 54050 00014F12 B207                    year:       dw 1970
 54051 00014F14 0100                    month:      dw 1
 54052 00014F16 0100                    day:  dw 1
 54053 00014F18 0000                    hour:       dw 0
 54054 00014F1A 0000                    minute: dw 0
 54055 00014F1C 0000                    second: dw 0
 54056
 54057                                  DMonth:
 54058 00014F1E 0000                          dw 0
 54059 00014F20 1F00                          dw 31
 54060 00014F22 3B00                          dw 59
 54061 00014F24 5A00                          dw 90
 54062 00014F26 7800                          dw 120
 54063 00014F28 9700                          dw 151
 54064 00014F2A B500                          dw 181
 54065 00014F2C D400                          dw 212
 54066 00014F2E F300                          dw 243
 54067 00014F30 1101                          dw 273
 54068 00014F32 3001                          dw 304
 54069 00014F34 4E01                          dw 334
 54070
 54071                                  ; 20/02/2017
 54072                                  KERNELFSIZE  equ $  ; 04/07/2016
 54073
 54074                                  bss_start:
 54075
 54076                                  ABSOLUTE bss_start
 54077
 54078 00014F36 <res 00000002>          alignb 8 ; 25/12/2016
 54079
 54080                                        ; 15/04/2016
 54081                                        ; TRDOS 386 (TRDOS v2.0)
 54082                                        ;     80 interrupts
 54083                                        ; 11/03/2015
 54084                                        ; Interrupt Descriptor Table (20/08/2014)
 54085                                  idt:
 54086                                        ;resb  64*8 ; INT 0 to INT 3Fh
 54087                                        ; 15/04/2016
 54088 00014F38 <res 00000280>                resb   80*8 ; INT 0 to INT 4Fh
 54089
 54090                                  idt_end:
 54091
 54092                                  ;alignb 4
 54093
 54094                                  task_state_segment:
 54095                                        ; 24/03/2015
 54096 000151B8 <res 00000002>          tss.link:   resw 1
 54097 000151BA <res 00000002>                    resw 1
 54098                                  ; tss offset 4
 54099 000151BC <res 00000004>          tss.esp0:   resd 1
 54100 000151C0 <res 00000002>          tss.ss0:    resw 1
 54101 000151C2 <res 00000002>                    resw 1
 54102 000151C4 <res 00000004>          tss.esp1:   resd 1
 54103 000151C8 <res 00000002>          tss.ss1:    resw 1
 54104 000151CA <res 00000002>                    resw 1
 54105 000151CC <res 00000004>          tss.esp2:   resd 1
 54106 000151D0 <res 00000002>          tss.ss2:    resw 1
 54107 000151D2 <res 00000002>                    resw 1
 54108                                  ; tss offset 28
 54109 000151D4 <res 00000004>          tss.CR3:    resd 1
 54110 000151D8 <res 00000004>          tss.eip:    resd 1
 54111 000151DC <res 00000004>          tss.eflags: resd 1
 54112                                  ; tss offset 40
 54113 000151E0 <res 00000004>          tss.eax:    resd 1
 54114 000151E4 <res 00000004>          tss.ecx:    resd 1
 54115 000151E8 <res 00000004>          tss.edx:    resd 1
 54116 000151EC <res 00000004>          tss.ebx:    resd 1
 54117 000151F0 <res 00000004>          tss.esp:    resd 1
 54118 000151F4 <res 00000004>          tss.ebp:    resd 1
 54119 000151F8 <res 00000004>          tss.esi:    resd 1
 54120 000151FC <res 00000004>          tss.edi:    resd 1
 54121                                  ; tss offset 72
 54122 00015200 <res 00000002>          tss.ES:     resw 1
 54123 00015202 <res 00000002>                    resw 1
 54124 00015204 <res 00000002>          tss.CS:          resw 1
 54125 00015206 <res 00000002>                    resw 1
 54126 00015208 <res 00000002>          tss.SS:          resw 1
 54127 0001520A <res 00000002>                    resw 1
 54128 0001520C <res 00000002>          tss.DS:          resw 1
 54129 0001520E <res 00000002>                    resw 1
 54130 00015210 <res 00000002>          tss.FS:          resw 1
 54131 00015212 <res 00000002>                    resw 1
 54132 00015214 <res 00000002>          tss.GS:          resw 1
 54133 00015216 <res 00000002>                    resw 1
 54134 00015218 <res 00000002>          tss.LDTR:   resw 1
 54135 0001521A <res 00000002>                    resw 1
 54136                                  ; tss offset 100
 54137 0001521C <res 00000002>                    resw 1
 54138 0001521E <res 00000002>          tss.IOPB:   resw 1
 54139                                  ; tss offset 104
 54140                                  tss_end:
```

```
54141
54142 00015220 <res 00000004>          k_page_dir:  resd 1 ; Kernel's (System) Page Directory address
54143                                               ; (Physical address = Virtual address)
54144 00015224 <res 00000004>          memory_size: resd 1 ; memory size in pages
54145 00015228 <res 00000004>          free_pages:  resd 1 ; number of free pages
54146 0001522C <res 00000004>          next_page:   resd 1 ; offset value in M.A.T. for
54147                                               ; first free page search
54148 00015230 <res 00000004>          last_page:   resd 1 ; offset value in M.A.T. which
54149                                               ; next free page search will be
54150                                               ; stopped after it. (end of M.A.T.)
54151 00015234 <res 00000004>          first_page:  resd 1 ; offset value in M.A.T. which
54152                                               ; first free page search
54153                                               ; will be started on it. (for user)
54154 00015238 <res 00000004>          mat_size:    resd 1 ; Memory Allocation Table size in pages
54155
54156                                  ; 02/09/2014 (Retro UNIX 386 v1)
54157                                  ; 04/12/2013 (Retro UNIX 8086 v1)
54158 0001523C <res 00000002>          CRT_START:   resw 1        ; starting address in regen buffer
54159                                                   ; NOTE: active page only
54160 0001523E <res 00000010>          CURSOR_POSN: resw 8 ; cursor positions for video pages
54161                                  ACTIVE_PAGE:
54162 0001524E <res 00000001>          ptty:            resb 1 ; current tty
54163                                  ; 01/07/2015 - 29/01/2016
54164 0001524F <res 00000001>          ccolor:          resb 1 ; current color attribute
54165                                  ; 26/10/2015
54166                                  ; 07/09/2014
54167 00015250 <res 00000014>          ttychr:      resw ntty+2 ; Character buffer (multiscreen)
54168
54169                                  ; 18/05/2015 (03/06/2013 - Retro UNIX 8086 v1 feature only!)
54170 00015264 <res 00000004>          p_time:      resd 1     ; present time (for systime & sysmdate)
54171
54172                                  ; 18/05/2015 (16/08/2013 - Retro UNIX 8086 v1 feature only !)
54173                                  ; (open mode locks for pseudo TTYs)
54174                                  ; [ major tty locks (return error in any conflicts) ]
54175 00015268 <res 00000014>          ttyl:        resw ntty+2 ; opening locks for TTYs.
54176
54177                                  ; 15/04/2015 (Retro UNIX 386 v1)
54178                                  ; 22/09/2013 (Retro UNIX 8086 v1)
54179 0001527C <res 0000000A>          wlist:       resb ntty+2 ; wait channel list (0 to 9 for TTYs)
54180                                  ; 15/04/2015 (Retro UNIX 386 v1)
54181                                  ;; 12/07/2014 -> sp_init set comm. parameters as 0E3h
54182                                  ;; 0 means serial port is not available
54183                                  ;;comprm: ; 25/06/2014
54184 00015286 <res 00000001>          com1p:       resb 1  ;;0E3h
54185 00015287 <res 00000001>          com2p:       resb 1  ;;0E3h
54186
54187                                  ; 17/11/2015
54188                                  ; request for response (from the terminal)
54189 00015288 <res 00000002>          req_resp:    resw 1
54190                                  ; 07/11/2015
54191 0001528A <res 00000001>          ccomport:    resb 1 ; current COM (serial) port
54192                                                  ; (0= COM1, 1= COM2)
54193                                  ; 09/11/2015
54194 0001528B <res 00000001>          comqr:           resb 1 ; 'query or response' sign (u9.s, 'sndc')
54195                                  ; 07/11/2015
54196 0001528C <res 00000002>          rchar:           resw 1 ; last received char for COM 1 and COM 2
54197 0001528E <res 00000002>          schar:           resw 1 ; last sent char for COM 1 and COM 2
54198
54199                                  ; 22/08/2014 (RTC)
54200                                  ; (Packed BCD)
54201 00015290 <res 00000001>          time_seconds: resb 1
54202 00015291 <res 00000001>          time_minutes: resb 1
54203 00015292 <res 00000001>          time_hours:   resb 1
54204 00015293 <res 00000001>          date_wday:    resb 1
54205 00015294 <res 00000001>          date_day:     resb 1
54206 00015295 <res 00000001>          date_month:   resb 1
54207 00015296 <res 00000001>          date_year:    resb 1
54208 00015297 <res 00000001>          date_century: resb 1
54209
54210                                  ; 24/01/2016
54211 00015298 <res 00000004>          RTC_LH:           resd 1
54212 0001529C <res 00000001>          RTC_WAIT_FLAG: resb 1
54213 0001529D <res 00000001>          USER_FLAG:     resb 1
54214                                  ; 19/05/2016
54215                                  ;RTC_second:
54216 0001529E <res 00000001>          RTC_2Hz:       resb 1 ;  from 2Hz interrupt to 1Hz timer event function
54217
54218                                  %include 'diskbss.s'     ; UNINITIALIZED DISK (BIOS) DATA
54219                             <1> ; ****************************************************************************
54220                             <1> ; TRDOS386.ASM (TRDOS 386 Kernel) - v2.0.0 - diskbss.s
54221                             <1> ; ----------------------------------------------------------------------------
54222                             <1> ; Last Update: 24/01/2016
54223                             <1> ; ----------------------------------------------------------------------------
54224                             <1> ; Beginning: 24/01/2016
54225                             <1> ; ----------------------------------------------------------------------------
54226                             <1> ; Assembler: NASM version 2.11 (trdos386.s)
54227                             <1> ; ----------------------------------------------------------------------------
54228                             <1> ; Turkish Rational DOS
54229                             <1> ; Operating System Project v2.0 by ERDOGAN TAN (Beginning: 04/01/2016)
54230                             <1> ;
54231                             <1> ; Derived from 'Retro UNIX 386 Kernel - v0.2.1.0' source code by Erdogan Tan
54232                             <1> ; diskbss.inc (10/07/2015)
54233                             <1> ;
54234                             <1> ; Derived from 'IBM PC-XT-286' BIOS source code (1986)
54235                             <1> ; ****************************************************************************
54236                             <1>
54237                             <1> ; Retro UNIX 386 v1 Kernel - DISKBSS.INC
54238                             <1> ; Last Modification: 10/07/2015
54239                             <1> ;     (Unnitialized Disk Parameters Data section for 'DISKIO.INC')
54240                             <1>
54241 0001529F <res 00000001>      <1> alignb 2
54242                             <1>
54243                             <1> ;---------------------------------------
```

```
54244                                  <1> ;     TIMER DATA AREA           :
54245                                  <1> ;---------------------------------------
54246                                  <1>
54247                                  <1> TIMER_LH:    ; 16/02/205
54248 000152A0 <res 00000002>          <1> TIMER_LOW:      resw     1               ; LOW WORD OF TIMER COUNT
54249 000152A2 <res 00000002>          <1> TIMER_HIGH:     resw     1               ; HIGH WORD OF TIMER COUNT
54250 000152A4 <res 00000001>          <1> TIMER_OFL:      resb     1               ; TIMER HAS ROLLED OVER SINCE LAST READ
54251                                  <1>
54252                                  <1> ;---------------------------------------
54253                                  <1> ;     DISKETTE DATA AREAS       :
54254                                  <1> ;---------------------------------------
54255                                  <1>
54256 000152A5 <res 00000001>          <1> SEEK_STATUS: resb   1
54257 000152A6 <res 00000001>          <1> MOTOR_STATUS:      resb   1
54258 000152A7 <res 00000001>          <1> MOTOR_COUNT: resb   1
54259 000152A8 <res 00000001>          <1> DSKETTE_STATUS:    resb   1
54260 000152A9 <res 00000007>          <1> NEC_STATUS:  resb   7
54261                                  <1>
54262                                  <1> ;---------------------------------------
54263                                  <1> ;     ADDITIONAL MEDIA DATA           :
54264                                  <1> ;---------------------------------------
54265                                  <1>
54266 000152B0 <res 00000001>          <1> LASTRATE:    resb   1
54267 000152B1 <res 00000001>          <1> HF_STATUS:   resb   1
54268 000152B2 <res 00000001>          <1> HF_ERROR:    resb   1
54269 000152B3 <res 00000001>          <1> HF_INT_FLAG: resb   1
54270 000152B4 <res 00000001>          <1> HF_CNTRL:    resb   1
54271 000152B5 <res 00000004>          <1> DSK_STATE:   resb   4
54272 000152B9 <res 00000002>          <1> DSK_TRK:     resb   2
54273                                  <1>
54274                                  <1> ;---------------------------------------
54275                                  <1> ;     FIXED DISK DATA AREAS           :
54276                                  <1> ;---------------------------------------
54277                                  <1>
54278 000152BB <res 00000001>          <1> DISK_STATUS1:      resb   1            ; FIXED DISK STATUS
54279 000152BC <res 00000001>          <1> HF_NUM:            resb   1            ; COUNT OF FIXED DISK DRIVES
54280 000152BD <res 00000001>          <1> CONTROL_BYTE:      resb   1            ; HEAD CONTROL BYTE
54281                                  <1> ;@PORT_OFF   resb   1            ; RESERVED (PORT OFFSET)
54282                                  <1> ;port1_off   resb   1            ; Hard disk controller 1 - port offset
54283                                  <1> ;port2_off   resb   1            ; Hard idsk controller 2 - port offset
54284                                  <1>
54285 000152BE <res 00000002>          <1> alignb 4
54286                                  <1>
54287                                  <1> ;HF_TBL_VEC: resd   1            ; Primary master disk param. tbl. pointer
54288                                  <1> ;HF1_TBL_VEC:      resd   1            ; Primary slave disk param. tbl. pointer
54289                                  <1> HF_TBL_VEC: ; 22/12/2014
54290 000152C0 <res 00000004>          <1> HDPM_TBL_VEC:      resd   1            ; Primary master disk param. tbl. pointer
54291 000152C4 <res 00000004>          <1> HDPS_TBL_VEC:      resd   1            ; Primary slave disk param. tbl. pointer
54292 000152C8 <res 00000004>          <1> HDSM_TBL_VEC:      resd   1            ; Secondary master disk param. tbl. pointer
54293 000152CC <res 00000004>          <1> HDSS_TBL_VEC:      resd   1            ; Secondary slave disk param. tbl. pointer
54294                                  <1>
54295                                  <1> ; 03/01/2015
54296 000152D0 <res 00000001>          <1> LBAMode:           resb   1
54297                                  <1>
54298                                  <1> ; ****************************************************************************
54299
54300                                      ;;; Real Mode Data (10/07/2015 - BSS)
54301
54302                                      ;alignb 2
54303
54304                                      ; 10/01/2016
54305                                      %include 'trdoskx.s'      ; UNINITIALIZED KERNEL (Logical Drive & FS) DATA
54306                                  <1> ; ****************************************************************************
54307                                  <1> ; TRDOS386.ASM (TRDOS 386 Kernel - v2.0.0) - UNINITIALIZED DATA : trdoskx.s
54308                                  <1> ; ----------------------------------------------------------------------------
54309                                  <1> ; Last Update: 28/08/2017
54310                                  <1> ; ----------------------------------------------------------------------------
54311                                  <1> ; Beginning: 04/01/2016
54312                                  <1> ; ----------------------------------------------------------------------------
54313                                  <1> ; Assembler: NASM version 2.11 (trdos386.s)
54314                                  <1> ; ----------------------------------------------------------------------------
54315                                  <1> ; Derived from TRDOS Operating System v1.0 (8086) source code by Erdogan Tan
54316                                  <1> ; TRDOS2.ASM (09/11/2011)
54317                                  <1> ; ****************************************************************************
54318                                  <1> ; DRV_INIT.ASM [26/09/2009] Last Update: 07/08/2011
54319                                  <1> ; MAINPROG.ASM [17/01/2004] Last Update: 09/11/2011
54320                                  <1> ; DIR.ASM      [17/01/2004] Last Update: 09/10/2011
54321                                  <1> ; CMD_INTR.ASM [29/01/2005] Last update: 09/11/2011
54322                                  <1> ; DRV_FAT.ASM  [07/07/2009] Last update: 21/08/2011
54323                                  <1>
54324 000152D1 <res 00000003>          <1> alignb 4
54325                                  <1>
54326                                  <1> ; MAINPROG.ASM
54327 000152D4 <res 00000004>          <1> MainProgCfg_FileSize:   resd 1 ; 14/04/2016
54328 000152D8 <res 00000004>          <1> MainProgCfg_LineOffset: resd 1 ; 14/04/2016
54329                                  <1>
54330 000152DC <res 00000004>          <1> Current_VolSerial: resd 1
54331                                  <1>
54332 000152E0 <res 00000004>          <1> Current_Dir_FCluster: resd 1
54333                                  <1>
54334 000152E4 <res 00000001>          <1> Current_Dir_Level: resb 1
54335 000152E5 <res 00000001>          <1> Current_FATType: resb 1
54336 000152E6 <res 00000001>          <1> Current_Drv: resb 1
54337 000152E7 <res 00000001>          <1> Current_Dir_Drv:   resb 1 ; '?'
54338 000152E8 <res 00000001>          <1>                    resb 1 ; ':'
54339 000152E9 <res 00000001>          <1> Current_Dir_Root:  resb 1 ; '/'
54340 000152EA <res 0000005A>          <1> Current_Directory: resb 90
54341 00015344 <res 00000001>          <1> End_Of_Current_Dir_Str: resb 1
54342 00015345 <res 00000001>          <1> Current_Dir_StrLen: resb 1
54343                                  <1>
54344 00015346 <res 00000001>          <1> CursorColumn:      resb 1
54345 00015347 <res 00000001>          <1> CmdArgStart:    resb 1
54346                                  <1>
```

```
54347                                  <1> ; 03/02/2016
54348 00015348 <res 0000004E>          <1> Remark:            resb 78
54349                                  <1>
54350 00015396 <res 00000050>          <1> CommandBuffer:     resb 80
54351                                  <1>
54352 000153E6 <res 00000100>          <1> TextBuffer:  resb 256
54353                                  <1>
54354                                  <1> MasterBootBuff:
54355 000154E6 <res 000001BE>          <1> MasterBootCode: resb 1BEh
54356 000156A4 <res 00000040>          <1> PartitionTable: resb 64
54357 000156E4 <res 00000002>          <1> MBIDCode: resw 1
54358                                  <1>
54359                                  <1> PTable_Buffer:
54360 000156E6 <res 00000040>          <1> PTable_hd0: resb 64
54361 00015726 <res 00000040>          <1> PTable_hd1: resb 64
54362 00015766 <res 00000040>          <1> PTable_hd2: resb 64
54363 000157A6 <res 00000040>          <1> PTable_hd3: resb 64
54364 000157E6 <res 00000040>          <1> PTable_ep0: resb 64
54365 00015826 <res 00000040>          <1> PTable_ep1: resb 64
54366 00015866 <res 00000040>          <1> PTable_ep2: resb 64
54367 000158A6 <res 00000040>          <1> PTable_ep3: resb 64
54368                                  <1>
54369 000158E6 <res 00000001>          <1> scount:     resb 1 ; 16/05/2016 (diskio.s, 'int33h:')
54370 000158E7 <res 00000001>          <1> HD_LBA_yes: resb 1
54371 000158E8 <res 00000001>          <1> PP_Counter: resb 1
54372 000158E9 <res 00000001>          <1> EP_Counter: resb 1
54373                                  <1>
54374 000158EA <res 00000004>          <1> EP_StartSector: resd 1
54375 000158EE <res 00000004>          <1>                 resd 1
54376 000158F2 <res 00000004>          <1>                 resd 1
54377 000158F6 <res 00000004>          <1>                 resd 1
54378                                  <1>
54379 000158FA <res 00000200>          <1> DOSBootSectorBuff: resb 512
54380                                  <1>
54381                                  <1> FAT_BuffDescriptor:
54382 00015AFA <res 00000004>          <1> FAT_CurrentCluster: resd 1
54383 00015AFE <res 00000001>          <1> FAT_BuffValidData: resb 1
54384 00015AFF <res 00000001>          <1> FAT_BuffDrvName: resb 1
54385 00015B00 <res 00000002>          <1> FAT_BuffOffset: resw 1
54386 00015B02 <res 00000004>          <1> FAT_BuffSector: resd 1
54387                                  <1>
54388 00015B06 <res 00000004>          <1> FAT_ClusterCounter: resd 1
54389 00015B0A <res 00000004>          <1> LastCluster: resd 1
54390                                  <1>
54391                                  <1> ; 16/05/2016
54392                                  <1> ;; 18/03/2016 (TRDOS v2.0)
54393                                  <1> ;ClusterBuffer_Valid: resb 1
54394                                  <1>
54395                                  <1> Dir_BuffDescriptor:
54396 00015B0E <res 00000001>          <1> DirBuff_DRV: resb 1
54397 00015B0F <res 00000001>          <1> DirBuff_FATType: resb 1
54398 00015B10 <res 00000001>          <1> DirBuff_ValidData: resb 1
54399 00015B11 <res 00000002>          <1> DirBuff_CurrentEntry: resw 1
54400 00015B13 <res 00000002>          <1> DirBuff_LastEntry: resw 1
54401 00015B15 <res 00000004>          <1> DirBuff_Cluster: resd 1
54402 00015B19 <res 00000002>          <1> DirBuffer_Size: resw 1
54403                                  <1> ;DirBuff_EntryCounter: resw 1
54404                                  <1>
54405                                  <1> ; 01/02/2016
54406                                  <1> ; these are on (real mode) segment 8000h and later
54407                                  <1> ; FAT_Buffer:      resb 1536 ; 3 sectors
54408                                  <1> ; Dir_Buffer:      resb 512*32
54409                                  <1> ; Logical_DOSDisks:  resb 6656 ; 26 * 256 bytes
54410                                  <1>
54411                                  <1> ; 18/01/2016
54412                                  <1>
54413 00015B1B <res 00000004>          <1> FreeClusterCount: resd 1
54414                                  <1>
54415 00015B1F <res 00000004>          <1> VolSize_Unit1:   resd 1
54416 00015B23 <res 00000004>          <1> VolSize_Unit2:   resd 1
54417                                  <1>
54418 00015B27 <res 00000004>          <1> Vol_Tot_Sec_Str_Start:       resd 1
54419 00015B2B <res 0000000A>          <1> Vol_Tot_Sec_Str:       resb 10
54420 00015B35 <res 00000001>          <1> Vol_Tot_Sec_Str_End:         resb 1
54421 00015B36 <res 00000001>          <1> resb 1
54422 00015B37 <res 00000004>          <1> Vol_Free_Sectors_Str_Start: resd 1
54423 00015B3B <res 0000000A>          <1> Vol_Free_Sectors_Str:        resb 10
54424 00015B45 <res 00000001>          <1> Vol_Free_Sectors_Str_End:   resb 1
54425                                  <1>
54426                                  <1> ; 10/02/2016
54427 00015B46 <res 00000001>          <1> RUN_CDRV: resb 1 ; CMD_INTR.ASM  ; 09/11/2011
54428                                  <1>
54429                                  <1> ; 24/01/2016
54430 00015B47 <res 00000080>          <1> PATH_Array:     resb 128 ; DIR.ASM ; 09/10/2011
54431                                  <1> ; 06/02/2016
54432 00015BC7 <res 00000004>          <1> CCD_DriveDT: resd 1 ; DIR.ASM ; (word)
54433 00015BCB <res 00000001>          <1> CCD_Level:   resb 1 ; DIR.ASM
54434 00015BCC <res 00000001>          <1> Last_Dir_Level:    resb 1 ; DIR.ASM
54435                                  <1> ;
54436 00015BCD <res 00000002>          <1> CDLF_AttributesMask: resw 1 ; DIR.ASM
54437 00015BCF <res 00000004>          <1> CDLF_FNAddress:    resd 1 ; DIR.ASM (word)
54438 00015BD3 <res 00000002>          <1> CDLF_DEType: resw 1 ; DIR.ASM
54439                                  <1> ;
54440 00015BD5 <res 00000001>          <1> CD_COMMAND:  resb 1 ; DIR.ASM
54441                                  <1>
54442 00015BD6 <res 00000002>          <1> alignb 4
54443                                  <1>
54444                                  <1> ; 29/01/2016
54445 00015BD8 <res 00000001>          <1> Program_Exit:      resb 1 ; CMD_INTR.ASM  ; 09/11/2011
54446                                  <1>
54447                                  <1> ;alignb 4
54448                                  <1> ; 23/02/2016
54449 00015BD9 <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
00015BDA <res 00000001>         <1> retry_count:       resb 1 ; DISK_IO.ASM ; 20/07/2011 (CHS_RetryCount)
00015BDB <res 00000001>         <1> disk_rw_err:       resb 1 ; DISK_IO.ASM ; (Disk_IO_err_code)
00015BDC <res 00000004>         <1> sector_count:      resd 1 ; DISK_IO.ASM ; (Disk_RW_SectorCount)
                                <1>
                                <1> ; 06/02/2016 (long name)
00015BE0 <res 00000002>         <1> FDE_AttrMask:         resw 1 ; DIR.ASM
00015BE2 <res 00000002>         <1> AmbiguousFileName: resw 1 ; DIR.ASM
00015BE4 <res 00000001>         <1> PreviousAttr:         resb 1 ; DIR.ASM
                                <1> ;
00015BE5 <res 00000001>         <1> LongNameFound:   resb 1     ; DIR.ASM
00015BE6 <res 00000001>         <1> LFN_EntryLength: resb 1   ; DIR.ASM
00015BE7 <res 00000001>         <1> LFN_CheckSum:    resb 1   ; DIR.ASM
00015BE8 <res 00000084>         <1> LongFileName:    resb 132 ; DIR.ASM
                                <1>
                                <1> ;PATH_Array_Ptr: resw 1 ; DIR.ASM
00015C6C <res 00000001>         <1> PATH_CDLevel:       resb 1 ; DIR.ASM
00015C6D <res 00000001>         <1> PATH_Level:  resb 1 ; DIR.ASM
                                <1>
                                <1> ; 07/02/2016
00015C6E <res 0000000D>         <1> Dir_File_Name:     resb 13 ; DIR.ASM ; 09/10/2011
                                <1>
                                <1> ; 10/02/2016
00015C7B <res 0000000D>         <1> Dir_Entry_Name:    resb 13 ; DIR.ASM
                                <1>
                                <1> alignb 2
                                <1>
00015C88 <res 00000002>         <1> AttributesMask: resw 1 ; CMD_INTR.ASM ; 09/11/2011
                                <1>
                                <1> ; 10/02/2016 (128 bytes -> 126 bytes)
                                <1> ; 08/02/2016
                                <1> ;FFF Structure (128 bytes) ; DIR.ASM ; 09/10/2011
00015C8A <res 00000001>         <1> FindFile_Drv:              resb 1
00015C8B <res 00000041>         <1> FindFile_Directory:        resb 65
00015CCC <res 0000000D>         <1> FindFile_Name:             resb 13
                                <1> FindFile_LongNameEntryLength:
00015CD9 <res 00000001>         <1> FindFile_LongNameYes:      resb 1 ; Sign for longname procedures
                                <1> ;Above 80 bytes form
                                <1> ;TR-DOS Source/Destination File FullName Format/Structure
00015CDA <res 00000002>         <1> FindFile_AttributesMask:  resw 1
00015CDC <res 00000020>         <1> FindFile_DirEntry:   resb 32
00015CFC <res 00000004>         <1> FindFile_DirFirstCluster: resd 1
00015D00 <res 00000004>         <1> FindFile_DirCluster:       resd 1
00015D04 <res 00000002>         <1> FindFile_DirEntryNumber:  resw 1
00015D06 <res 00000002>         <1> FindFile_MatchCounter:     resw 1
00015D08 <res 00000002>         <1> FindFile_Reserved:   resw 1 ; 06/03/2016
                                <1>
00015D0A <res 00000004>         <1> First_Path_Pos: resd 1    ; DIR.ASM ; 09/10/2011
00015D0E <res 00000004>         <1> Last_Slash_Pos: resd 1    ; DIR.ASM
                                <1>
                                <1> ; 10/02/2016
00015D12 <res 00000002>         <1> File_Count:     resw 1    ; DIR.ASM ; 09/10/2011
00015D14 <res 00000002>         <1> Dir_Count:      resw 1
00015D16 <res 00000004>         <1> Total_FSize:    resd 1
00015D1A <res 00000004>         <1> TFS_Dec_Begin:  resd 1
00015D1E <res 0000000A>         <1>                 resb 10
00015D28 <res 00000001>         <1> TFS_Dec_End:    resb 1
                                <1>
00015D29 <res 00000001>         <1> PrintDir_RowCounter: resb 1
                                <1>
00015D2A <res 00000002>         <1> alignb 4
                                <1> ; 15/02/2015 ('show' command variables)
00015D2C <res 00000004>         <1> Show_FDT:    resd 1
00015D30 <res 00000004>         <1> Show_LDDDT:  resd 1
00015D34 <res 00000004>         <1> Show_Cluster:      resd 1
00015D38 <res 00000004>         <1> Show_FileSize:     resd 1
00015D3C <res 00000004>         <1> Show_FilePointer: resd 1
00015D40 <res 00000002>         <1> Show_ClusterPointer: resw 1
00015D42 <res 00000002>         <1> Show_ClusterSize: resw 1
00015D44 <res 00000001>         <1> Show_RowCount:     resb 1
                                <1>
00015D45 <res 00000003>         <1> alignb 4
                                <1> ; 21/02/2016
00015D48 <res 00000004>         <1> DelFile_FNPointer: resd 1 ; ; CMD_INTR.ASM (word) ; 09/11/2011
                                <1> ; 27/02/2016
                                <1> ; DIR.ASM (09/10/2011)
00015D4C <res 00000004>         <1> DelFile_FCluster:  resd 1
00015D50 <res 00000002>         <1> DelFile_EntryCounter:    resw 1
00015D52 <res 00000001>         <1> DelFile_LNEL:            resb 1
00015D53 <res 00000001>         <1> resb 1
                                <1>
                                <1> ; DIR.ASM
00015D54 <res 00000004>         <1> mkdir_DirName_Offset:    resd 1
00015D58 <res 00000004>         <1> mkdir_FFCluster:   resd 1
00015D5C <res 00000004>         <1> mkdir_LastDirCluster:    resd 1
00015D60 <res 00000004>         <1> mkdir_FreeSectors: resd 1
00015D64 <res 00000002>         <1> mkdir_attrib:            resw 1
00015D66 <res 00000001>         <1> mkdir_SecPerClust: resb 1
00015D67 <res 00000001>         <1> mkdir_add_new_cluster:   resb 1
00015D68 <res 0000000D>         <1> mkdir_Name:        resb 13
00015D75 <res 00000002>         <1> resw 1 ; 01/03/2016
                                <1> ; 27/02/2016
00015D77 <res 00000001>         <1> RmDir_MultiClusters:     resb 1
00015D78 <res 00000004>         <1> RmDir_DirEntryOffset:    resd 1 ; 01/03/2016 (word -> dword)
00015D7C <res 00000004>         <1> RmDir_ParentDirCluster: resd 1
00015D80 <res 00000004>         <1> RmDir_DirLastCluster:   resd 1
00015D84 <res 00000004>         <1> RmDir_PreviousCluster:  resd 1
                                <1> ; 22/02/2016
00015D88 <res 00000001>         <1> UPDLMDT_CDirLevel: resb 1
00015D89 <res 00000004>         <1> UPDLMDT_CDirFCluster:    resd 1
                                <1>
00015D8D <res 00000003>         <1> alignb 4
```

```
54553                                  <1> ; DRV_FAT.ASM ; 21/08/2011
54554 00015D90 <res 00000004>          <1> gffc_next_free_cluster:  resd 1
54555 00015D94 <res 00000004>          <1> gffc_first_free_cluster: resd 1
54556 00015D98 <res 00000004>          <1> gffc_last_free_cluster:  resd 1
54557                                  <1>
54558                                  <1> ;29/04/2016
54559                                  <1> Cluster_Index: ; resd 1
54560                                  <1> ; 22/02/2016
54561 00015D9C <res 00000004>          <1> ClusterValue:      resd 1
54562                                  <1> ; 04/03/2016
54563 00015DA0 <res 00000001>          <1> Attributes:  resb 1
54564                                  <1> ;;CFS_error:  resb 1 ;; 01/03/2016
54565 00015DA1 <res 00000001>          <1> resb 1
54566 00015DA2 <res 00000001>          <1> CFS_OPType: resb 1
54567 00015DA3 <res 00000001>          <1> CFS_Drv:    resb 1
54568 00015DA4 <res 00000004>          <1> CFS_CC:         resd 1
54569 00015DA8 <res 00000004>          <1> CFS_FAT32FSINFOSEC: resd 1
54570 00015DAC <res 00000004>          <1> CFS_FAT32FC: resd 1
54571                                  <1>
54572                                  <1> ; 27/02/2016
54573                                  <1> ;alignb 4
54574 00015DB0 <res 00000004>          <1> glc_prevcluster: resd 1 ; DRV_FAT.ASM (21/08/2011)
54575                                  <1> ; 22/10/2016
54576 00015DB4 <res 00000004>          <1> glc_index:   resd 1 ;  Last Cluster Index (22/10/2016)
54577                                  <1>
54578                                  <1> ; DIR.ASM
54579 00015DB8 <res 00000002>          <1> DLN_EntryNumber: resw 1
54580 00015DBA <res 00000001>          <1> DLN_40h:     resb 1
54581                                  <1> ; 28/02/2016
54582 00015DBB <res 00000001>          <1> TCC_FATErr:  resb 1 ; DRV_FAT.ASM
54583                                  <1>
54584                                  <1> alignb 4
54585                                  <1> ; DIR.ASM (09/10/2011)
54586 00015DBC <res 00000002>          <1> LCDE_EntryIndex: resw 1 ; LCDE_EntryOffset
54587 00015DBE <res 00000002>          <1> LCDE_ClusterSN:  resw 1
54588 00015DC0 <res 00000004>          <1> LCDE_Cluster:       resd 1
54589 00015DC4 <res 00000004>          <1> LCDE_ByteOffset: resd 1
54590                                  <1>
54591                                  <1> ;alignb4
54592                                  <1> ; 06/03/2016 (word -> dword)
54593                                  <1> ; CMD_INTR.ASM (01/08/2010)
54594 00015DC8 <res 00000004>          <1> SourceFilePath:         resd 1
54595 00015DCC <res 00000004>          <1> DestinationFilePath: resd 1
54596                                  <1>
54597                                  <1> ;alignb 4
54598                                  <1> ; 06/03/2016
54599                                  <1> ; FILE.ASM (09/10/2011)
54600                                  <1> ;Source File Structure (same with 'Find File' Structure)
54601 00015DD0 <res 00000001>          <1> SourceFile_Drv:                 resb 1
54602 00015DD1 <res 00000041>          <1> SourceFile_Directory:           resb 65
54603 00015E12 <res 0000000D>          <1> SourceFile_Name:          resb 13
54604                                  <1> SourceFile_LongNameEntryLength:
54605 00015E1F <res 00000001>          <1> SourceFile_LongNameYes:         resb 1 ; Sign for longname procedures
54606                                  <1> ;Above 80 bytes
54607                                  <1> ;is TR-DOS Source File FullName Format/Structure
54608 00015E20 <res 00000002>          <1> SourceFile_AttributesMask:      resw 1
54609 00015E22 <res 00000020>          <1> SourceFile_DirEntry:            resb 32
54610 00015E42 <res 00000004>          <1> SourceFile_DirFirstCluster:     resd 1
54611 00015E46 <res 00000004>          <1> SourceFile_DirCluster:          resd 1
54612 00015E4A <res 00000002>          <1> SourceFile_DirEntryNumber:      resw 1
54613 00015E4C <res 00000002>          <1> SourceFile_MatchCounter:  resw 1
54614                                  <1> ; 16/03/2016
54615 00015E4E <res 00000001>          <1> SourceFile_SecPerClust:         resb 1
54616 00015E4F <res 00000001>          <1> SourceFile_Reserved:            resb 1
54617                                  <1> ; Above is 128 bytes
54618                                  <1>
54619                                  <1> ;Destination File Structure (same with 'Find File' Structure)
54620 00015E50 <res 00000001>          <1> DestinationFile_Drv:            resb 1
54621 00015E51 <res 00000041>          <1> DestinationFile_Directory:      resb 65
54622 00015E92 <res 0000000D>          <1> DestinationFile_Name:           resb 13
54623                                  <1> DestinationFile_LongNameEntryLength:
54624 00015E9F <res 00000001>          <1> DestinationFile_LongNameYes:    resb 1 ; Sign for longname procedures
54625                                  <1> ;Above 80 bytes
54626                                  <1> ;is TR-DOS Destination File FullName Format/Structure
54627 00015EA0 <res 00000002>          <1> DestinationFile_AttributesMask: resw 1
54628 00015EA2 <res 00000020>          <1> DestinationFile_DirEntry: resb 32
54629 00015EC2 <res 00000004>          <1> DestinationFile_DirFirstCluster: resd 1
54630 00015EC6 <res 00000004>          <1> DestinationFile_DirCluster:     resd 1
54631 00015ECA <res 00000002>          <1> DestinationFile_DirEntryNumber: resw 1
54632 00015ECC <res 00000002>          <1> DestinationFile_MatchCounter:   resw 1
54633                                  <1> ; 16/03/2016
54634 00015ECE <res 00000001>          <1> DestinationFile_SecPerClust:    resb 1
54635 00015ECF <res 00000001>          <1> DestinationFile_Reserved: resb 1
54636                                  <1> ; Above is 128 bytes
54637                                  <1>
54638                                  <1> ; 24/04/2016
54639 00015ED0 <res 00000002>          <1> resw 1
54640                                  <1>
54641                                  <1> ; 10/03/2016
54642                                  <1> ; FILE.ASM
54643 00015ED2 <res 00000001>          <1> move_cmd_phase:       resb 1
54644 00015ED3 <res 00000001>          <1> msftdf_sf_df_drv:  resb 1
54645 00015ED4 <res 00000004>          <1> msftdf_drv_offset: resd 1
54646                                  <1>
54647                                  <1> ; 11/03/2016
54648                                  <1> ; DRV_FAT.ASM (21/08/2011)
54649 00015ED8 <res 00000004>          <1> FAT_anc_LCluster:  resd 1
54650 00015EDC <res 00000004>          <1> FAT_anc_FFCluster: resd 1
54651                                  <1>
54652                                  <1> ;alignb 4
54653                                  <1>
54654                                  <1> ; 14/03/2016
54655                                  <1> ; TRDOS 386 = TRDOS v2.0 feature only !
```

```
                                <1> ; 'allocate_memory_block' in 'memory.s'
00015EE0 <res 00000004>         <1> mem_ipg_count:     resd 1 ; page count (for contiguous allocation)
00015EE4 <res 00000004>         <1> mem_pg_count:      resd 1 ; page count (for count down)
00015EE8 <res 00000004>         <1> mem_aperture:      resd 1 ; contiguous free pages (current)
00015EEC <res 00000004>         <1> mem_max_aperture: resd 1 ; maximum value of contiguous free pages
00015EF0 <res 00000004>         <1> mem_pg_pos:  resd 1 ; mem. position (page #) of current aperture
00015EF4 <res 00000004>         <1> mem_max_pg_pos: resd 1 ; mem. position (page #) of max. aperture
                                <1>
                                <1> ; 15/03/2016
                                <1> ; FILE.ASM ('copy_source_file_to_destination_file')
00015EF8 <res 00000001>         <1> copy_cmd_phase:       resb 1
00015EF9 <res 00000001>         <1> csftdf_rw_err:           resb 1
00015EFA <res 00000001>         <1> DestinationFileFound: resb 1
00015EFB <res 00000001>         <1> csftdf_cdrv:             resb 1
00015EFC <res 00000004>         <1> csftdf_filesize:      resd 1
                                <1> ; TRDOS386 (TRDOS v2.0)
00015F00 <res 00000004>         <1> csftdf_sf_mem_addr:   resd 1
00015F04 <res 00000004>         <1> csftdf_sf_mem_bsize:  resd 1
                                <1> ;
                                <1>
00015F08 <res 00000004>         <1> csftdf_sf_cluster:    resd 1 ; 16/03/2016
00015F0C <res 00000004>         <1> csftdf_df_cluster:    resd 1
                                <1> ; 16/03/2016
00015F10 <res 00000004>         <1> csftdf_r_size:        resd 1
00015F14 <res 00000004>         <1> csftdf_w_size:        resd 1
00015F18 <res 00000004>         <1> csftdf_sf_rbytes:     resd 1
00015F1C <res 00000004>         <1> csftdf_df_wbytes:     resd 1
00015F20 <res 00000001>         <1> csftdf_percentage:    resb 1
                                <1> ; 17/03/2016
00015F21 <res 00000001>         <1> csftdf_videopage:     resb 1
00015F22 <res 00000002>         <1> csftdf_cursorpos:     resw 1
00015F24 <res 00000004>         <1> csftdf_sf_drv_dt:     resd 1
00015F28 <res 00000004>         <1> csftdf_df_drv_dt:     resd 1
                                <1>
                                <1> ; 21/03/2016
                                <1> ; 20/03/2016
                                <1> ; FILE.ASM
00015F2C <res 00000004>         <1> createfile_Name_Offset:  resd 1
00015F30 <res 00000004>         <1> createfile_FreeSectors:  resd 1
00015F34 <res 00000004>         <1> createfile_size:         resd 1
00015F38 <res 00000004>         <1> createfile_FFCluster:    resd 1 ; 11/03/2016
00015F3C <res 00000004>         <1> createfile_LastDirCluster: resd 1
00015F40 <res 00000004>         <1> createfile_Cluster:      resd 1
00015F44 <res 00000004>         <1> createfile_PCluster:     resd 1
00015F48 <res 00000001>         <1> createfile_attrib:  resb 1
00015F49 <res 00000001>         <1> createfile_SecPerClust:  resb 1
00015F4A <res 00000002>         <1> createfile_DirIndex:     resw 1
00015F4C <res 00000004>         <1> createfile_CCount:  resd 1
00015F50 <res 00000002>         <1> createfile_BytesPerSec:   resw 1 ; 23/03/2016
00015F52 <res 00000001>         <1> createfile_wfc:             resb 1
00015F53 <res 00000001>         <1> createfile_UpdatePDir:    resb 1 ; 31/03/2016
                                <1>
                                <1> ;alignb 4
                                <1>
                                <1> ; 11/04/2016
00015F54 <res 00000002>         <1> env_var_length:       resw 1
                                <1>
00015F56 <res 00000002>         <1> alignb 4
                                <1>
                                <1> ; 25/04/2016
00015F58 <res 00000001>         <1> readi.valid: resb 1 ; valid data (>0 = valid for readi)
00015F59 <res 00000001>         <1> readi.drv:   resb 1 ; drive number (0, 1,2,3,4..)
00015F5A <res 00000001>         <1> readi.spc:   resb 1 ; sectors per cluster for 'readi' drive
00015F5B <res 00000001>         <1> readi.s_index:  resb 1 ; sector index in current cluster (buffer)
00015F5C <res 00000004>         <1> readi.sector:      resd 1 ; current disk sector
00015F60 <res 00000002>         <1> readi.bpc:   resw 1 ; bytes per cluster - 1
00015F62 <res 00000002>         <1> readi.offset:      resw 1 ; byte offset in cluster buffer
00015F64 <res 00000004>         <1> readi.cluster:  resd 1 ; current cluster number
00015F68 <res 00000004>         <1> readi.c_index:     resd 1 ; cluster index of the current cluster (0,1,2,3..)
00015F6C <res 00000004>         <1> readi.fclust:      resd 1 ; first cluster of the current cluster
00015F70 <res 00000004>         <1> readi.fs_index: resd 1 ; sector index in disk/file section (for Singlix FS)
                                <1> ;readi.buffer:     resd 1 ; readi sector buffer address
                                <1>
                                <1> ;alignb 4
                                <1>
00015F74 <res 00000001>         <1> writei.valid:      resb 1 ; valid data (>0 = valid for writei)
00015F75 <res 00000001>         <1> writei.drv:  resb 1 ; drive number (0, 1,2,3,4..)
00015F76 <res 00000001>         <1> writei.spc:  resb 1 ; sectors per cluster for 'writei' drive
00015F77 <res 00000001>         <1> writei.s_index: resb 1 ; sector index in current cluster (buffer)
00015F78 <res 00000004>         <1> writei.sector:     resd 1 ; current disk sector
00015F7C <res 00000002>         <1> writei.bpc:  resw 1 ; bytes per cluster - 1
00015F7E <res 00000002>         <1> writei.offset:     resw 1 ; byte offset in cluster buffer
00015F80 <res 00000004>         <1> writei.cluster: resd 1 ; current cluster number
00015F84 <res 00000004>         <1> writei.c_index:    resd 1 ; cluster index of the current cluster (0,1,2,3..)
00015F88 <res 00000004>         <1> writei.fclust:  resd 1 ; first cluster of the current cluster
00015F8C <res 00000004>         <1> writei.fs_index: resd 1 ; sector index in disk/file section (for Singlix FS)
                                <1> ;writei.buffer:    resd 1 ; writei sector buffer address
00015F90 <res 00000004>         <1> writei.lclust:     resd 1 ; writei last cluster (mget_w) ; 23/10/2016
00015F94 <res 00000004>         <1> writei.l_index:    resd 1 ; writei last cluster index (mget_w) ; 23/10/2016
00015F98 <res 00000001>         <1> writei.ofn:  resb 1 ; open file number (to be written) ; 23/10/2016
                                <1>
00015F99 <res 00000003>         <1> alignb 4
                                <1>
                                <1> ; 29/04/2016
00015F9C <res 00000004>         <1> Run_CDirFC:  resd 1
00015FA0 <res 00000001>         <1> Run_Auto_Path:     resb 1
00015FA1 <res 00000001>         <1> Run_Manual_Path: resb 1 ; 0 -> auto path sequence needed
00015FA2 <res 00000001>         <1> EXE_ID:            resb 1
00015FA3 <res 00000001>         <1> EXE_dot:    resb 1
                                <1>
                                <1> ; 06/05/2016
00015FA4 <res 00000004>         <1> mainprog_return_addr: resd 1
00015FA8 <res 00000004>         <1> last_error:  resd 1  ; this will be used to return error code to MainProg
```

```
                                <1>                     ; 'lasterror' keyword will be used later to get the
                                <1>                     ; last error code/number/status.
                                <1> ; 12/05/2016
00015FAC <res 00000004>         <1> video_eax:   resd 1  ; eax return value of video function
                                <1>
                                <1> ; 01/06/2016
00015FB0 <res 00000004>         <1> user_buffer: resd 1  ; 'diskio.s' (INT 33h, Function 08h, floppy disk type)
                                <1>
                                <1> ; 21/05/2016 - TRDOS 386 ('swap/switch', 'rswap', [u.pri])
00015FB4 <res 00000001>         <1> priority:    resb 1  ; running priority level of process (0,1,2)
                                <1>                     ; (run queue which is process comes from)
                                <1> ; 22/05/2016 - TRDOS 386 ('set_run_sequence', 'rtc_int', 'u_timer')
00015FB5 <res 00000001>         <1> p_change:    resb 1  ; process change status (for timer events)
                                <1> ; 23/05/2016 - TRDOS 386 ('clock')
00015FB6 <res 00000001>         <1> multi_tasking:     resb 1   ; Multi Tasking status (0 = disabled, >0 = enabled)
                                <1>                     ; (EBX will return with user buffer addr or disk type)
                                <1> ; 07/06/2016
00015FB7 <res 00000001>         <1> timer_events:      resb 1  ; number of (active) timer events, <= 16
                                <1>
                                <1> ; 24/06/2016
00015FB8 <res 00000001>         <1> w_str_cmd:   resb 1 ; WRITE_STRING command (0,1,2,3) ; video.s
00015FB9 <res 00000001>         <1> p_crt_mode:  resb 1  ; previous video mode (=3 or 0), backup mark/sign
                                <1> ; 26/06/2016
00015FBA <res 00000001>         <1> p_crt_page:  resb 1  ; previous active page (for 'set_mode')
                                <1> ; 04/07/2016
00015FBB <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
00015FBC <res 00000002>         <1> CRT_LEN:     resw 1  ; length of regen buffer in bytes
00015FBE <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)
00015FCE <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
00015FD2 <res 00000001>         <1> VGA_MTYPE:   resb 1  ; 0=CTEXT,1=MTEXT,2=CGA,3=PLANAR1,4=PLANAR4,5=LINEAR
                                <1>
                                <1> ; 23/10/2016
00015FD3 <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
00015FD4 <res 00000004>         <1> FFF_UBuffer: resd 1  ; User's buffer address for FFF & FNF system calls
                                <1> ; 15/10/2016
00015FD8 <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)
00015FD9 <res 00000001>         <1> FFF_Attrib:  resb 1 ; Find First File attributes for Find Next File (LB)
00015FDA <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)
00015FDB <res 00000001>         <1> SWP_inv_fname:     resb 1 ; Set Working Path - Invalid File Name
00015FDC <res 00000002>         <1> SWP_Mode:    resw 1 ; Set Working Path - Mode
00015FDE <res 00000001>         <1> SWP_DRV:     resb 1 ; Set Working Path - Drive
00015FDF <res 00000001>         <1> SWP_DRV_chg: resb 1 ; Set Working Path - Drive Change
                                <1>
                                <1> ; 27/02/2017
00015FE0 <res 00000001>         <1> fpready:     resb 1 ; '80387 fpu is ready' flag
                                <1>
                                <1> ; 08/10/2016
00015FE1 <res 00000009>         <1> device_name:    resb 9  ; capitalized (and zero padded) device canem
                                <1>                     ; (example: "TTY0",0,0,0,0,0")
                                <1>
00015FEA <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)
00015FEC <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>
0001600C <res 00000040>             <1> IDEV_NAME:  resb 8*NUMIDEV
                                    <1>                     ; 8 byte names of installable device drivers
                                    <1>
0001604C <res 00000008>             <1> IDEV_TYPE:  resb NUMIDEV ; Driver type of installable device drivers
00016054 <res 00000008>             <1> IDEV_FLAGS: resb NUMIDEV ; Device access parameters for installable
                                    <1>                          ; device drivers (These values are set while
                                    <1>                       ; the device driver is being loaded.)
0001605C <res 00000020>             <1> IDEV_OADDR: resd NUMIDEV ; open function addr for installable dev driver
0001607C <res 00000020>             <1> IDEV_CADDR: resd NUMIDEV ; close function addr for installable dev driver
0001609C <res 00000020>             <1> IDEV_RADDR: resd NUMIDEV ; read function addr for installable dev driver
000160BC <res 00000020>             <1> IDEV_WADDR: resd NUMIDEV ; write function addr for installable dev driver
                                    <1>
                                    <1> ; 08/10/2016
                                    <1> ; 07/10/2016
                                    <1> ; Device Open and Access parameters
000160DC <res 0000001E>             <1> DEV_ACCESS:   resb NUMOFDEVICES   ; bit 0 = accessable by normal users
                                    <1>                                   ; bit 1 = read access permission
                                    <1>                                   ; bit 2 = write access permission
                                    <1>                                   ; bit 3 = IOCTL permission to users
                                    <1>                                   ; bit 4 = block device if it is set
                                    <1>                                   ; bit 5 = 16 bit or 1024 byte data
                                    <1>                                   ; bit 6 = 32 bit or 2048 byte data
                                    <1>                                   ; bit 7 = installable device driver
000160FA <res 0000001E>             <1> DEV_R_OWNER:  resb NUMOFDEVICES   ; Reading owner no (u.uid) of devices
00016118 <res 0000001E>             <1> DEV_R_OPENCOUNT: resb NUMOFDEVICES   ; Reading open count
00016136 <res 0000001E>             <1> DEV_W_OWNER:  resb NUMOFDEVICES   ; Writing owner no (u.uid) of devices
00016154 <res 0000001E>             <1> DEV_W_OPENCOUNT: resb NUMOFDEVICES   ; Writing open count
00016172 <res 0000001E>             <1> DEV_DRIVER:   resb NUMOFDEVICES   ; device driver number (1 to 7Fh)
                                    <1>                                   ; *if bit 7 is set (80 to FFh)
                                    <1>                                   ; *if it is installable device driver
                                    <1>                                   ; *index (0 to 7Fh)
                                    <1>                                   ; otherwise it is kernel device index
00016190 <res 0000001E>             <1> DEV_OPENMODE:       resb NUMOFDEVICES   ; 1 = read mode
                                    <1>                                   ; 2 = write mode
                                    <1>                                   ; 3 = read & write
                                    <1>                                   ; 0 = not open (free)
000161AE <res 00000078>             <1> DEV_NAME_PTR:    resd NUMOFDEVICES   ; pointers to name addresses of drivers
                                    <1>                                   ; Address base: KDEV_NAME+
                                    <1>                                   ; or IDEV_NAME+
00016226 <res 00000078>             <1> DEV_R_POINTER:      resd NUMOFDEVICES   ; reading pointer, writing pointer
0001629E <res 00000078>             <1> DEV_W_POINTER:      resd NUMOFDEVICES   ; sector number if block device
                                    <1>                                   ; character offset if char device
00016316 <res 00000002>             <1> alignb 4
                                    <1>
                                    <1> ; 06/10/2016
                                    <1> ; Open File Parameters
00016318 <res 00000028>             <1> OF_FCLUSTER: resd OPENFILES  ; First clusters of open files
00016340 <res 0000000A>             <1> OF_DRIVE:    resb OPENFILES  ; Logical DOS drive numbers of open files
0001634A <res 0000000A>             <1> OF_MODE:     resb OPENFILES  ; Open mode (1 = read, 2 = write, 3 = r&w)
00016354 <res 0000000A>             <1> OF_STATUS:   resb OPENFILES  ; (bit 0 = read, bit 1 = write)
0001635E <res 0000000A>             <1> OF_OPENCOUNT:      resb OPENFILES  ; Open counts of open files
00016368 <res 00000028>             <1> OF_POINTER:  resd OPENFILES      ; File seek/read/write pointer
00016390 <res 00000028>             <1> OF_SIZE:     resd OPENFILES      ; File sizes of open files (in bytes)
000163B8 <res 00000028>             <1> OF_DIRFCLUSTER:    resd OPENFILES  ; Directory First Clusters of open files
000163E0 <res 00000028>             <1> OF_DIRCLUSTER:     resd OPENFILES  ; Directory (Entry) Clusters of open files
00016408 <res 00000028>             <1> OF_VOLUMEID: resd OPENFILES  ; Vol ID for removable drives of open files
00016430 <res 00000028>             <1> OF_CCLUSTER: resd OPENFILES  ; Current clusters of open files
00016458 <res 00000028>             <1> OF_CCINDEX:  resd OPENFILES  ; Cluster index numbers of current clusters
                                    <1> ; 24/10/2016
00016480 <res 00000014>             <1> OF_DIRENTRY: resw OPENFILES  ; Directory entry index no. in dir cluster
                                    <1>                              ; Sector index = entry index / 16
                                    <1> ;alignb 2
                                    <1>
00016494 <res 00000060>             <1> DTA:         resd 24             ; Find First File data transfer area
                                    <1>
                                    <1> ; 19/12/2016
000164F4 <res 00000001>             <1> tcallback:   resb 1       ; Timer callback method flag for 'systimer'
000164F5 <res 00000001>             <1> trtc:        resb 1       ; Timer interrupt type flag for 'systimer'
                                    <1> ; 20/02/2017
000164F6 <res 00000001>             <1> no_page_swap:      resb 1       ; Swap lock for Signal Response Byte pages
                                    <1> ;;15/01/2017
                                    <1> ; 02/01/2017
                                    <1> ;;intflg:    resb 1       ; software interrupt in progress signal
                                    <1>                           ; (for timer interrupt)
                                    <1>
000164F7 <res 00000001>             <1> alignb 4
                                    <1> ; 13/04/2017
000164F8 <res 0000001E>             <1> DEV_INTR:    resb NUMOFDEVICES ; Device Interrupt (IRQ) number + 1
                                    <1>                              ; (0= not available, 1= IRQ 0, 16= IRQ 15)
00016516 <res 00000040>             <1> DEV_INT_HNDLR:     resd 16             ; Device Interrupt Handler addr, if > 0
                                    <1>
                                    <1>
```

```
54965                                  <1> ;alignb 4
54966                                  <1>
54967                                  <1> ; 26/02/2017 ; IRQ Callback parameters ('syscalbac')
54968                                  <1> ;Index: ; 0 to 8
54969                                  <1> ;     0 = IRQ3, 1 = IRQ4, 2 = IRQ5, 3 = IRQ7
54970                                  <1> ;     4 = IRQ9, 5 = IRQ10, 6 = IRQ11, 7 = IRQ12, 8 = IRQ13
54971 00016556 <res 00000009>          <1> IRQ.owner:   resb 9       ; owner, 0 = free, >0 = [u.uno]
54972 0001655F <res 00000009>          <1> IRQ.dev:     resb 9       ; 0 = default/kernel, >0 = device number
54973 00016568 <res 00000009>          <1> IRQ.method:  resb 9             ; 0 = Signal Response Byte, 1 = Callback
54974 00016571 <res 00000009>          <1> IRQ.srb:     resb 9             ; Signal Response/Return Byte value
54975 0001657A <res 00000024>          <1> IRQ.addr:    resd 9       ; Rignal Response Byte address (physical)
54976                                  <1>                           ; or Callback service address (virtual)
54977                                  <1> ; 28/02/2017
54978 0001659E <res 00000004>          <1> IRQ_cr3:     resd 1       ; for saving cr3 register in IRQ handler
54979 000165A2 <res 00000001>          <1> IRQnum:            resb 1       ; IRQ number for IRQ handler (trdosk8.s)
54980                                  <1>
54981                                  <1> ; 10/04/2017
54982                                  <1> ; 03/04/2017
54983                                  <1> ; UNINITIALIZED AUDIO DATA
54984 000165A3 <res 00000001>          <1> alignb 4
54985 000165A4 <res 00000001>          <1> audio_pci:   resb 1
54986 000165A5 <res 00000001>          <1> audio_device:      resb 1
54987 000165A6 <res 00000001>          <1> audio_mode:  resb 1
54988 000165A7 <res 00000001>          <1> audio_intr:  resb 1
54989 000165A8 <res 00000001>          <1> audio_busy:  resb 1  ; Busy flag for audio irq ; 21/04/2017
54990 000165A9 <res 00000001>          <1> audio_reserved: resb 1
54991 000165AA <res 00000002>          <1> audio_io_base:     resw 1       ; Base I/O address of audio device
54992 000165AC <res 00000004>          <1> audio_dev_id:      resd 1 ; BUS/DEV/FN ; 00000000BBBBBBBBDDDDDFFF00000000
54993 000165B0 <res 00000004>          <1> audio_vendor:      resd 1
54994 000165B4 <res 00000004>          <1> audio_stats_cmd: resd 1
54995                                  <1> ;
54996 000165B8 <res 00000004>          <1> audio_buffer:      resd 1 ; virtual address of user's audio buffer
54997 000165BC <res 00000004>          <1> audio_p_buffer:    resd 1 ; Physical address of user's audio buffer
54998 000165C0 <res 00000004>          <1> audio_buff_size: resd 1 ; user's audio buffer size (half buffer size)
54999 000165C4 <res 00000004>          <1> audio_dma_buff: resd 1  ; dma buffer address
55000 000165C8 <res 00000004>          <1> audio_dmabuff_size: resd 1 ; dma buffer size (2 * half buffer size)
55001 000165CC <res 00000001>          <1> audio_flag:  resb 1  ; dma buffer flag (1st half = 0, 2nd half = 1)
55002 000165CD <res 00000001>          <1> audio_user:  resb 1 ; user number of the owner
55003 000165CE <res 00000001>          <1> audio_cb_mode:     resb 1 ; 0 = signal response byte method
55004                                  <1>                           ; 1 = callback method
55005                                  <1>                           ; 2 = s.r.b. method with auto increment
55006 000165CF <res 00000001>          <1> audio_srb:   resb 1 ; signal response byte value
55007 000165D0 <res 00000004>          <1> audio_cb_addr:     resd 1  ; callback service address or s.r.b. address
55008                                  <1>                           ; (s.r.b. addr is physical, cbs addr is virtual)
55009                                  <1>
55010 000165D4 <res 00000001>          <1> audio_bps:   resb 1  ; selected mode: 8 bit, 16 bit
55011 000165D5 <res 00000001>          <1> audio_stmo:  resb 1 ; selected mode: mono /stereo
55012 000165D6 <res 00000002>          <1> audio_freq:  resw 1 ; sampling rate
55013                                  <1>
55014                                  <1> ; 21/04/2017
55015 000165D8 <res 00000001>          <1> audio_play_cmd: resb 1  ; Play/Stop command (1 = play, 0 = stop)
55016                                  <1> audio_civ: ; 28/05/2017 ; Current Buffer Index (AC'97)
55017 000165D9 <res 00000001>          <1> audio_flag_eol:    resb 1  ; End of Link status (vt8233, EOL/FLAG)
55018                                  <1>
55019                                  <1> audio_master_volume:
55020 000165DA <res 00000001>          <1> audio_master_volume_l: resb 1 ; sound volume (lineout) left channel
55021 000165DB <res 00000001>          <1> audio_master_volume_r: resb 1 ; sound volume (lineout) right channel
55022                                  <1>
55023                                  <1> alignb 4
55024                                  <1> ; 28/05/2017
55025                                  <1> ; AC'97 Audio Controller Base Adress Registers
55026 000165DC <res 00000002>          <1> NAMBAR:            resw 1 ; Native Audio Mixer Base Address
55027 000165DE <res 00000002>          <1> NABMBAR:    resw 1 ; Native Audio Bus Mastering Base Address
55028                                  <1>
55029                                  <1> ;alignb 4
55030                                  <1> ; 21/04/2017
55031 000165E0 <res 00000400>          <1> audio_bdl_buff:    resd 32*8  ; VT8233 (AC97) BDL Buffer Size
55032                                  <1> ; 12/05/2017
55033 000169E0 <res 00000004>          <1> base_addr:   resd 1 ; 'direct_memory_access' (memory.s)
55034                                  <1>
55035                                  <1> ; 28/08/2017
55036                                  <1> ; 20/08/2017
55037 000169E4 <res 00000001>          <1>             resb 1  ;
55038 000169E5 <res 00000001>          <1> dma_user:    resb 1 ; user number for sysdma
55039 000169E6 <res 00000001>          <1> dma_channel: resb 1 ; dma channel for sysdma
55040 000169E7 <res 00000001>          <1> dma_mode:    resb 1  ; dma mode for sysdma
55041 000169E8 <res 00000004>          <1> dma_addr:    resd 1 ; dma buffer physical addr for sysdma
55042 000169EC <res 00000004>          <1> dma_size:    resd 1  ; dma buffer size (in bytes) for sysdma
55043 000169F0 <res 00000004>          <1> dma_start:   resd 1  ; dma start address for sysdma
55044 000169F4 <res 00000004>          <1> dma_count:   resd 1  ; dma count (in bytes) for sysdma
55045                                  <1>
55046 000169F8 <res 00009608>          <1> alignb 65536
55047                                  <1> ; 09/08/2017
55048                                  <1> ; 12/05/2017
55049 00020000 <res 00010000>          <1> sb16_dma_buffer: resb 65536 ; DMA buffer for sb16 audio playing.
55050                                      ; 24/01/2016
55051                                      %include 'ubss.s'  ; UNINITIALIZED KERNEL (USER) DATA
55052                                  <1> ; ****************************************************************************
55053                                  <1> ; TRDOS386.ASM (TRDOS 386 Kernel - v2.0.0) - UNINITIALIZED USER DATA : ubss.s
55054                                  <1> ; ----------------------------------------------------------------------------
55055                                  <1> ; Last Update: 28/02/2017
55056                                  <1> ; ----------------------------------------------------------------------------
55057                                  <1> ; Beginning: 24/01/2016
55058                                  <1> ; ----------------------------------------------------------------------------
55059                                  <1> ; Assembler: NASM version 2.11 (trdos386.s)
55060                                  <1> ; ----------------------------------------------------------------------------
55061                                  <1> ; Derived from 'Retro UNIX 386 Kernel - v0.2.1.0' source code by Erdogan Tan
55062                                  <1> ; ux.s (04/12/2015)
55063                                  <1> ; ****************************************************************************
55064                                  <1>
55065                                  <1> ; Retro UNIX 386 v1 Kernel - ux.s
55066                                  <1> ; Last Modification: 04/12/2015
55067                                  <1> ;
```

```
55068                                  <1> ; ///////// RETRO UNIX 386 V1 SYSTEM DEFINITIONS //////////////
55069                                  <1> ; (Modified from
55070                                  <1> ;     Retro UNIX 8086 v1 system definitions in 'UNIX.ASM', 01/09/2014)
55071                                  <1> ; ((UNIX.ASM (RETRO UNIX 8086 V1 Kernel), 11/03/2013 - 01/09/2014))
55072                                  <1> ; ----------------------------------------------------------------------------
55073                                  <1> ; Derived from UNIX Operating System (v1.0 for PDP-11)
55074                                  <1> ; (Original) Source Code by Ken Thompson (1971-1972)
55075                                  <1> ; <Bell Laboratories (17/3/1972)>
55076                                  <1> ; <Preliminary Release of UNIX Implementation Document>
55077                                  <1> ; (Section E10 (17/3/1972) - ux.s)
55078                                  <1> ; ****************************************************************************
55079                                  <1>
55080                                  <1> alignb 2
55081                                  <1>
55082                                  <1> inode:
55083                                  <1>       ; 11/03/2013.
55084                                  <1>       ;Derived from UNIX v1 source code 'inode' structure (ux).
55085                                  <1>       ;i.
55086                                  <1>
55087 00030000 <res 00000002>        <1>       i.flgs:       resw 1
55088 00030002 <res 00000001>        <1>       i.nlks:       resb 1
55089 00030003 <res 00000001>        <1>       i.uid:  resb 1
55090                                  <1>         ;i.size:  resw 1 ; size
55091 00030004 <res 00000002>        <1>       resw 1 ; 29/04/2016
55092 00030006 <res 00000010>        <1>       i.dskp:       resw 8 ; 16 bytes
55093 00030016 <res 00000004>        <1>       i.ctim:       resd 1
55094 0003001A <res 00000004>        <1>       i.mtim:       resd 1
55095 0003001E <res 00000002>        <1>       i.rsvd:  resw 1 ; Reserved (ZERO/Undefined word for UNIX v1.)
55096                                  <1>
55097                                  <1> I_SIZE      equ $ - inode
55098                                  <1>
55099                                  <1> process:
55100                                  <1>       ; 19/12/2016
55101                                  <1>       ; 21/05/2016
55102                                  <1>       ; 19/05/2016 - TRDOS 386 (TRDOS v2.0)
55103                                  <1>       ; 06/05/2015 - Retro UNIX 386 v1
55104                                  <1>       ; 11/03/2013 - 05/02/2014 (Retro UNIX 8086 v1)
55105                                  <1>       ;Derived from UNIX v1 source code 'proc' structure (ux).
55106                                  <1>       ;p.
55107                                  <1>
55108 00030020 <res 00000020>        <1>         p.pid:   resw nproc
55109 00030040 <res 00000020>        <1>         p.ppid:  resw nproc
55110 00030060 <res 00000020>        <1>         p.break: resw nproc
55111 00030080 <res 00000010>        <1>         p.ttyc:  resb nproc ; console tty in Retro UNIX 8086 v1.
55112 00030090 <res 00000010>        <1>       p.waitc: resb nproc ; waiting channel in Retro UNIX 8086 v1.
55113 000300A0 <res 00000010>        <1>       p.link:       resb nproc
55114 000300B0 <res 00000010>        <1>       p.stat:       resb nproc
55115                                  <1>
55116                                  <1>       ; 06/05/2015 (Retro UNIX 386 v1 feature only !)
55117 000300C0 <res 00000040>        <1>       p.upage: resd nproc ; Physical address of the process's
55118                                  <1>                           ; 'user' structure
55119                                  <1>       ; 21/05/2016
55120                                  <1>       ; 19/05/2016 (TRDOS 386 feature only!)
55121 00030100 <res 00000010>        <1>       p.timer: resb nproc ; number of timer events of the processs
55122                                  <1>
55123                                  <1>       ; 19/12/2016
55124 00030110 <res 00000040>        <1>       p.tcb: resd nproc ; timer callback service address (if > 0)
55125                                  <1>
55126                                  <1> P_SIZE      equ $ - process
55127                                  <1>
55128                                  <1> ; fsp table (original UNIX v1)
55129                                  <1> ;
55130                                  <1> ;Entry
55131                                  <1> ;          15                                      0
55132                                  <1> ;  1     |---|--------------------------------------|
55133                                  <1> ;        |r/w|       i-number of open file          |
55134                                  <1> ;        |---|--------------------------------------|
55135                                  <1> ;        |               device number              |
55136                                  <1> ;        |------------------------------------------|
55137                                  <1> ;    (*) | offset pointer, i.e., r/w pointer to file |
55138                                  <1> ;        |------------------------------------------|
55139                                  <1> ;        |  flag that says    | number of processes  |
55140                                  <1> ;        |   file deleted     | that have file open  |
55141                                  <1> ;        |------------------------------------------|
55142                                  <1> ;  2     |                                          |
55143                                  <1> ;        |------------------------------------------|
55144                                  <1> ;        |                                          |
55145                                  <1> ;        |------------------------------------------|
55146                                  <1> ;        |                                          |
55147                                  <1> ;        |------------------------------------------|
55148                                  <1> ;        |                                          |
55149                                  <1> ;        |------------------------------------------|
55150                                  <1> ;  3     |                                          |
55151                                  <1> ;        |                                          |
55152                                  <1> ;
55153                                  <1> ; (*) Retro UNIX 386 v1 modification: 32 bit offset pointer
55154                                  <1>
55155                                  <1>
55156                                  <1> ; 15/04/2015
55157 00030150 <res 000001F4>        <1> fsp:   resb nfiles * 10 ; 11/05/2015 (8 -> 10)
55158 00030344 <res 00000002>        <1> idev:  resw 1 ; device number is 1 byte in Retro UNIX 8086 v1 !
55159 00030346 <res 00000002>        <1> cdev:    resw 1 ; device number is 1 byte in Retro UNIX 8086 v1 !
55160                                  <1> ; 18/05/2015
55161                                  <1> ; 26/04/2013 device/drive parameters (Retro UNIX 8086 v1 feature only!)
55162                                  <1> ; 'UNIX' device numbers (as in 'cdev' and 'u.cdrv')
55163                                  <1> ;     0 -> root device (which has Retro UNIX 8086 v1 file system)
55164                                  <1> ;     1 -> mounted device (which has Retro UNIX 8086 v1 file system)
55165                                  <1> ; 'Retro UNIX 8086 v1' device numbers: (for disk I/O procedures)
55166                                  <1> ;     0 -> fd0 (physical drive, floppy disk 1), physical drive number = 0
55167                                  <1> ;     1 -> fd1 (physical drive, floppy disk 2), physical drive number = 1
55168                                  <1> ;     2 -> hd0 (physical drive, hard disk 1), physical drive number = 80h
55169                                  <1> ;     3 -> hd1 (physical drive, hard disk 2), physical drive number = 81h
55170                                  <1> ;     4 -> hd2 (physical drive, hard disk 3), physical drive number = 82h
```

```
55171                                  <1> ;     5 -> hd3 (physical drive, hard disk 4), physical drive number = 83h
55172 00030348 <res 00000001>          <1> rdev:  resb 1 ; root device number ; Retro UNIX 8086 v1 feature only!
55173                                  <1>               ; as above, for physical drives numbers in following table
55174 00030349 <res 00000001>          <1> mdev:  resb 1 ; mounted device number ; Retro UNIX 8086 v1 feature only!
55175                                  <1> ; 15/04/2015
55176 0003034A <res 00000001>          <1> active:      resb 1
55177 0003034B <res 00000001>          <1>        resb 1 ; 09/06/2015
55178 0003034C <res 00000002>          <1> mnti:  resw 1
55179 0003034E <res 00000002>          <1> mpid:  resw 1
55180 00030350 <res 00000002>          <1> rootdir: resw 1
55181                                  <1>
55182                                  <1> ; 21/05/2016 - TRDOS 386 (TRDOS v2.0) - priority levels, 3 run queues
55183                                  <1> runq:
55184 00030352 <res 00000002>          <1> runq_event:   resw 1 ; high priority, 'run for event'            ; 2
55185 00030354 <res 00000002>          <1> runq_normal:  resw 1 ; normal/regular priority, 'run as requlat' ; 1
55186 00030356 <res 00000002>          <1> runq_background: resw 1 ; low priority, 'run on background'         ; 0
55187                                  <1> ;
55188 00030358 <res 00000001>          <1> imod:  resb 1
55189 00030359 <res 00000001>          <1> smod:  resb 1
55190 0003035A <res 00000001>          <1> mmod:  resb 1
55191 0003035B <res 00000001>          <1> sysflg:      resb 1
55192                                  <1>
55193                                  <1> alignb 4
55194                                  <1>
55195                                  <1> user:
55196                                  <1>       ; 13/01/2017
55197                                  <1>       ; 19/12/2016
55198                                  <1>       ; 21/05/2016 - TRDOS 386 (TRDOS v2.0)
55199                                  <1>       ;            [u.pri] usage method modification
55200                                  <1>       ; 04/12/2015
55201                                  <1>       ; 18/10/2015
55202                                  <1>       ; 12/10/2015
55203                                  <1>       ; 21/09/2015
55204                                  <1>       ; 24/07/2015
55205                                  <1>       ; 16/06/2015
55206                                  <1>       ; 09/06/2015
55207                                  <1>       ; 11/05/2015
55208                                  <1>       ; 16/04/2015 (Retro UNIX 386 v1 - 32 bit modifications)
55209                                  <1>       ; 10/10/2013
55210                                  <1>       ; 11/03/2013.
55211                                  <1>       ;Derived from UNIX v1 source code 'user' structure (ux).
55212                                  <1>       ;u.
55213                                  <1>
55214 0003035C <res 00000004>          <1>       u.sp:    resd 1 ; esp (kernel stack at the beginning of 'sysent')
55215 00030360 <res 00000004>          <1>       u.usp:   resd 1 ; esp (kernel stack points to user's registers)
55216 00030364 <res 00000004>          <1>       u.r0:    resd 1 ; eax
55217 00030368 <res 00000002>          <1>       u.cdir:        resw 1
55218 0003036A <res 0000000A>          <1>       u.fp:    resb 10
55219 00030374 <res 00000004>          <1>       u.fofp:        resd 1
55220 00030378 <res 00000004>          <1>       u.dirp:        resd 1
55221 0003037C <res 00000004>          <1>       u.namep:  resd 1
55222 00030380 <res 00000004>          <1>       u.off:   resd 1
55223 00030384 <res 00000004>          <1>       u.base:        resd 1
55224 00030388 <res 00000004>          <1>       u.count:  resd 1
55225 0003038C <res 00000004>          <1>       u.nread:  resd 1
55226 00030390 <res 00000004>          <1>       u.break:  resd 1 ; break
55227 00030394 <res 00000002>          <1>       u.ttyp:        resw 1
55228                                  <1>       ; 10/01/2017 (TRDOS 386, relocation and dword alignment)
55229                                  <1>       ; tty number (rtty, rcvt, wtty)
55230 00030396 <res 00000001>          <1>       u.ttyn:        resb 1 ; 28/07/2013 - Retro Unix 8086 v1 feature only !
55231 00030397 <res 00000001>          <1>       u.resb:   resb 1 ; 10/01/2017 (TRDOS 386, temporary)
55232 00030398 <res 00000010>          <1>       u.dirbuf: resb 16 ; 04/12/2015 (10 -> 16)
55233                                  <1>       ;u.pri:        resw 1 ; 14/02/2014
55234 000303A8 <res 00000001>          <1>       u.quant:  resb 1 ; Retro UNIX 8086 v1 Feature only ! (uquant)
55235 000303A9 <res 00000001>          <1>       u.pri:   resb 1 ; Modification: 21/05/2016 (priority levels: 0, 1, 2)
55236 000303AA <res 00000002>          <1>       u.intr:        resw 1
55237 000303AC <res 00000002>          <1>       u.quit:        resw 1
55238                                  <1>       ;u.emt:        resw 1 ; 10/10/2013
55239                                  <1>       ;u.ilgins: resw 1 ; 10/01/2017
55240 000303AE <res 00000002>          <1>       u.cdrv:        resw 1 ; cdev
55241 000303B0 <res 00000001>          <1>       u.uid:   resb 1 ; uid
55242 000303B1 <res 00000001>          <1>       u.ruid:        resb 1
55243 000303B2 <res 00000001>          <1>       u.bsys:        resb 1
55244 000303B3 <res 00000001>          <1>       u.uno:   resb 1
55245 000303B4 <res 00000004>          <1>         u.upage:  resd 1 ; 16/04/2015 - Retro Unix 386 v1 feature only !
55246 000303B8 <res 00000004>          <1>       u.pgdir:  resd 1 ; 09/03/2015 (page dir addr of process)
55247 000303BC <res 00000004>          <1>       u.ppgdir: resd 1 ; 06/05/2015 (page dir addr of the parent process)
55248 000303C0 <res 00000004>          <1>       u.pbase:  resd 1 ; 20/05/2015 (physical base/transfer address)
55249 000303C4 <res 00000002>          <1>       u.pcount: resw 1 ; 20/05/2015 (byte -transfer- count for page)
55250                                  <1>       ;u.pncount: resw 1
55251                                  <1>              ; 16/06/2015 (byte -transfer- count for page, 'namei', 'mkdir')
55252                                  <1>       ;u.pnbase:  resd 1
55253                                  <1>              ; 16/06/2015 (physical base/transfer address, 'namei', 'mkdir')
55254                                  <1>                     ; 09/06/2015
55255 000303C6 <res 00000001>          <1>       u.kcall:  resb 1 ; The caller is 'namei' (dskr) or 'mkdir' (dskw) sign
55256 000303C7 <res 00000001>          <1>       u.brwdev: resb 1 ; Block device number for direct I/O (bread & bwrite)
55257                                  <1>                     ; 24/07/2015 - 24/06/2015
55258                                  <1>       ;u.args:  resd 1 ; arguments list (line) offset from start of [u.upage]
55259                                  <1>                     ; (arg list/line is from offset [u.args] to 4096 in [u.upage])
55260                                  <1>                     ; ([u.args] points to argument count -argc- address offset)
55261                                  <1>                     ; 24/06/2015
55262                                  <1>       ;u.core:  resd 1 ; physical start address of user's memory space (for sys exec)
55263                                  <1>       ;u.ecore: resd 1 ; physical end address of user's memory space (for sys exec)
55264                                  <1>       ; last error number
55265 000303C8 <res 00000004>          <1>       u.error:  resd 1 ; 28/07/2013 - 09/03/2015
55266                                  <1>                  ; Retro UNIX 8086/386 v1 feature only!
55267                                  <1>                ; 21/09/2015 (debugging - page fault analyze)
55268 000303CC <res 00000004>          <1>       u.pfcount: resd 1 ; page fault count for (this) process (for sys geterr)
55269                                  <1>              ; 19/12/2016 (TRDOS 386)
55270 000303D0 <res 00000004>          <1>       u.tcb:   resd 1 ; Timer callback address/flag which will be used by timer int
55271                                  <1>              ; 13/01/2017 (TRDOS 386)
55272 000303D4 <res 00000001>          <1>       u.t_lock: resb 1 ; Timer interrupt (callback) lock (unlocked by 'sysrele')
55273 000303D5 <res 00000001>          <1>       u.t_mode: resb 1 ; running mode during timer interrupt (0= system, 0FFh= user)
```

```
55274                                  <1>             ; 26/02/2017 (TRDOS 386)
55275 000303D6 <res 00000001>          <1>       u.irqc:       resb 1  ; Count of IRQ callback services (IRQs in use)
55276                                  <1>             ; 28/02/2017 (TRDOS 386)
55277 000303D7 <res 00000001>          <1>       u.irqwait: resb 1 ; IRQ waiting for callback service flag (IRQ number, If > 0)
55278 000303D8 <res 00000001>          <1>       u.r_lock:  resb 1 ; 'IRQ callback service is in progress' flag (IRQ lock)
55279 000303D9 <res 00000001>          <1>       u.r_mode:  resb 1 ; running mode during hadware interrupt
55280                                  <1>             ; 27/02/2017 (TRDOS 386)
55281 000303DA <res 00000001>          <1>       u.fpsave: resb 1  ; TRDOS 386, 'save/restore FPU registers' flag
55282 000303DB <res 00000001>          <1> alignb 4
55283 000303DC <res 0000005E>          <1>       u.fpregs: resb 94 ; 94 byte area for saving and restoring FPU registers
55284                                  <1>
55285 0003043A <res 00000002>          <1> alignb 4
55286                                  <1>
55287                                  <1> U_SIZE      equ $ - user
55288                                  <1>
55289                                  <1> ; 18/10/2015 - Retro UNIX 386 v1 (local variables for 'namei' and 'sysexec')
55290 0003043C <res 00000004>          <1> pcore:  resd 1 ; physical start address of user's memory space (for sys exec)
55291 00030440 <res 00000004>          <1> ecore:  resd 1 ; physical start address of user's memory space (for sys exec)
55292 00030444 <res 00000004>          <1> nbase:      resd 1 ; physical base address for 'namei' & 'sysexec'
55293 00030448 <res 00000002>          <1> ncount: resw 1     ; remain byte count in page for 'namei' & 'sysexec'
55294 0003044A <res 00000002>          <1> argc: resw 1 ; argument count for 'sysexec'
55295 0003044C <res 00000004>          <1> argv: resd 1 ; argument list (recent) address for 'sysexec'
55296                                  <1>
55297                                  <1> ; 03/06/2015 - Retro UNIX 386 v1 Beginning
55298                                  <1> ; 07/04/2013 - 31/07/2013 - Retro UNIX 8086 v1
55299 00030450 <res 00000001>          <1> rw:    resb 1 ;; Read/Write sign (iget)
55300                                  <1>
55301                                  <1> ;alignb 4
55302                                  <1>
55303                                  <1> ; 24/04/2016
55304 00030451 <res 00000004>          <1> ii:         resd 1 ; first cluster of the program file
55305 00030455 <res 00000004>          <1> i.size:           resd 1 ; size of the program file
55306
55307 00030459 <res 00000003>          alignb 4
55308
55309                                  ; 23/05/2016 (TRDOS 386)
55310                                  ; 14/10/2015 (Retro UNIX 386 v1, 'unix386.s')
55311 0003045C <res 00000004>          cr3reg:      resd 1  ; cr3 register content at the beginning of the timer
55312                                                ; (or RTC) interrupt handler.
55313
55314                                  ; 10/12/2016 (callback)
55315                                  ; 10/06/2016
55316                                  ; 19/05/2016
55317                                  ; 18/05/2016 - TRDOS 386 feature only !
55318 00030460 <res 00000100>          timer_set: resd 16*4   ; 256 bytes memory space for 16 timer events
55319                                        ; Timer Event Structure: (max. 16 timer events, 16*16 bytes)
55320                                        ;       Owner:             resb 1 ; 0 = free
55321                                        ;                                 ;>0 = process number (u.uno)
55322                                        ;     Callback:    resb 1 ; 0 = response byte address (phy)
55323                                        ;                           1 = callback address (virtual)
55324                                        ;     Interrupt:      resb 1 ; 0 = Timer interrupt (or none)
55325                                        ;                             ; 1 = Real Time Clock interrupt
55326                                        ;     Response:       resb 1 ; 0 to 255, signal return value
55327                                        ;     Count Limit: resd 1 ; count of ticks (total/set)
55328                                        ;     Current Count:     resd 1 ; count of ticks (current)
55329                                        ;     Response Addr:  resd 1 ; response byte (pointer) address
55330                                        ;                             ; (or callback -user service- address)
55331
55332                                  ;; Memory (swap) Data (11/03/2015)
55333                                  ; 09/03/2015
55334 00030560 <res 00000002>          swpq_count: resw 1 ; count of pages on the swap queue
55335 00030562 <res 00000004>          swp_drv:    resd 1 ; logical drive description table address of the swap drive/disk
55336 00030566 <res 00000004>          swpd_size:  resd 1 ; size of swap drive/disk (volume) in sectors (512 bytes).

55337 0003056A <res 00000004>          swpd_free:  resd 1 ; free page blocks (4096 bytes) on swap disk/drive (logical)
55338 0003056E <res 00000004>          swpd_next:  resd 1 ; next free page block
55339 00030572 <res 00000004>          swpd_last:  resd 1 ; last swap page block
55340
55341 00030576 <res 00000002>          alignb 4
55342
55343                                  ; 10/07/2015
55344                                  ; 28/08/2014
55345 00030578 <res 00000004>          error_code:  resd 1
55346                                  ; 29/08/2014
55347 0003057C <res 00000004>          FaultOffset:       resd 1
55348                                  ; 21/09/2015
55349 00030580 <res 00000004>          PF_Count:   resd 1 ; total page fault count
55350                                                     ; (for debugging - page fault analyze)
55351                                                     ; 'page_fault_handler' (memory.inc)
55352                                                     ; 'sysgeterr' (u9.s)
55353
55354                                  ; 29/04/2016 (TRDOS 386 = TRDOS v2.0)
55355                                  ; 22/08/2015 (Retro UNIX 386 v1)
55356                                  buffer:
55357 00030584 <res 00000008>                resb   8
55358                                  readi_buffer:
55359 0003058C <res 00000200>                resb   512
55360 0003078C <res 00000008>                resb   8
55361                                  writei_buffer:
55362 00030794 <res 00000200>                resb   512
55363                                  ; 24/10/2016
55364 00030994 <res 00000008>                resb   8
55365                                  rw_buffer:
55366 0003099C <res 00000800>                resb   2048  ; general purposed, r/w sector buffer
55367
55368                                  bss_end:
55369
55370                                  ; 27/12/2013
55371                                  _end:  ; end of kernel code
```